

CTIS359

Principles of Software Engineering

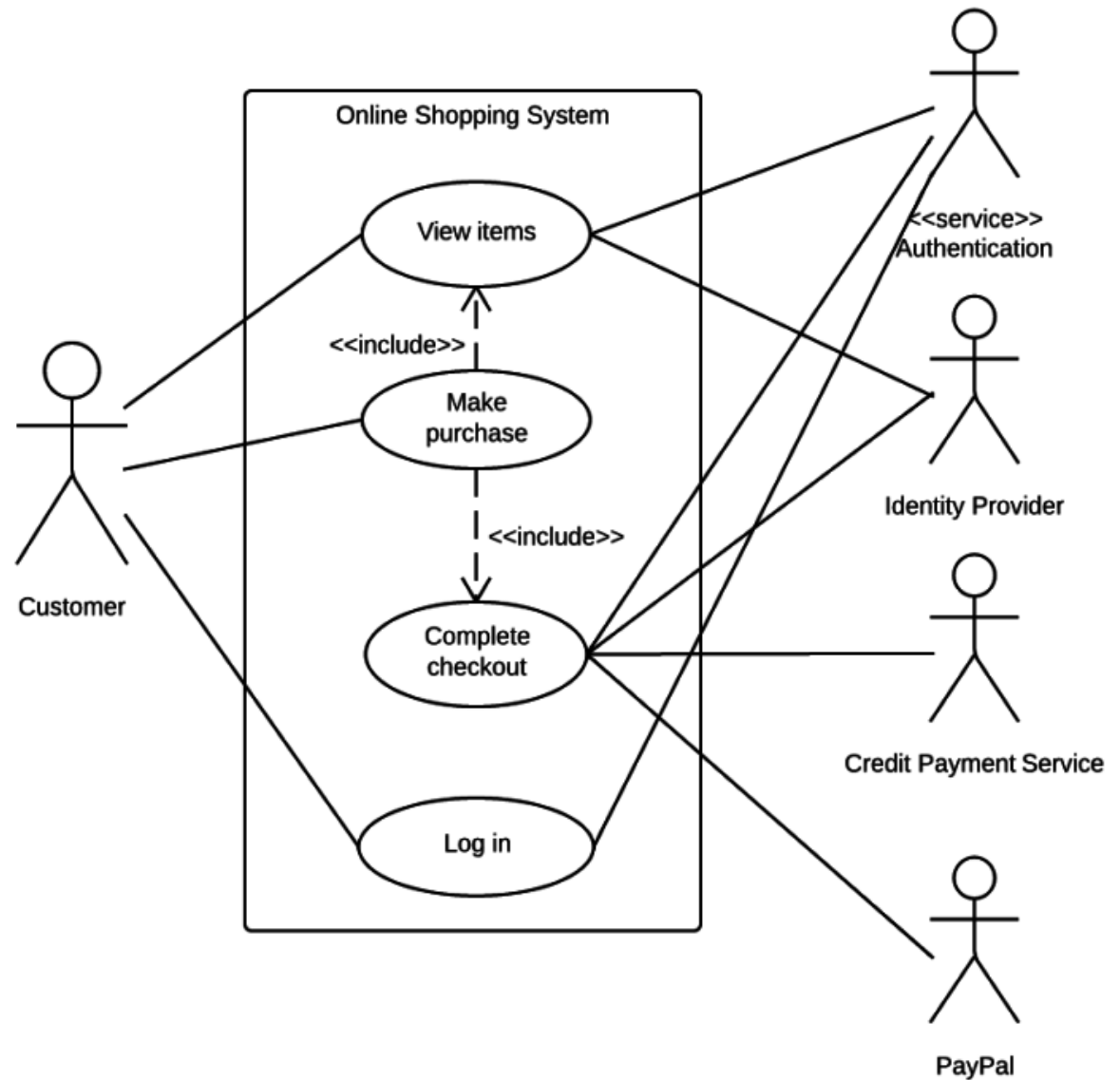
**Introduction to Project Planning
&
Project Effort Estimation
Use Case (UC) Points**

“Whenever You Can, Count.”
Francis Galton

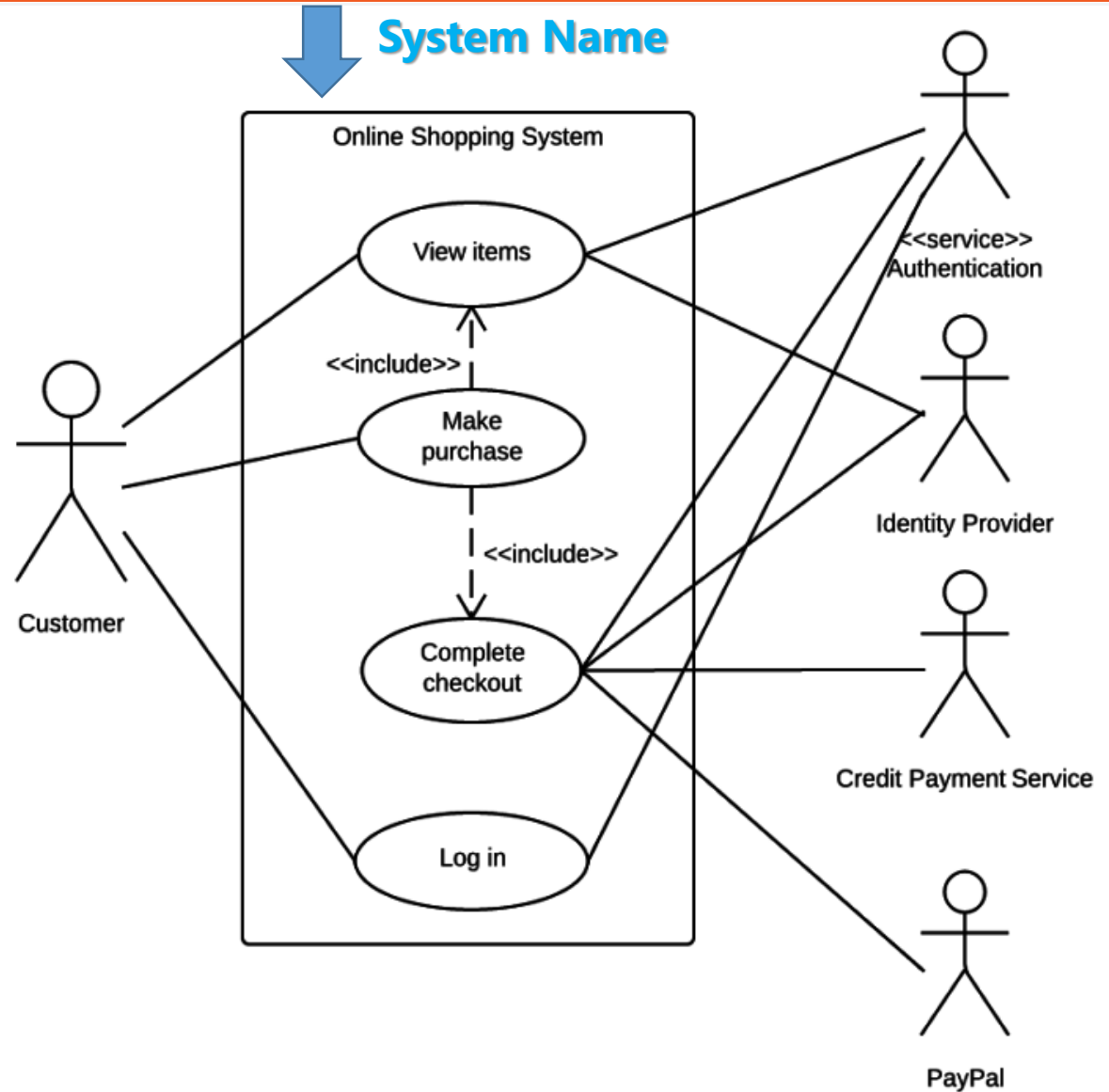
Today

- UML Use Case (UC) Modelling Review
- Use Case Points (UCP) by example
 - Counting rules

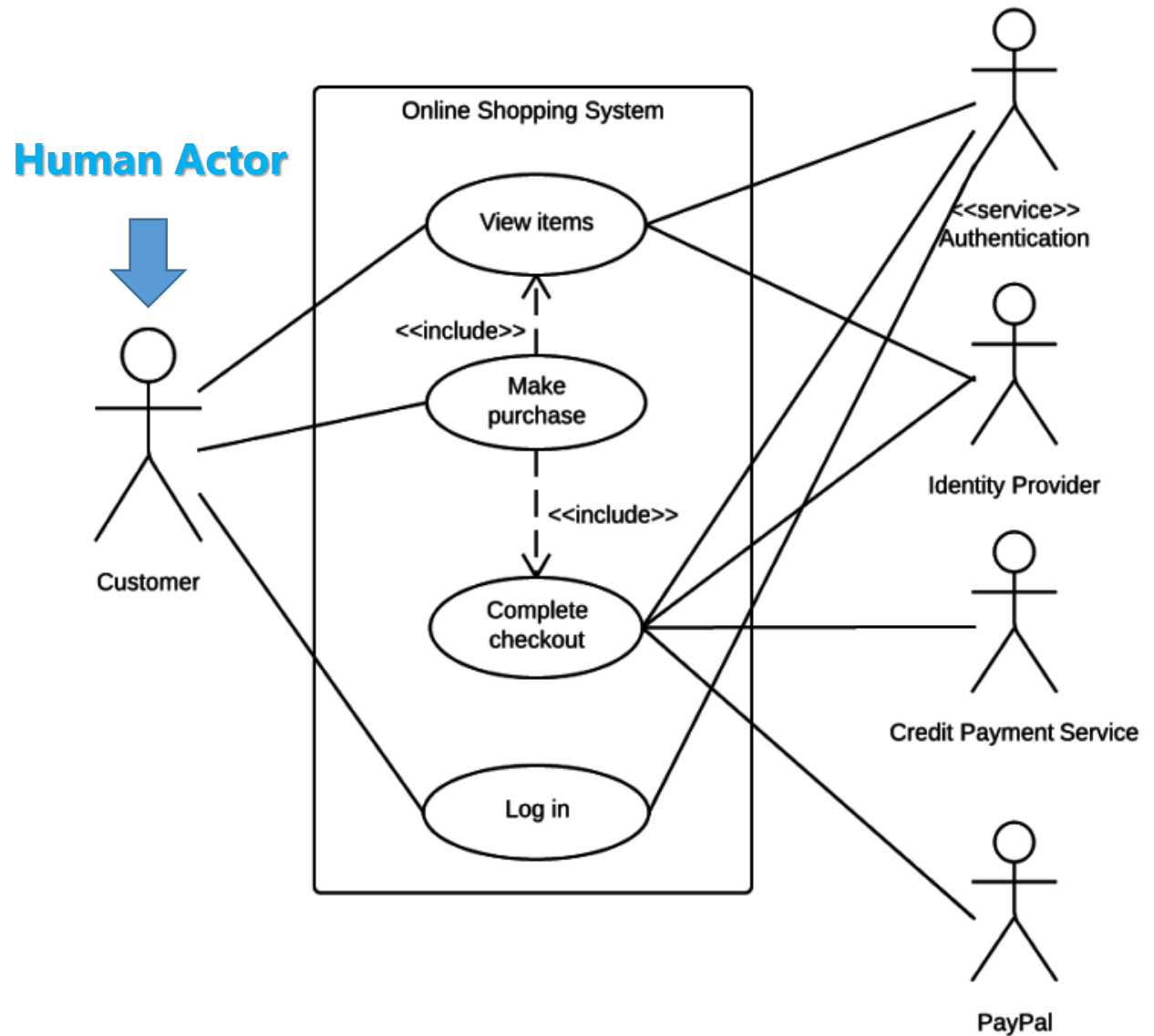
Use Case Diagram - Review



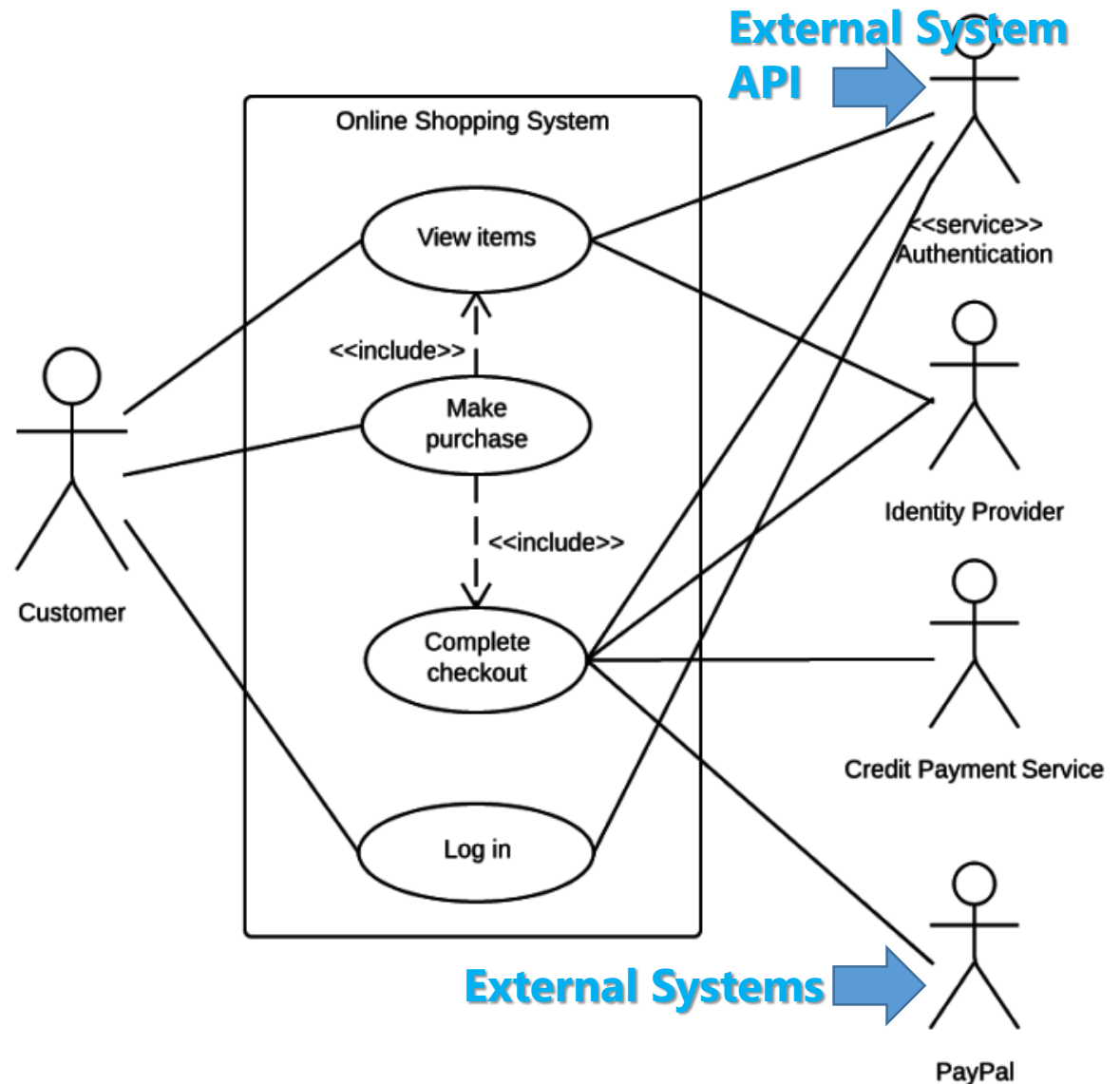
Use Case Diagram - Review



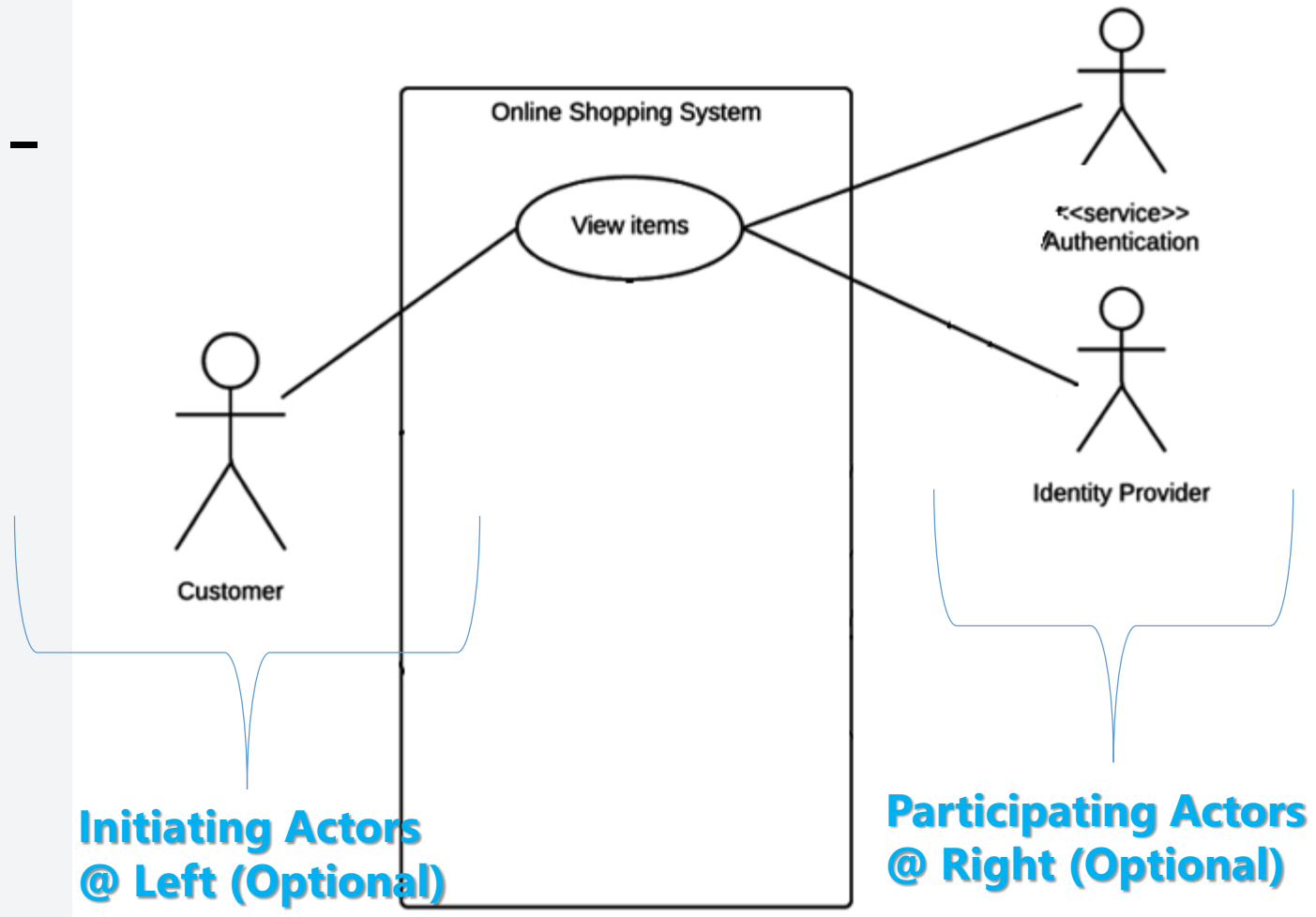
Use Case Diagram - Review



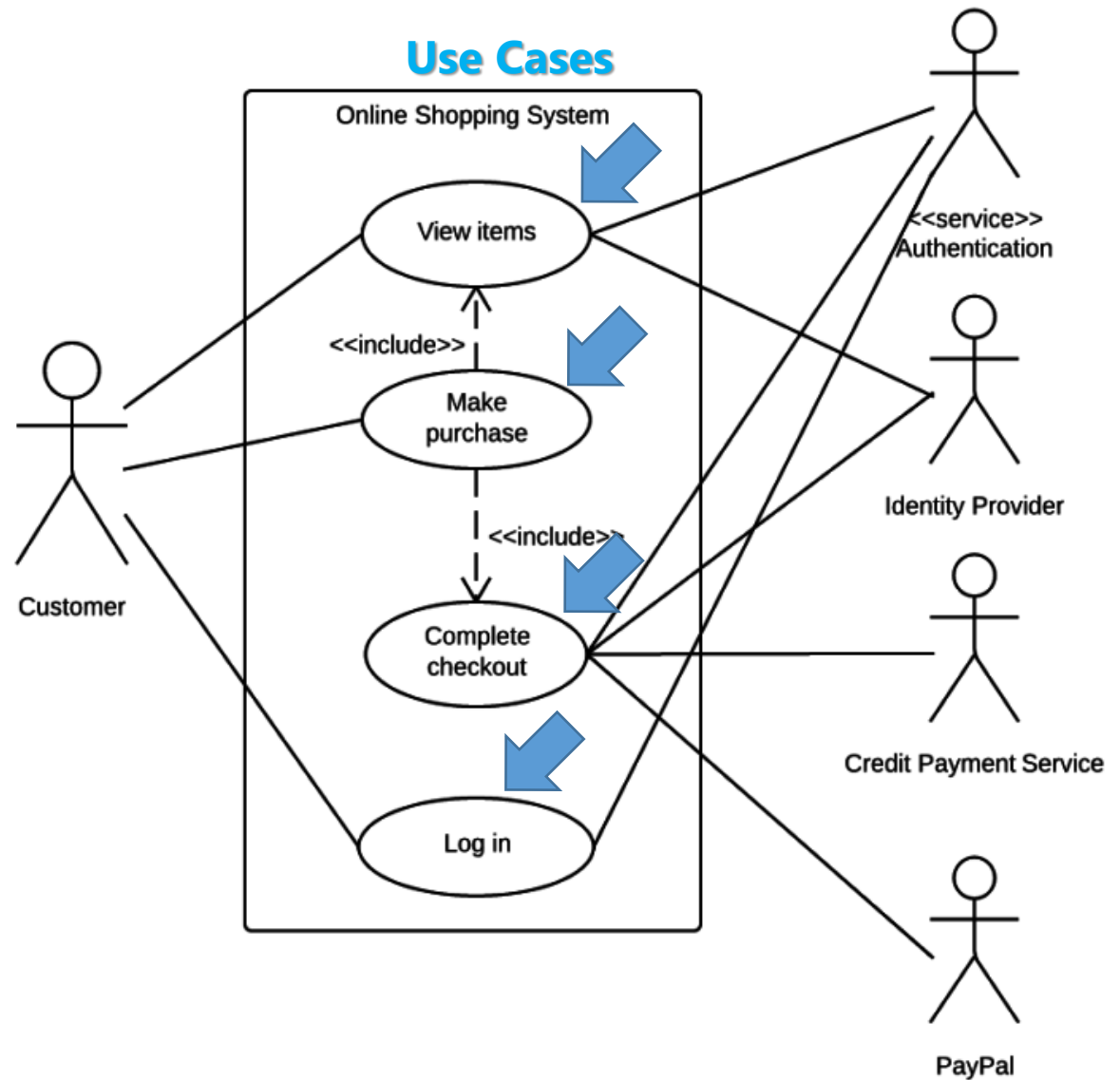
Use Case Diagram - Review



Use Case Diagram - Review



Use Case Diagram - Review



Use Case Descriptions- Review

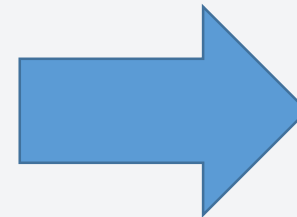
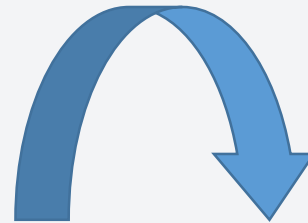
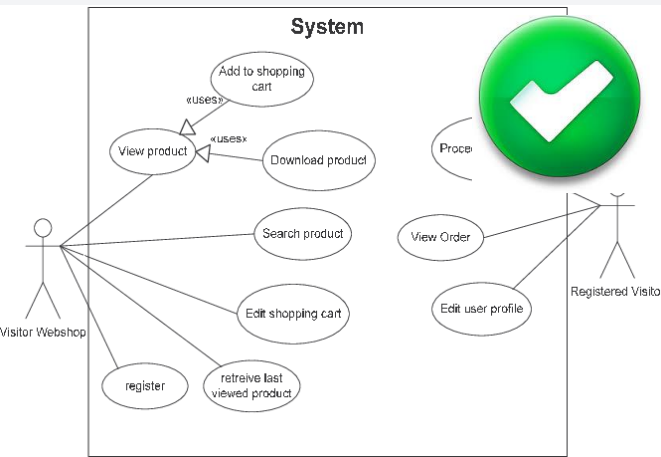
| | |
|---|--|
| ID: | 1 |
| Title: | View Items |
| Description: | |
| Primary Actor: | |
| Preconditions: | |
| Post-conditions: | |
| Main Success Scenario: (Main Flow) | 1.1. 2. 3. 4. 5. |
| Extensions: (Alternative flow) | <Use 2a, 2a 1, 2a 2 numbering> 2a1. 2a2 3a3 |
| Frequency of Use: | A few times every quarter |
| Priority: | <P1, P2, P3 – High, Medium, Low> |

Use-Case Points (UCP)

- There are a variety of ways to **estimate the time** required to build a system.
- Because the use-case points is UC driven, we use an approach that is based on UCs:
 - *Use-case points*, originally developed by Gustav Karner of Objectory AB (Objectory AB was acquired by Rational in 1995 and Rational is now part of IBM.), are based on unique features of UCs and object orientation.
 - From a practical point of view, to **estimate effort** using UCPs, the UCs and the UC diagram **MUST HAVE BEEN** created.



Use-Case Points (UCP)



**UCP
Counting
Technique**

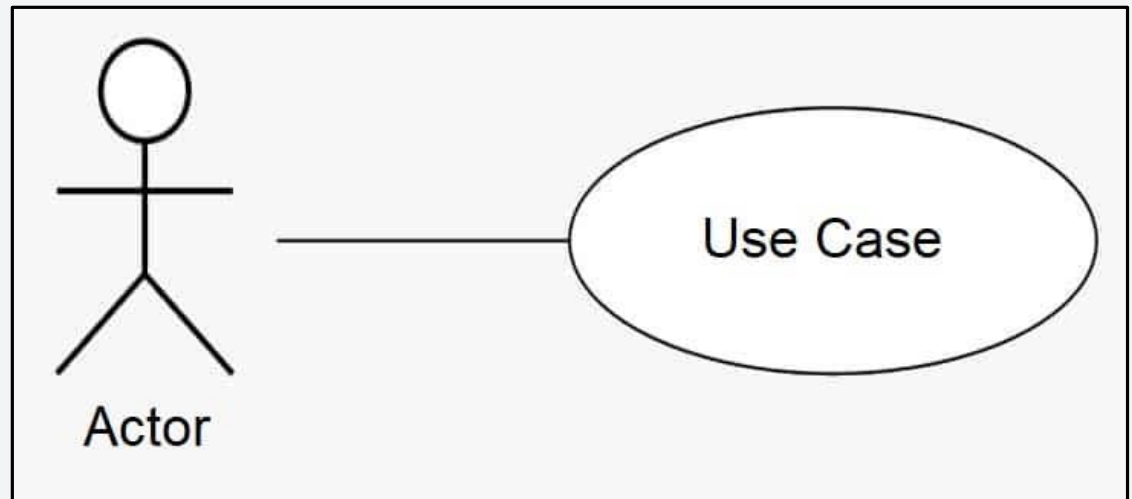
**Estimated
Effort**



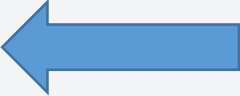
Use-Case Points (UCP)

- UC models have two **primary constructs**:

- Actors
- Use cases



Use-Case Points (UCP)

- UC models have two **primary constructs**:
 - Actors 
 - Use cases
- An *actor* represents **a role** (secretary or manager) that a user of the system plays, not a specific user.
- Actors can also represent **other systems** that will interact with the system under development.

Use-Case Points (UCP)

- For UCP estimation purposes, actors can be classified as
 - **Simple actors:** are **separate systems** with which the current system **must** communicate through **a well-defined application program interface (API)**.
 - **Average actors:** are **separate systems** that interact with the current system **using standard communication protocols**, such as TCP/IP, FTP, or HTTP, or an **external database** that can be accessed using standard SQL.
 - **Complex actors:** are typically **end users** communicating with the system.

Use-Case Points (UCP)

- Once all of the actors have been categorized as being **simple**, **average**, or **complex**, the project manager (PM) counts the # of actors in each category and enters the values into the unadjusted actor-weighting table contained in the UCPs–estimation worksheet.

Unadjusted Actor Weighting Table:

| Actor Type | Description | Weighting Factor | Number | Result |
|-------------------------------------|---|------------------|--------|--------|
| Simple | External System with well-defined API | 1 | | |
| Average | External System using a protocol-based interface, e.g., HTTP, TCT/IP, or a database | 2 | | |
| Complex | Human | 3 | | |
| Unadjusted Actor Weight Total (UAW) | | | | |

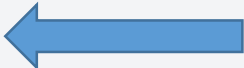
Use-Case Points (UCP)

- The PM then computes the *Unadjusted Actor Weight Total (UAW)*.
 - This is computed by summing the individual results that were computed by multiplying the weighting factor by the # of actors of each type.
 - **Ex:** If we assume that the UC diagram has 0 simple, 0 average, and 4 complex actors that interact with the system being developed, the $UAW=12$.

Unadjusted Actor Weighting Table:

| Actor Type | Description | Weighting Factor | Number | Result |
|--|---|------------------|--------|--------|
| Simple | External system with well-defined API | 1 | 0 | 0 |
| Average | External system using a protocol-based interface, e.g., HTTP, TCT/IP, or a database | 2 | 0 | 0 |
| Complex | Human | 3 | 4 | 12 |
| <i>Unadjusted Actor Weight Total (UAW)</i> | | | | 12 |

Use-Case Points (UCP)

- UC models have two primary constructs:
 - Actors
 - Use cases 
- A *use case* represents **a major business process** that the system will perform that benefits the actor(s) in some manner.
- Depending on the # of unique transactions that the UC must address, a UC can be categorized as:
 - Simple
 - Average
 - Complex

Use-Case Points (UCP)

- **A simple UC:** If the UC supports 1 to 3 transactions.
- **An average UC:** If the UC supports 4 to 7 transactions.
- **A complex UC:** If it supports more than 7 transactions.
- Once all of the UCs have been successfully categorized, the PM enters the # of each type of UC into the unadjusted use-case weighting table contained in the UC point–estimation worksheet.

Unadjusted Use Case Weighting Table:

| Use-Case Type | Description | Weighting Factor | Number | Result |
|--|------------------|------------------|--------|--------|
| Simple | 1–3 transactions | 5 | | |
| Average | 4–7 transactions | 10 | | |
| Complex | >7 transactions | 15 | | |
| <i>Unadjusted Use-Case Weight Total (UUCW)</i> | | | | |

Use-Case Points (UCP)

- **Transactions.**
- Each UC is defined as *simple*, *average* or *complex*, depending on the transaction in the UC description, including the secondary scenarios.
- A transaction is a set of activities, which is either performed entirely, or not at all.
- Counting the # of transactions can be done by counting the # of UC steps.



Use-Case Points (UCP)

- By multiplying by the appropriate weights and summing the results, we get the value for the *unadjusted use-case weight total* (UUCW).
- **Ex:** If we assume that we have 3 simple UCs, 4 average UCs, and 1 complex UC, the value for the unadjusted use-case weight total is 70

Unadjusted Use-Case Weighting Table:

| Use Case Type | Description | Weighting Factor | Number | Result |
|---|------------------|------------------|--------|--------|
| Simple | 1–3 transactions | 5 | 3 | 15 |
| Average | 4–7 transactions | 10 | 4 | 40 |
| Complex | >7 transactions | 15 | 1 | 15 |
| Unadjusted Use Case Weight Total (UUCW) | | | | 70 |

Use-Case Points (UCP)

- Next, the PM computes the value of the *unadjusted use-case points (UUCP)* by simply summing the *unadjusted actor weight total* and the unadjusted *use-case* weight total.
- In this case the value of the UUCP equals 82

Unadjusted Actor Weighting Table:

| Actor Type | Description | Weighting Factor | Number | Result |
|--|---|------------------|--------|-----------|
| Simple | External system with well-defined API | 1 | 0 | 0 |
| Average | External system using a protocol-based interface, e.g., HTTP, TCT/IP, or a database | 2 | 0 | 0 |
| Complex | Human | 3 | 4 | 12 |
| <i>Unadjusted Actor Weight Total (UAW)</i> | | | | 12 |

Unadjusted Use-Case Weighting Table:

| Use Case Type | Description | Weighting Factor | Number | Result |
|--|------------------|------------------|--------|-----------|
| Simple | 1–3 transactions | 5 | 3 | 15 |
| Average | 4–7 transactions | 10 | 4 | 40 |
| Complex | >7 transactions | 15 | 1 | 15 |
| <i>Unadjusted Use Case Weight Total (UUCW)</i> | | | | 70 |

Use-Case Points (UCP)

- UCP-based estimation also has a set of factors that are used to **adjust the UCP value**. Obviously, these types of factors can affect the effort that a team requires to develop a system.
- In this case, there are two sets of factors:
 - **Technical complexity Factors (TCFs):** 13 separate technical factors
 - **Environmental Factors (EFs):** 8 separate environmental factors
- The purpose of these factors is to allow the project as a whole to be evaluated for **the complexity of the system** being developed and **the experience levels of the development staff**, respectively.
- Each of these factors is **assigned a value between 0 and 5**.
 - 0 indicating that the factor is **irrelevant**
 - 5 indicating that the factor is **essential** for the system to be successful.

Use-Case Points (UCP)

- The assigned values are then multiplied by their respective weights. These weighted values are then summed up to create a *technical factor value* (*TFactor*) and an *environmental factor value* (*EFactor*).

Technical Complexity Factors:

| Factor Number | Description | Weight | Assigned Value (0–5) | Weighted Value | Notes |
|---|--|--------|----------------------|----------------|-------|
| T1 | Distributed system | 2.0 | | | |
| T2 | Response time or throughput performance objectives | 1.0 | | | |
| T3 | End-user online efficiency | 1.0 | | | |
| T4 | Complex internal processing | 1.0 | | | |
| T5 | Reusability of code | 1.0 | | | |
| T6 | Ease of installation | 0.5 | | | |
| T7 | Ease of use | 0.5 | | | |
| T8 | Portability | 2.0 | | | |
| T9 | Ease of change | 1.0 | | | |
| T10 | Concurrency | 1.0 | | | |
| T11 | Special security objectives included | 1.0 | | | |
| T12 | Direct access for third parties | 1.0 | | | |
| T13 | Special user training required | 1.0 | | | |
| <i>Technical Factor Value (TFactor)</i> | | | | | |

$$\text{Technical Complexity Factor (TCF)} = 0.6 + (0.01 * \text{TFactor})$$

Use-Case Points (UCP)

- The **technical factors** include the following:
 - Whether the system is going to be a distributed system
 - The importance of response time
 - The efficiency level of the end user using the system
 - The complexity of the internal processing of the system
 - The importance of code reuse
 - How easy the installation process has to be
 - The importance of the ease of using the system
 - How important it is for the system to be able to be ported to another platform
 - Whether system maintenance is important
 - Whether the system is going to have to handle parallel and concurrent processing
 - The level of special security required
 - The level of system access by third parties
 - Whether special end user training is to be required.

Use-Case Points (UCP)

- The assigned values are then multiplied by their respective weights. These weighted values are then summed up to create *a technical factor value (TFactor)* and *an environmental factor value (EFactor)*.

Technical Complexity Factors:

| Factor Number | Description | Weight | Assigned Value (0–5) | Weighted Value | Notes |
|----------------------------------|--|--------|----------------------|----------------|-------|
| T1 | Distributed system | 2.0 | 0 | 0 | |
| T2 | Response time or throughput performance objectives | 1.0 | 5 | 5 | |
| T3 | End-user online efficiency | 1.0 | 3 | 3 | |
| T4 | Complex internal processing | 1.0 | 1 | 1 | |
| T5 | Reusability of code | 1.0 | 1 | 1 | |
| T6 | Ease of installation | 0.5 | 2 | 1 | |
| T7 | Ease of use | 0.5 | 4 | 2 | |
| T8 | Portability | 2.0 | 0 | 0 | |
| T9 | Ease of change | 1.0 | 2 | 2 | |
| T10 | Concurrency | 1.0 | 0 | 0 | |
| T11 | Special security objectives included | 1.0 | 0 | 0 | |
| T12 | Direct access for third parties | 1.0 | 0 | 0 | |
| T13 | Special user training required | 1.0 | 0 | 0 | |
| Technical Factor Value (TFactor) | | | | 15 | |

Use-Case Points (UCP)

- The assigned values are then multiplied by their respective weights. These weighted values are then summed up to create a *technical factor value* (*TFactor*) and an *environmental factor value* (*EFactor*).

Environmental Factors:

| Factor Number | Description | Weight | Assigned Value (0–5) | Weighted Value | Notes |
|---|--|--------|----------------------|----------------|-------|
| E1 | Familiarity with system development process being used | 1.5 | | | |
| E2 | Application experience | 0.5 | | | |
| E3 | Object-oriented experience | 1.0 | | | |
| E4 | Lead analyst capability | 0.5 | | | |
| E5 | Motivation | 1.0 | | | |
| E6 | Requirements stability | 2.0 | | | |
| E7 | Part time staff | –1.0 | | | |
| E8 | Difficulty of programming language | –1.0 | | | |
| <i>Environmental Factor Value (EFactor)</i> | | | | | |

Use-Case Points (UCP)

- The **environmental factors** include the following:
 - The level of experience the development staff has with the development process being used
 - The application being developed
 - The level of **object-oriented** experience
 - The level of capability of the lead analyst
 - The level of motivation of the development team to deliver the system
 - The stability of the requirements
 - Whether part-time staff have to be included as part of the development team
 - The difficulty of the PL being used to implement the system

Use-Case Points (UCP)

- The assigned values are then multiplied by their respective weights. These weighted values are then summed up to create a *technical factor value* (TFactor) and an *environmental factor value* (EFactor).

Environmental Factors:

| Factor Number | Description | Weight | Assigned Value (0–5) | Weighted Value | Notes |
|---|--|--------|----------------------|----------------|-------|
| E1 | Familiarity with system development process being used | 1.5 | 4 | 6 | |
| E2 | Application experience | 0.5 | 4 | 2 | |
| E3 | Object-oriented experience | 1.0 | 4 | 4 | |
| E4 | Lead analyst capability | 0.5 | 5 | 2.5 | |
| E5 | Motivation | 1.0 | 5 | 5 | |
| E6 | Requirements stability | 2.0 | 5 | 10 | |
| E7 | Part-time staff | –1.0 | 0 | 0 | |
| E8 | Difficulty of programming language | –1.0 | 4 | –4.0 | |
| <i>Environmental Factor Value (EFactor)</i> | | | | 25.5 | |

Use-Case Points (UCP)

Technical Complexity Factor (TCF) = $0.6 + (0.01 * TFactor)$

Technical Complexity Factor (TCF) = $0.6 + (0.01 * 15)$

Technical Complexity Factor (TCF) = **0.75**

Use-Case Points (UCP)

$$\text{Environmental Factor (EF)} = 1.4 + (-0.03 * \text{EFactor})$$

$$\text{Environmental Factor (EF)} = 1.4 + (-0.03 * 25.5)$$

$$\text{Environmental Factor (EF)} = 0.635$$

$$\text{Adjusted Use Case Points (UCP)} = \text{UUCP} * \text{TCF} * \text{EF}$$

$$\text{Unadjusted Use Case Points (UUCP)} = \text{UAW} + \text{UUCW}$$

$$\text{Unadjusted Use Case Points (UUCP)} = 12 + 70$$

$$\text{Unadjusted Use Case Points (UUCP)} = 82$$

$$\text{Adjusted Use Case Points (UCP)} = 82 * 0.75 * 0.635$$

$$\text{Adjusted Use Case Points (UCP)} = 39.0525$$

Use-Case Points (UCP)

Effort (in Personal Hours) = UCP * PHM

Effort (in Personal Hours) = 39.0525 * 20

Effort = **781.05** Person Hours (PH)

Use-Case Points (UCP)

- Now that we know the estimated size of the system by means of the value of the **adjusted UC points**, we are ready to estimate the *effort* required to build the system.
- In Karner's original work, he suggested simply **multiplying the # of UC points by 20** to estimate the # of person-hours required to build the system.
- However, based on additional experiences using UCP, a decision rule to determine the value of the **person-hours multiplier (PHM)** has been created that suggests using either 20 or 28, based on the values assigned to the **individual environmental factors**.

Use-Case Points (UCP)

- The decision rule is:
 - **If** the sum of (# of Efactors E1 through E6 assigned value < 3) and (# of Efactors E7 and E8 assigned value > 3) = < 2
 - PHM = 20
 - **Else If** the sum of (# of Efactors E1 through E6 assigned value < 3) and (# of Efactors E7 and E8 assigned value > 3) = 3 or 4
 - PHM = 28
 - **Else**
 - Rethink project; it has too high of a risk for failure.

Use-Case Points (UCP)

- $E1 = 4 < 3 \rightarrow 0$
- $E2 = 4 < 3 \rightarrow 0$
- $E3 = 4 < 3 \rightarrow 0$
- $E4 = 5 < 3 \rightarrow 0$
- $E5 = 5 < 3 \rightarrow 0$
- $E6 = 5 < 3 \rightarrow 0$
- $E7 = 0 > 3 \rightarrow 0$
- $E8 = 4 > 3 \rightarrow 1$
- The sum of the number EFactors is 1
- Thus, the system should use a PHM of 20.

References

- Systems Analysis and Design: An Object Oriented Approach with UML, 5th Edition, Alan Dennis, Barbara Haley Wixom, David Tegarden, 2015.
- Use Case Points An Estimation Approach, Banerjee 2001.