# PXL – Digital
# 42TIN280 Software Analysis - Cheat sheet
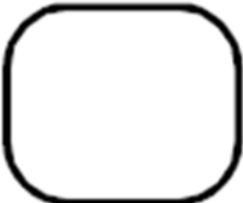
**Semester 01**

**Luc Doumen**
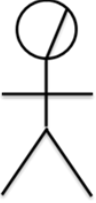
**Nathalie Fuchs**

# UML Domain modeling – Context diagram

| Symbol | Descriptions |
|--------|--------------|
| ▭ | External Entity |
| → | Data Flow |
| ⬭ | Process |
| ⊏ | Data Store |

# UML Domain modeling – BUC diagram (1)

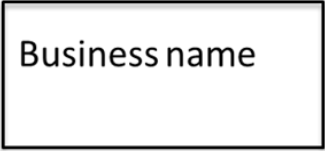| | |
|---|---|
| <br>Business use case | Describes one **business process** as a sequence of (inter)actions between a business actor and a worker as a whole to fulfill a goal of the business actor. (e.g., manual payment processing, expense report approval, manage corporate real estate.)<br><br>The business use case will describe a process that provides **value to the business actor**, and it describes **what the process does**.<br><br>Granularity: document one business use case for every individual business event !!!!! |
| <br>Busines actor | A business actor represents a business role (customer, order intaker,...) that interacts with the business environment/process. |
| <br>Business boundary, subject | A subject of a use case defines and represents boundaries of a business |
| <br>Association | Association is a relationship between classifiers |
| <br>Generalization | A generalization relationship is a relationship in which one model element (the child) is based on another model element (the parent). The child receives all of the attributes, operations, and relationships that are defined in the parent. |

# UML Domain modeling – BUC diagram (1)

BUC A - - - - - - - - BUC B
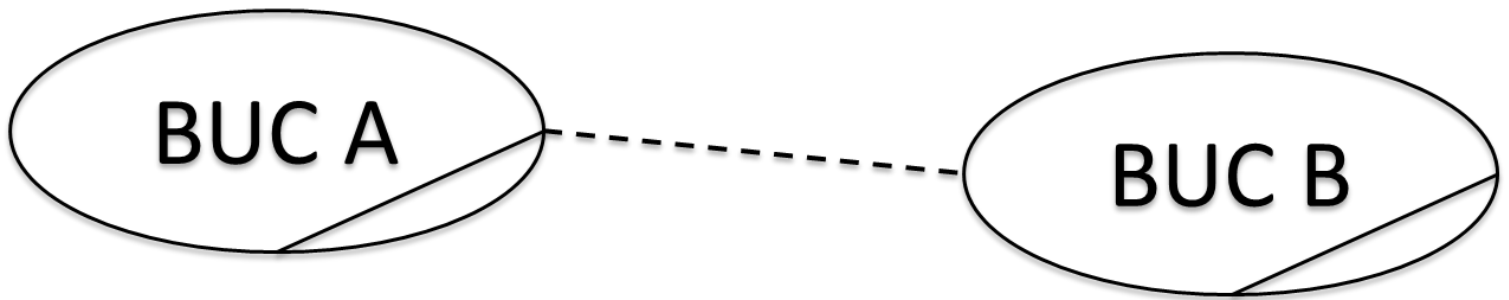
|  | **<Include>** | **<Extend>** |
|---|---|---|
| **Use case A** | Can not without B | Can exist without B. Does not know that B exists |
| **Use case B** | Does not know which use case is calling | Knows to which use case it belongs |

| Adornment | Semantics |
|---|---|
| 0..1 | Zero or 1 |
| 1 | Exactly 1 |
| 0..* | Zero or more |
| * | Zero or more |
| 1..* | 1 or more |
| 1..6 | 1 to 6 |
| 1..3,7..10,15, 19..* | 1 to 3 *or* 7 to 10 *or* 15 exactly *or* 19 to many |

# UML Domain modeling – BUC description (1)

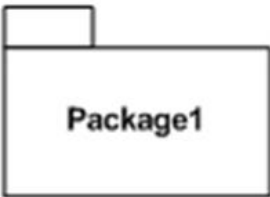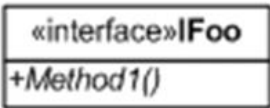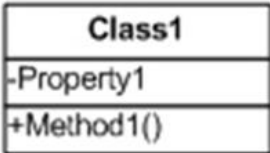| BUC nr. | . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . |
|---|---|
| **BUC name** | – Give each business use case a unique identifier and a name that communicates the functionality.<br><br>– The name should be an active strong verb plus + specific direct object.<br><br>– For example, Record Library Loan, Enroll New Student, Pay Benefit, Generate Sales Report. |
| **Business Event Description** | – A description of the business event to which the business use case responds.<br><br>– A business event is something that happens outside the scope of the business use case or any other organizational work, to which the business use case/work responds.<br><br>– A business event takes place outside the scope of the work/organization.<br><br>– The work/organization learns that an event took place through the arrival of some information (trigger).<br><br>– The work/organization will react to this business event following a business use case scenario. |
| **Triggering Business Event** | – The data or request for a service that arrives from an external source and triggers a response from the work.<br><br>– The trigger may be the arrival of data from an adjacent systems or from a business actor.<br><br>– Alternatively, the trigger may be the arrival of the temporal condition that causes the use case to activate for example, the end of the month.<br><br>– Happen when a pre-arranged time is reached ...<br>    ▪ a periodic occurrence (e.g. 0 a.m. every day),<br>    ▪ a fixed time interval (e.g.24 hours since last occurrence)<br>    ▪ a certain amount of time elapsing since another business event (30 days after sending an invoice.) |
| **Preconditions** | – Sometimes certain conditions must be true before the use case can be executed. For example, a customer has to be registered before he can access his frequent-flyer statement.<br><br>– Note that another business use case usually takes care of the precondition. In the preceding example, the customer would have registered using the Register Passenger use case. |

# UML Domain modeling – BUC description (2)

| Interested stakeholders Secondary actor(s) | | — Stakeholders, actors indirectly involved |
|---|---|---|
| Active stakeholders Primary actor(s) | | — The people, organizations, and/or computer systems that take an active part in the process.<br><br>— Don't think about users just yet; instead, think of the real people who are involved in the work of the business use case.<br><br>— The people, organizations, and/or representatives of computer systems who have knowledge necessary to specify this use case or who have an interest in this use case. |
| **Normal Business Flow** | **Step** | **Action** |
| | 1. | — The normal business flow should describe the normal & most used path of the business performing the process. |
| | 2. | — Typical flow should look like:<br><br>    ▪ Business actor X does action …<br><br>    ▪ Business actor Y does action …<br><br>    ▪ Business actor X does action …<br><br>    ▪ . . . . |
| | 3. | — Typical step should look like: Business actor + active verb + a specific direct object.<br><br>— Write clear, unambiguous steps that are understandable to all stakeholders related to the project.<br><br>— Do not mention the system in the business use case.<br><br>— Adapt a 'sunny day' strategy for elaborating business processes. Thus start with modeling the primary flow. Address branching, concurrency and exceptions as secondary considerations.<br><br>— There are usually between three and fifteen steps.<br><br>— Each step must make the actor move forward in fulfilling his goal. |
| | … | — … |

# UML Domain modeling – BUC description (3)

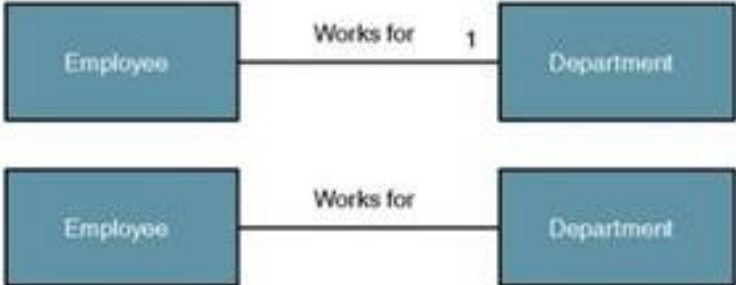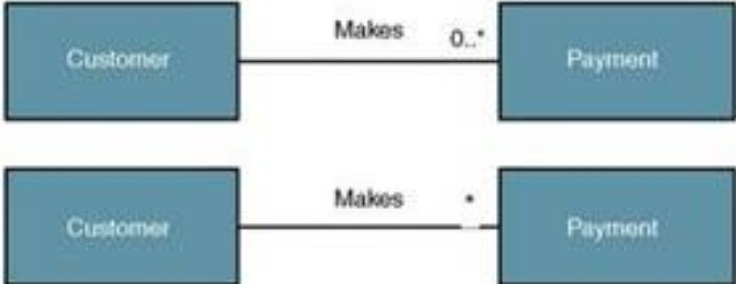| Alternative Business Flows | Step | Action |
|---|---|---|
| | 2a | <ul><li>Alternatives are acceptable variations on the normal case of processing/proceeding.</li><li>If the alternative action is simple, you can make it part of the normal Flow:</li><li>Step 4. Attach the frequent-flyer number to the reservation.</li><li>Alternative 4.1 Issue a lounge invitation if the passenger holds a gold card.</li><li>Tag each alternative to the appropriate step + define condition for alternative clearly.</li></ul> |
| | 2a1 | <ul><li>…</li></ul> |
| | … | <ul><li>…</li></ul> |
| **Exception Business Flows** | **Step** | **Action** |
| | 1a | <ul><li>These are unwanted but necessary variations.</li><li>For example, a customer may have insufficient funds for a withdrawal at an ATM. In this case, the procedure has to offer a lower amount, or offer a loan, or do whatever the stakeholders decide is appropriate.</li><li>Tag each exception to the appropriate step + define condition for exception clearly.</li></ul> |
| | 1b | <ul><li></li></ul> |
| **Outcome (post condition)** | | <ul><li>The desired situation at the end of this use case. Think of it as the stake</li></ul> |

| Shape | Description |
|---|---|
| **Package1** | **Package** <br> A collection of interaces and classes. |
| «interface»**IFoo** <br> +*Method1()* | **Interface** <br> Microsoft guidelines specify that interfaces should start with I. This graphic can also sometimes be used as an abstract class. |
| **Class1** <br> -Property1 <br> +Method1() | **Class** <br> Properties or attributes sit at the top, methods or operations at the bottom. + indicates public and # indicates protected. |

These are both typically drawn vertically:

B ——————▷ A  **Inheritence** - B inherits from A. "is-a" relationship.

B – – – – –▷ A  **Generalization** - B implements A,

A ——————— B  **Association** - A and B call each other

A ——————▹ B  **One way Association.** A can call B's properties/methods, but not visa versa.

A ◇—————— B  **Aggregation** A "has-a"instance of B. B can survive if A is disposed.

A ◆—————— B  **Composition** A has an instance of B, B cannot exist without A.

**A note** <br> Some descriptive text attached to any item.

# UML Domain modeling – Domain model (2)
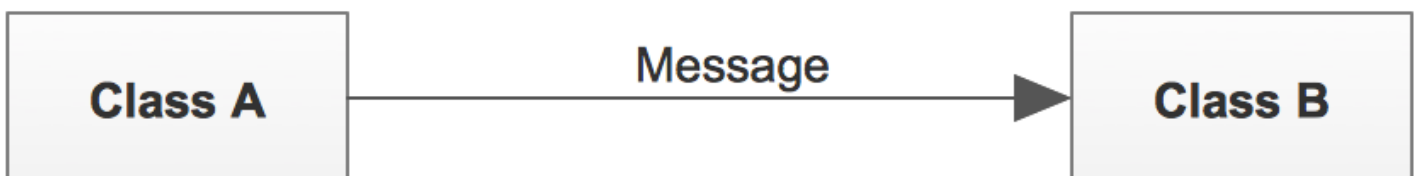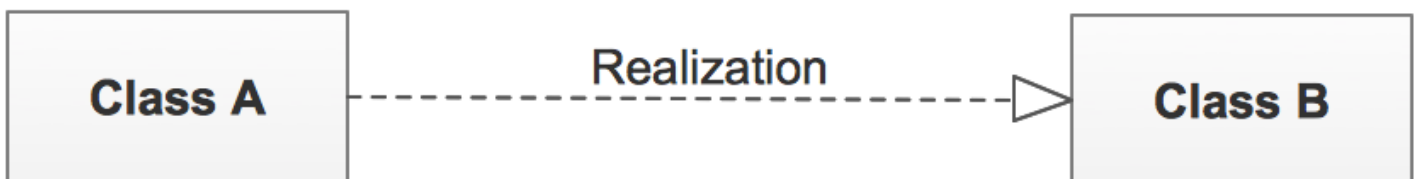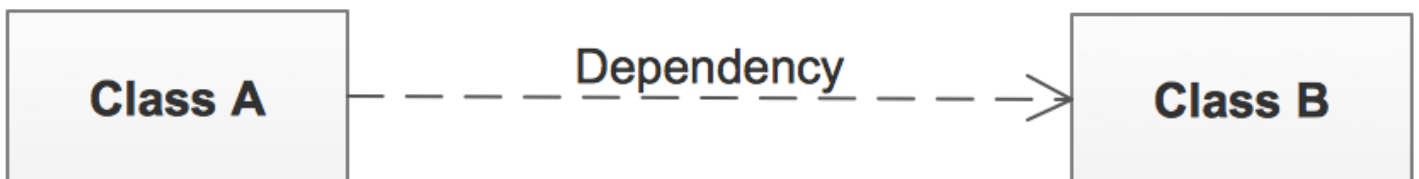
# UML Domain modeling – Domain model (3)



| Multiplicity | UML Multiplicity Notation | Association with Multiplicity | Association Meaning |
|---|---|---|---|
| Exactly 1 | 1 or *leave blank* | Employee — Works for 1 — Department; Employee — Works for — Department | An employee works for one and only one department. |
| Zero or 1 | 0..1 | Employee — Has 0..1 — Spouse | An employee has either one or no spouse. |
| Zero or more | 0..* or * | Customer — Makes 0..* — Payment; Customer — Makes * — Payment | A customer can make no payment up to many payments. |
| 1 or more | 1..* | University — Offers 1..* — Course | A university offers at least 1 course up to many courses. |
| Specific range | 7..9 | Team — Has scheduled 7..9 — Game | A team has either 7, 8, or 9 games scheduled |

1. Prepare problem statement for the system being developed ➜ BUC description

2. Identify concepts (these are the classes & objects) by underlining the nouns

3. Develop a common vocabulary, dictionary, glossary
   a) Make an alphabetic list
   b) Count the occurrences
   c) Make a glossary of terms ➜ domain classes

4. Create a first domain class diagram

5. Identify associations + reading direction between concepts

6. Identify generalization, composition, aggregation

7. Assign attributes to the concepts

8. Check for multiplicities and indicate in domain model

9. Iterate and refine the model

# UML Domain modeling – Domain model (5)

**Association • Aggregation • Composition**

Owner

Pet

Dog
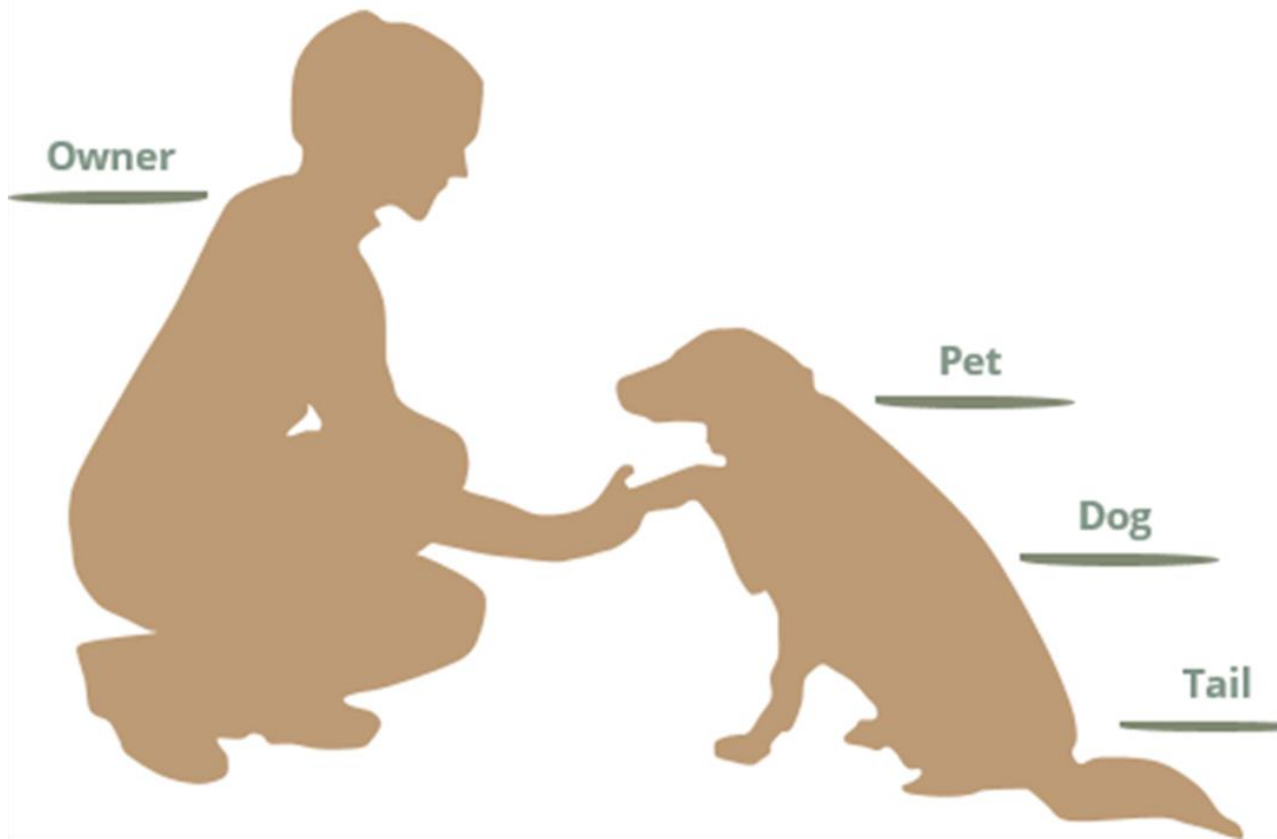
Tail

- owners feed pets, pets please owners (association)
- a tail is a part of both dogs and cats (aggregation / composition)
- a cat is a kind of pet (inheritance / generalization)

# UML Domain modeling – SUC diagram (1)
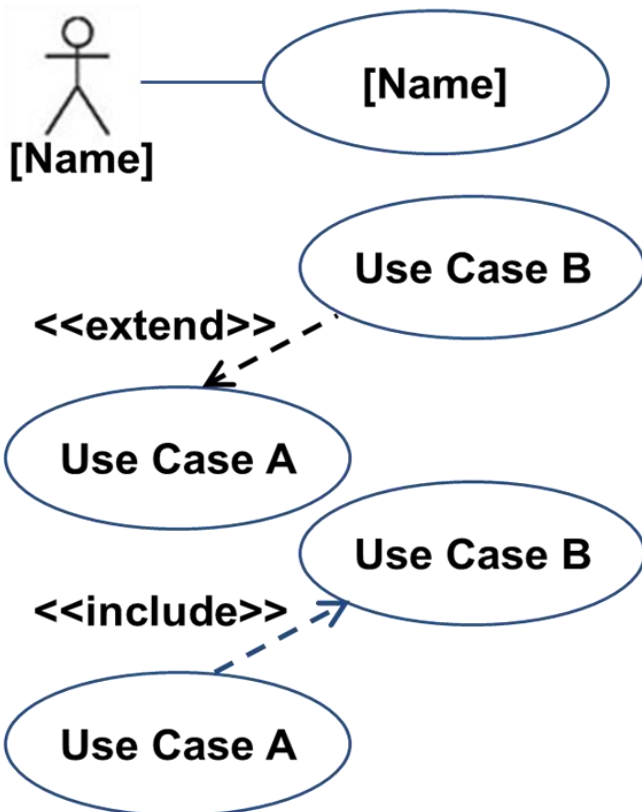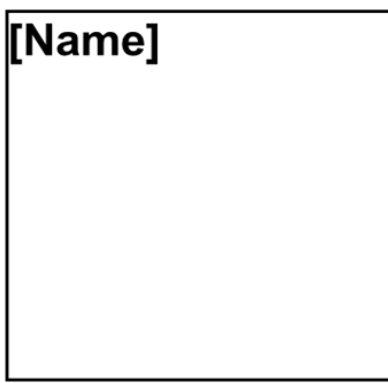


- *Use Case* (verbs)

- *Actor*: persons and/or other systems that interact with the system

- *System Boundary*: determines what's inside (the use cases) and outside (persons and other systems)

- *Communication* between actor & use case

- *Extend*: indicates that the behavior of the extension use case may be inserted in the extended use case under some conditions

- *Include*: indicates that the use case to which the arrow points is included in the use case on the other side of the arrow

# UML Domain modeling – SUC diagram (2)

- An *association* is a relationship between classifiers

- A *generalization* relationship is a relationship in which one model element (the child) is based on another model element (the parent). The child receives all of the attributes, operations, and relationships that are defined in the parent.

# UML Domain modeling – SUC description (1)

| BUC nr. | .......................................................... |
|---|---|
| **BUC name** | – Give each business use case a unique identifier and a name that communicates the functionality.<br><br>– The name should be an active strong verb plus + specific direct object.<br><br>– For example, Record Library Loan, Enroll New Student, Pay Benefit, Generate Sales Report. |
| **Business Event Description** | – A description of the business event to which the business use case responds.<br><br>– A business event is something that happens outside the scope of the business use case or any other organizational work, to which the business use case/work responds.<br><br>– A business event takes place outside the scope of the work/organization.<br><br>– The work/organization learns that an event took place through the arrival of some information (trigger).<br><br>– The work/organization will react to this business event following a business use case scenario. |
| **Triggering Business Event** | – The data or request for a service that arrives from an external source and triggers a response from the work.<br><br>– The trigger may be the arrival of data from an adjacent systems or from a business actor.<br><br>– Alternatively, the trigger may be the arrival of the temporal condition that causes the use case to activate for example, the end of the month.<br><br>– Happen when a pre-arranged time is reached ...<br><br>    ▪ a periodic occurrence (e.g. 0 a.m. every day),<br><br>    ▪ a fixed time interval (e.g.24 hours since last occurrence)<br><br>    ▪ a certain amount of time elapsing since another business event (30 days after sending an invoice.) |
| **Preconditions** | – Sometimes certain conditions must be true before the use case can be executed. For example, a customer has to be registered before he can access his frequent-flyer statement.<br><br>– Note that another business use case usually takes care of the precondition. In the preceding example, the customer would have registered using the Register Passenger use case. |

# UML Domain modeling – SUC description (2)

| | | |
|---|---|---|
| **Interested stakeholders**<br>**Secondary actor(s)** | — Stakeholders, actors indirectly involved | |
| **Active stakeholders**<br>**Primary actor(s)** | — The people, organizations, and/or computer systems that take an active part in the process.<br><br>— Don't think about users just yet; instead, think of the real people who are involved in the work of the business use case.<br><br>— The people, organizations, and/or representatives of computer systems who have knowledge necessary to specify this use case or who have an interest in this use case. | |

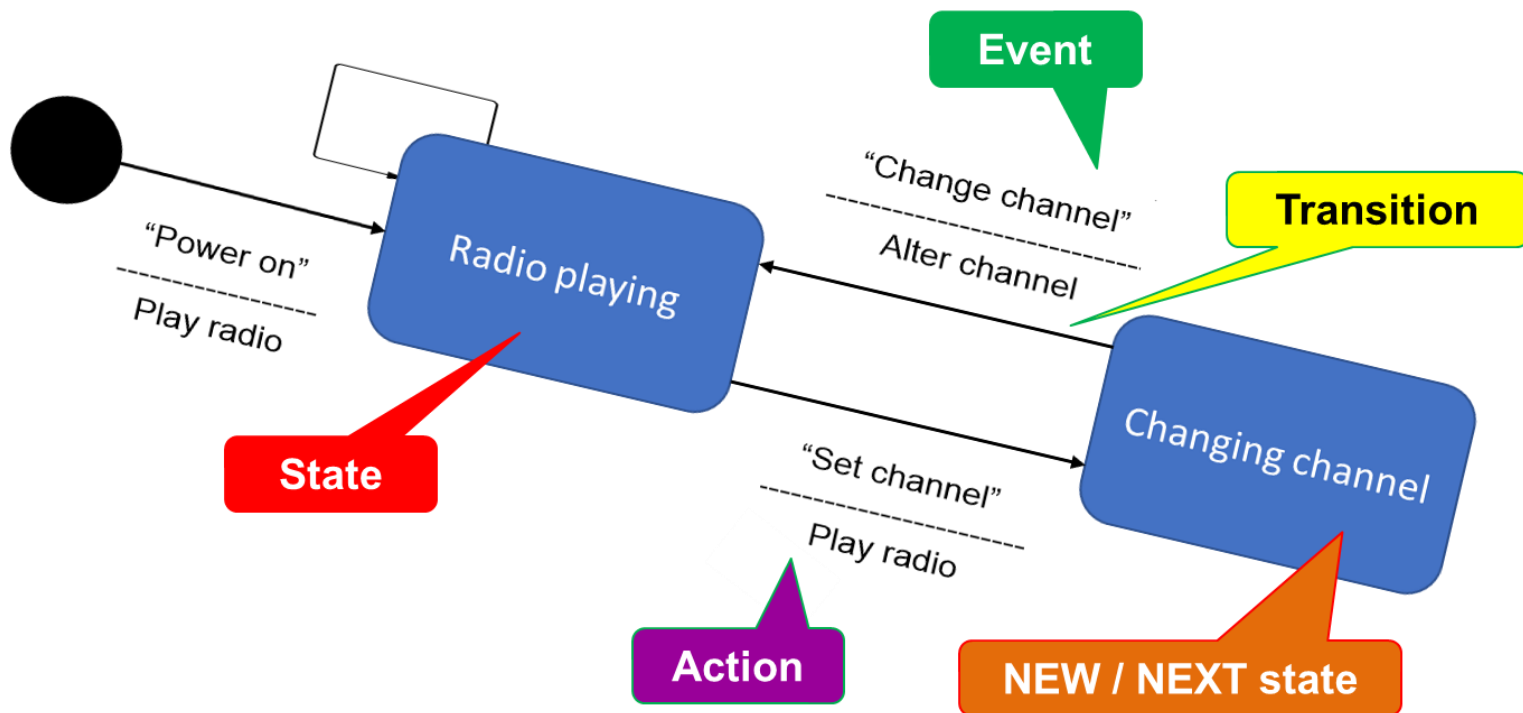| **Normal Business Flow** | **Step** | **Action** |
|---|---|---|
| | 1. | — The normal business flow should describe the normal & most used path of the business performing the process. |
| | 2. | — Typical flow should look like:<br><br>    ▪ Business actor  X  does action …<br><br>    ▪ Business actor  Y  does action …<br><br>    ▪ Business actor  X  does action …<br><br>    ▪ . . . . |
| | 3. | — Typical step should look like: Business actor + active verb + a specific direct object.<br><br>— Write clear, unambiguous steps that are understandable to all stakeholders related to the project.<br><br>— Do not mention the system in the business use case.<br><br>— Adapt a 'sunny day' strategy for elaborating business processes. Thus start with modeling the primary flow.  Address branching, concurrency and exceptions as secondary considerations.<br><br>— There are usually between three and fifteen steps.<br><br>— Each step must make the actor move forward in fulfilling his goal. |
| | … | — … |

# UML Domain modeling – SUC description (3)

| Alternative Business Flows | Step | Action |
|---|---|---|
| | 2a | — Alternatives are acceptable variations on the normal case of processing/proceeding.<br>— If the alternative action is simple, you can make it part of the normal Flow:<br>— Step 4. Attach the frequent-flyer number to the reservation.<br>— Alternative 4.1 Issue a lounge invitation if the passenger holds a gold card.<br>— Tag each alternative to the appropriate step + define condition for alternative clearly. |
| | 2a1 | — … |
| | … | — … |
| Exception Business Flows | Step | Action |
| | 1a | — These are unwanted but necessary variations.<br>— For example, a customer may have insufficient funds for a withdrawal at an ATM. In this case, the procedure has to offer a lower amount, or offer a loan, or do whatever the stakeholders decide is appropriate.<br>— Tag each exception to the appropriate step + define condition for exception clearly. |
| | 1b | — |
| Outcome (post condition) | — The desired situation at the end of this use case. Think of it as the stake | |

# UML Domain modeling – Activity diagram
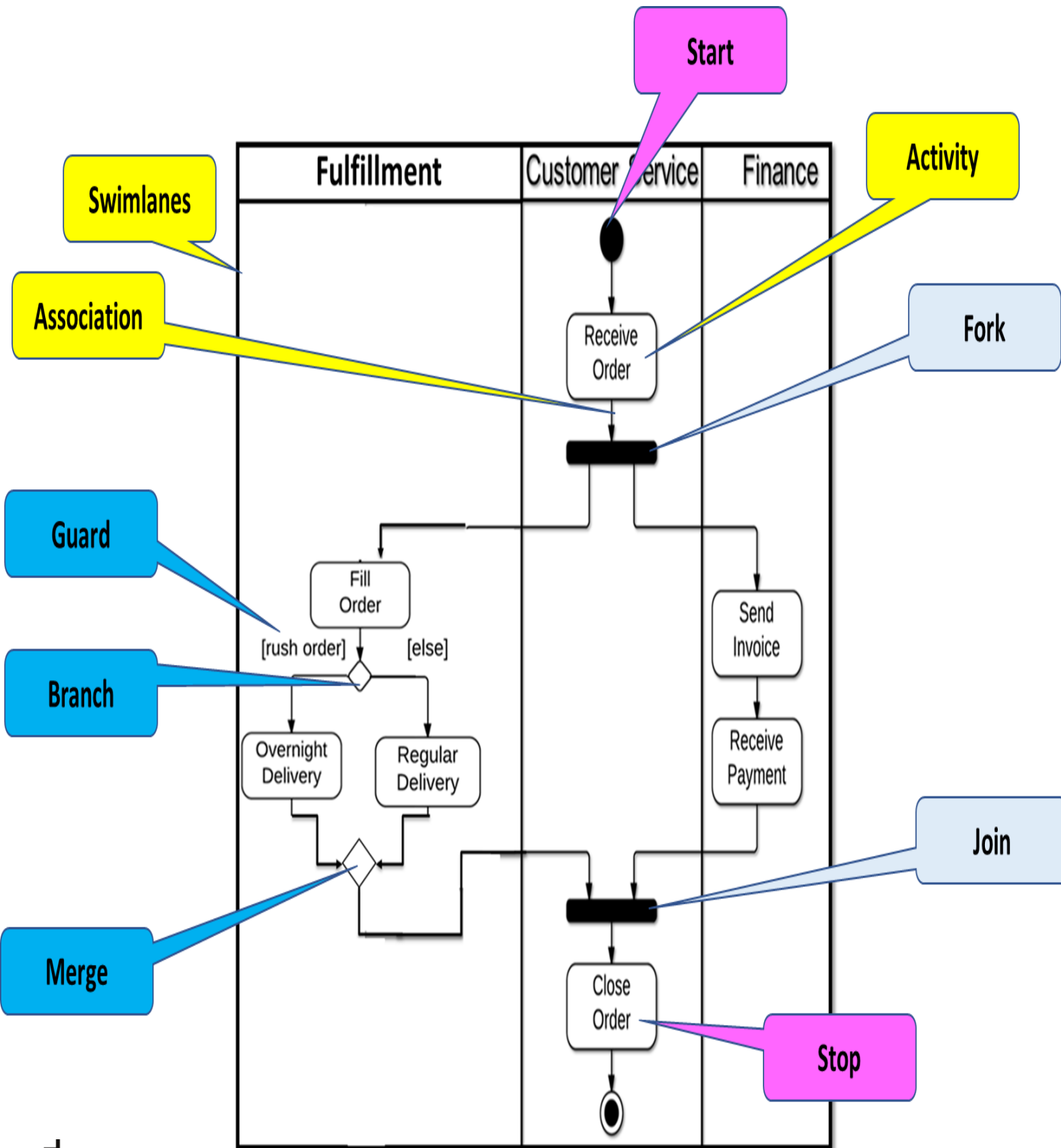
# UML Domain modeling – State transition diagram & table



| EVENT / STATE | Power on | Change channel | Set channel |
|---|---|---|---|
| NULL | A: Play radio<br>$S_{new}$: Radio playing | n.a. | n.a. |
| Radio playing | n.a. | A: Alter channel<br>$S_{new}$: Channel changing | n.a. |
| Channel changing | n.a. | n.a. | A: Play radio<br>$S_{new}$: Radio playing |