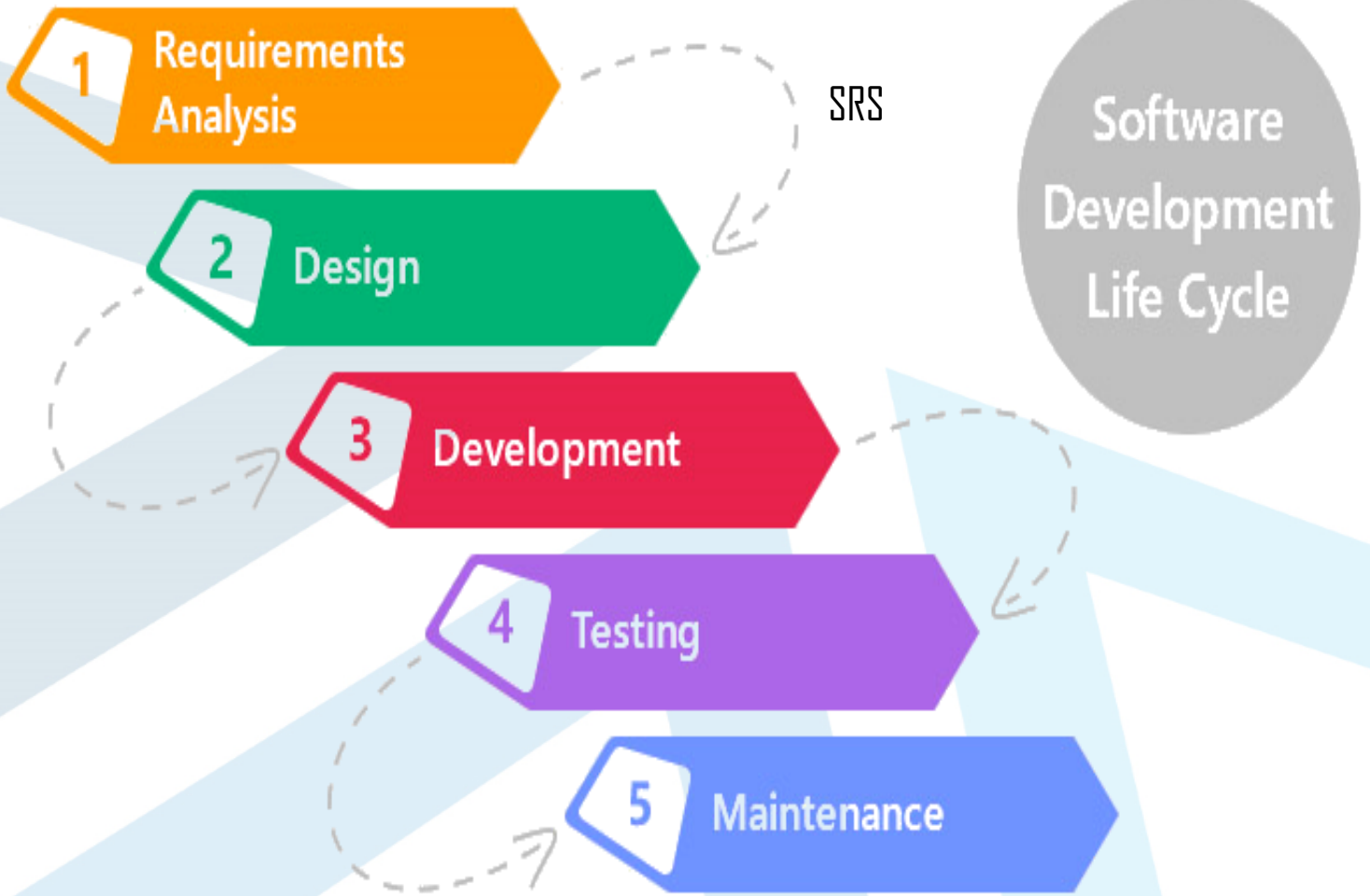
هندسة برمجيات ١

المحاضرة الرابعة

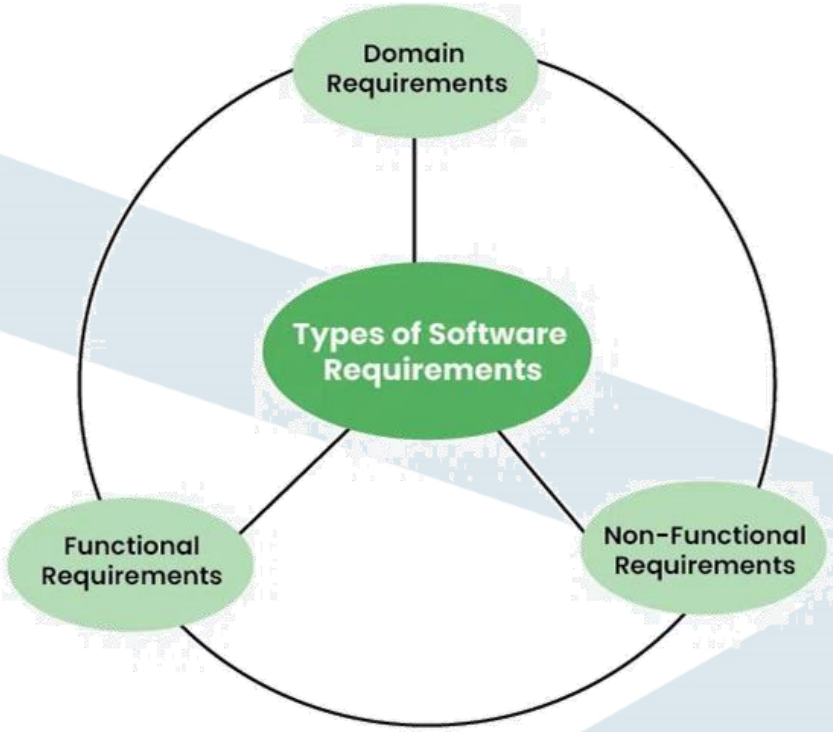
د. اناس ليلى

الفصل الدراسي الاول ٢٠٢٥ / ٢٠٢٦


1
Requirements Analysis
2
Design
3
Development
4
Testing
5
Maintenance
SRS
Software Development Life Cycle

يعتبر فهم وتحديد متطلبات المشروع البرمجي أساس نجاح هذا المشروع، ويمكن لأي سوء فهم لمتطلبات الزبون مهما كان بسيطاً أن يؤدي إلى الوصول لمنتج برمجي مختلف جذرياً عما يرغب به الزبون وبالتالي لا يلبي حاجته حتى وإن تم اختيار ال process model التي تضبط عملية الdevelopment المناسبة للمشروع ، وبناء على ذلك لابد من وجود طريقة لضبط عملية جمع و تحديد المتطلبات ، فظهر مايسمى ب"هندسة المتطلبات requirement engineering" وهي مجموعة من النشاطات والخطوات المنظمة لضبط عملية جمع وتوثيق المتطلبات بشكل صحيح خالٍ من التناقضات والأخطاء.

والمتطلبات هي توصيف لمجموعة الخدمات التي يقدمها النظام، والقيود التي لا ينبغي لهذا النظام أن تجاوزها.



جَـامعة المَـنارة
MANARA UNIVERSITY

Types of Software Requirements

Software Requirements are mainly classified into three types:

- Functional requirements
- Non-functional requirements
- Domain requirements

المتطلبات الوظيفية Functional Requirements :

تصف المتطلبات الوظيفية ما ينبغي أن يقوم به النظام البرمجي، تجمع هذه المتطلبات من أصحاب المصلحة ليقوم فريق العمل بتطوير البرنامج ويجب رؤية هذه المتطلبات مباشرةً في المنتج النهائي أمثلة:

مصادقة المستخدم: يجب أن يسمح النظام للمستخدمين بتسجيل الدخول باستخدام اسم مستخدم وكلمة مرور.

وظيفة البحث: يجب أن يُمكّن البرنامج المستخدمين من البحث عن المنتجات حسب الاسم أو الفئة.

إنشاء التقارير: يجب أن يكون النظام قادرًا على إنشاء تقارير مبيعات لفترة زمنية محددة.

مثال:

في تطبيق أتمتة عيادة طبية ، يمكن أن تكون المتطلبات الوظيفية :

- حجز موعد .
- اقتراح دواء مناسب.
- طباعة وصفة طبية .
- عرض قائمة المرضى الذين ينبغي معاينتهم كل يوم.
- تمكين المستخدم من البحث في قوائم مواعيدهم في جميع العيادات.

ويكون ال stakeholders المستفيدين من هذا النظام :

- المرضى الذين يملكون سجلات في النظام.
- الأطباء.
- الممرضين.
- موظفي الاستقبال المسؤولين عن تنظيم المعاينات.
- الفريق المطور للنظام ويتضمن المسؤولين عن صيانته.
- مدير أخلاقيات المهنة الذي يتأكد من أن النظام يراعي المبادئ الأخلاقية والسياسات التوجيهية لرعاية المرضى .

المتطلبات غير الوظيفية Non-functional Requirements

تصف المتطلبات غير الوظيفية كيفية أداء النظام البرمجي لمهمة ما وليس ما ينبغي أن يفعله، كما تُعرف بأنها قيود الجودة التي يجب أن يفي بها النظام وفقاً للعقد الذي تم توقيعه. ويتم التركيز في هذه المتطلبات على جوانب الأداء، والأمان، وقابلية التوسع، وسهولة الاستخدام وغيرها. أمثلة:

الأداء: يجب أن يعالج النظام ١٠٠٠ معاملة في الثانية.

سهولة الاستخدام: يجب أن يكون البرنامج سهل الاستخدام وذو واجهة مستخدم سهلة.

التوافرية: يجب أن يتمتع النظام بوقت تشغيل ٩٩,٩٪

الأمان: يجب تشفير البيانات أثناء النقل والتخزين.

اذاً تتعلق المتطلبات غير الوظيفية بجودة النظام. وهي تضمن أن يلبي البرنامج معايير معينة من الأداء وسهولة الاستخدام والموثوقية والأمان.

متطلبات النطاق Domain Requirements

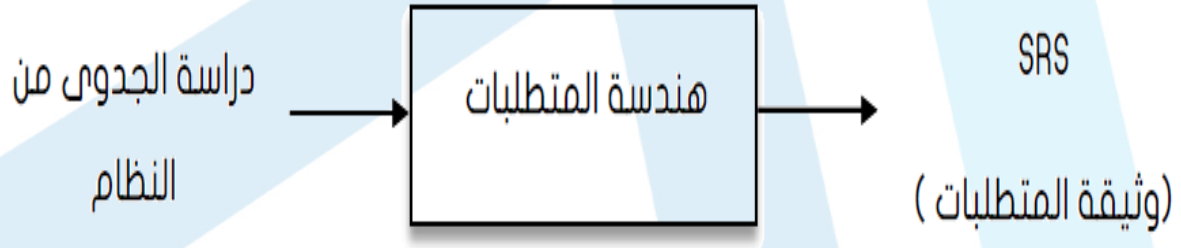
التعريف: متطلبات النطاق خاصة بالمجال أو القطاع الذي يعمل فيه البرنامج. وتشمل المصطلحات والقواعد والمعايير ذات الصلة بهذا القطاع. أمثلة:

الرعاية الصحية: يجب أن يتوافق البرنامج مع لوائح قانون نقل التأمين الصحي والمساءلة (HIPAA) لمعالجة بيانات المرضى.

الشؤون المالية: يجب أن يلتزم النظام بمعايير المحاسبة المقبولة عمومًا (GAAP) لإعداد التقارير المالية.

التجارة الإلكترونية: يجب أن يدعم البرنامج بوابات دفع متنوعة مثل PayPal وStripe وبطاقات الائتمان.

اذا تعكس متطلبات النطاق الاحتياجات والقيود الفريدة لكل قطاع. وهي تضمن أن يكون البرنامج ملائمًا ومتوافقًا مع اللوائح والمعايير الخاصة بكل قطاع.



و وثيقة ال SRS تلعب 3 أدوار و هي :

1- تكون بمثابة وثيقة لتوقيع العقد مع الزبون .

2- هذه الوثيقة تكون دخل لمرحلة التصميم و هي نقطة البداية التي يبدأ بها المصممون لتصميم النظام اعتماداً عليها .

3- أهم دور لها أنها مرجع أساسي في عمليات ال testing و قبول النظام أو بما نسميه (اختبار قبول النظام) أو acceptance testing .

طرق التدوين في الوثيقة :

1- الطريقة النصية : كتابة المتطلبات باللغة الطبيعية ، بحيث كل جملة تصف متطلب واحد وهي طريقة مفهومة للزبائن وللمستخدمين .

من مساؤها :

- من الصعب كتابة نص دقيق وسهل الفهم بنفس الوقت .
- يمكن الخلط بين المتطلبات الوظيفية وغير الوظيفية سواء في طريقة التعبير أو في فهم الزبون للنص.
- من أكبر مشاكل الطريقة النصية هي إمكانية التفسير المختلف للنص بين المطور والزبون حسب السياق.

2- الطريقة الرسومية graphical notation : وهي حل لمشكلة السياق النصي ، إذ يمكن لمخطط رسومي صغير مزود بتعليق نصي صغير التعبير عن محتوى نصي كبير ، ولأن معاني الرموز تختلف حسب الثقافات والبلدان تم الاتفاق على مجموعة من الرسوم والمخططات المعيارية فوضعت لغة النمذجة الموحدة UML التي تقترح مجموعة من الرموز المتعارف عليها .

نمذجة النظام System modeling

هي عملية تطوير نماذج مجردة لنظام ما، حيث يُقدم كل نموذج رؤية أو منظوراً مختلفاً لذلك النظام ويتم الاعتماد في أغلب الأحيان على لغة النمذجة الموحدة (UML) لنمذجة النظم البرمجية. تتطلب التطبيقات المعقدة التعاون والتخطيط من فرق متعددة وبالتالي تتطلب طريقة واضحة وموجزة للتواصل فيما بينهم ويتم تحقيق ذلك بواسطة UML

UML هي اختصارا لـ (Unified Modeling Language) اي لغة النمذجة الموحدة، وهي لغة نمذجة موحدة تتكون من مجموعة متكاملة من المخططات التي طُوّرت لمساعدة مطوري الأنظمة والبرمجيات على تحديد مكونات أنظمة البرمجيات وتصورها وإنشائها وتوثيقها.

تستخدم مخططات UML لعرض سلوك النظام وبنيته وهي ليست لغة برمجة بل هي لغة بصرية تساعد مهندسي البرمجيات، ومهندسي النظم في النمذجة والتصميم والتحليل.

ترتبط UML بالتحليل والتصميم **غرضي التوجه** object-oriented design and analysis وأول ما يلفت الانتباه فيها هو توفر العديد من المخططات (النماذج) المختلفة التي يُمكن استخدامها والتي تعرض النظام من وجهة نظر مختلفة

Types of UML Diagrams

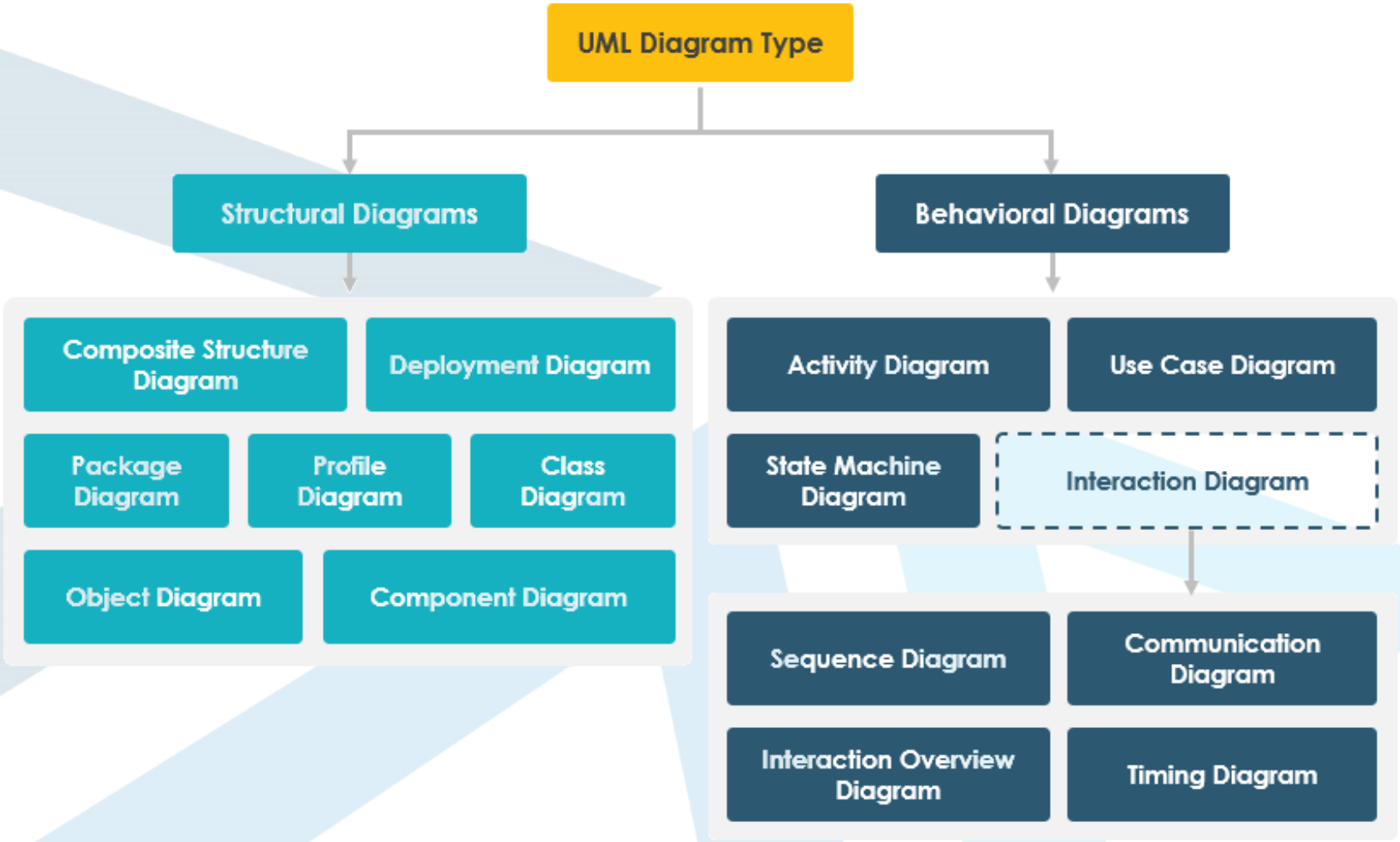
1- Structural UML Diagrams

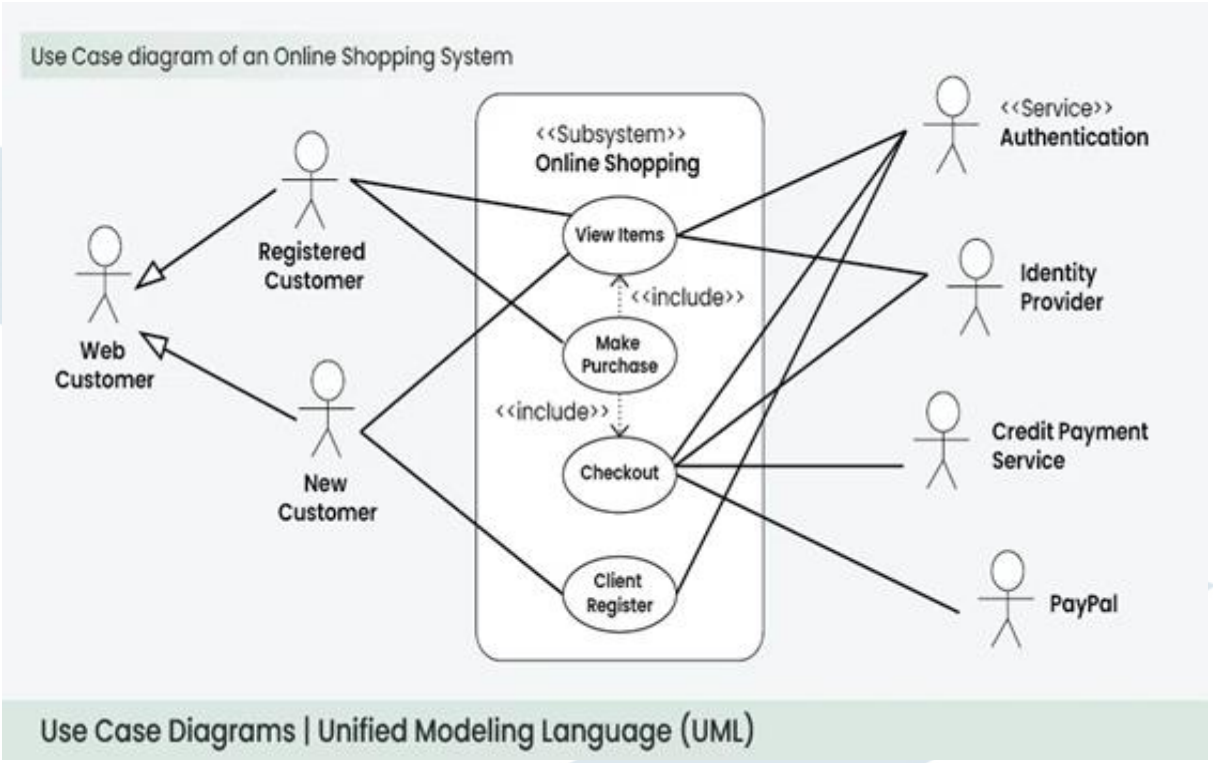
مخططات UML الهيكلية: هي تمثيلات بصرية توضح الجوانب الثابتة للنظام، بما في ذلك فئاته وكائناته ومكوناته وعلاقاتها(classes, objects, components, and their relationships)، مما يوفر رؤية واضحة لبنية النظام.

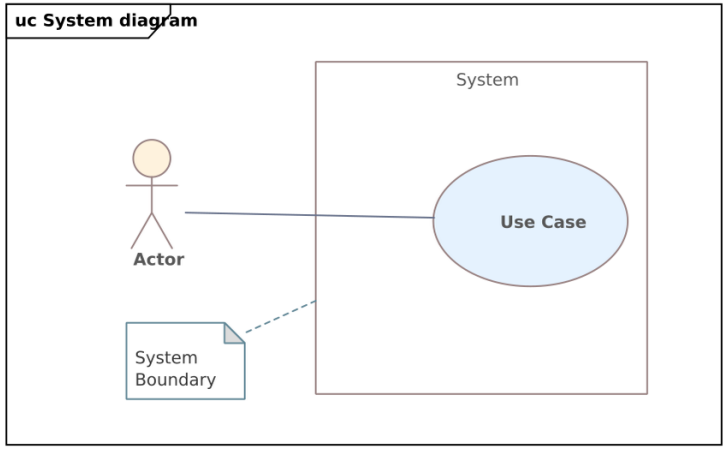
2- Behavioral UML Diagrams

مخططات UML السلوكية: هي تمثيلات بصرية تُظهر الجوانب الديناميكية للنظام، وتوضح كيفية تفاعل الكائنات وسلوكها مع مرور الوقت استجابةً للأحداث.

جَـامعة
المَـنارة
MANARA UNIVERSITY





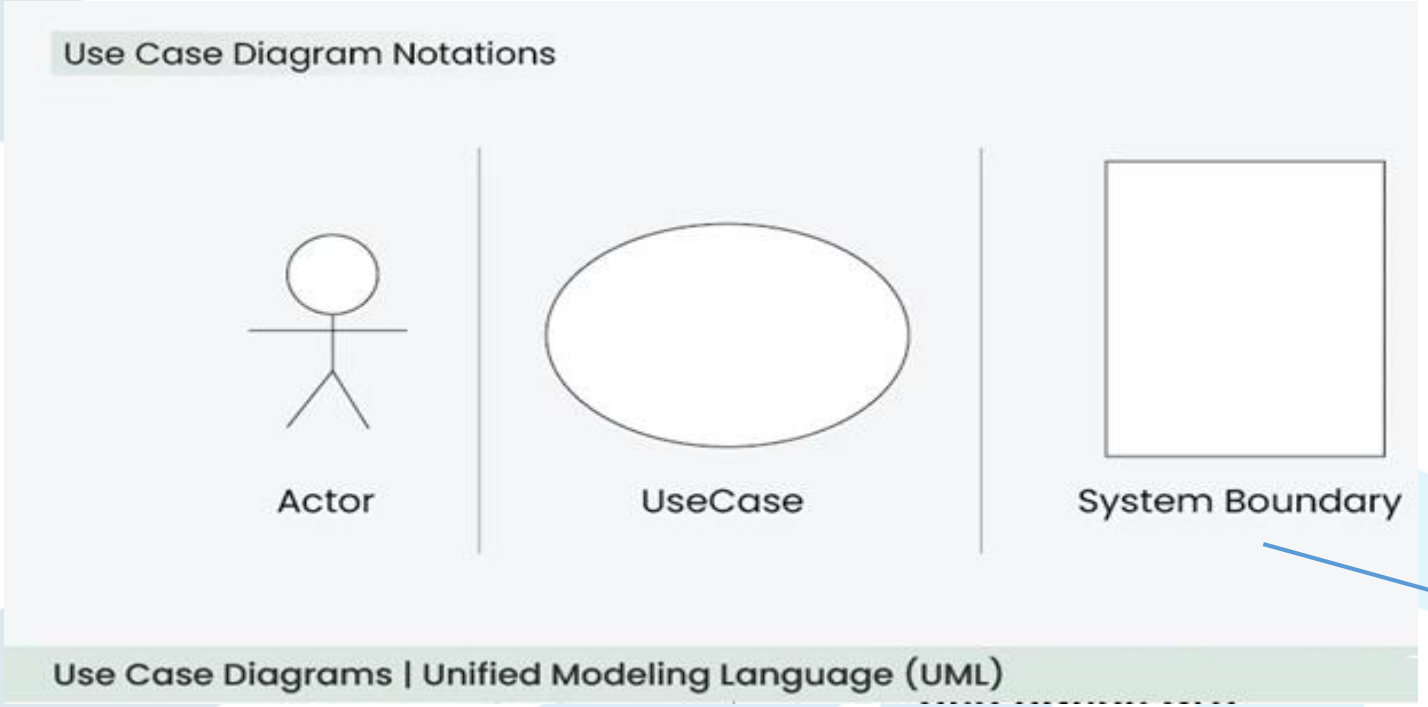


مخططات حالات الاستخدام Use Case Diagrams

تُستخدم مخططات حالات الاستخدام على نطاق واسع لتوضيح المتطلبات الوظيفية للنظام وتفاعله مع(actors).

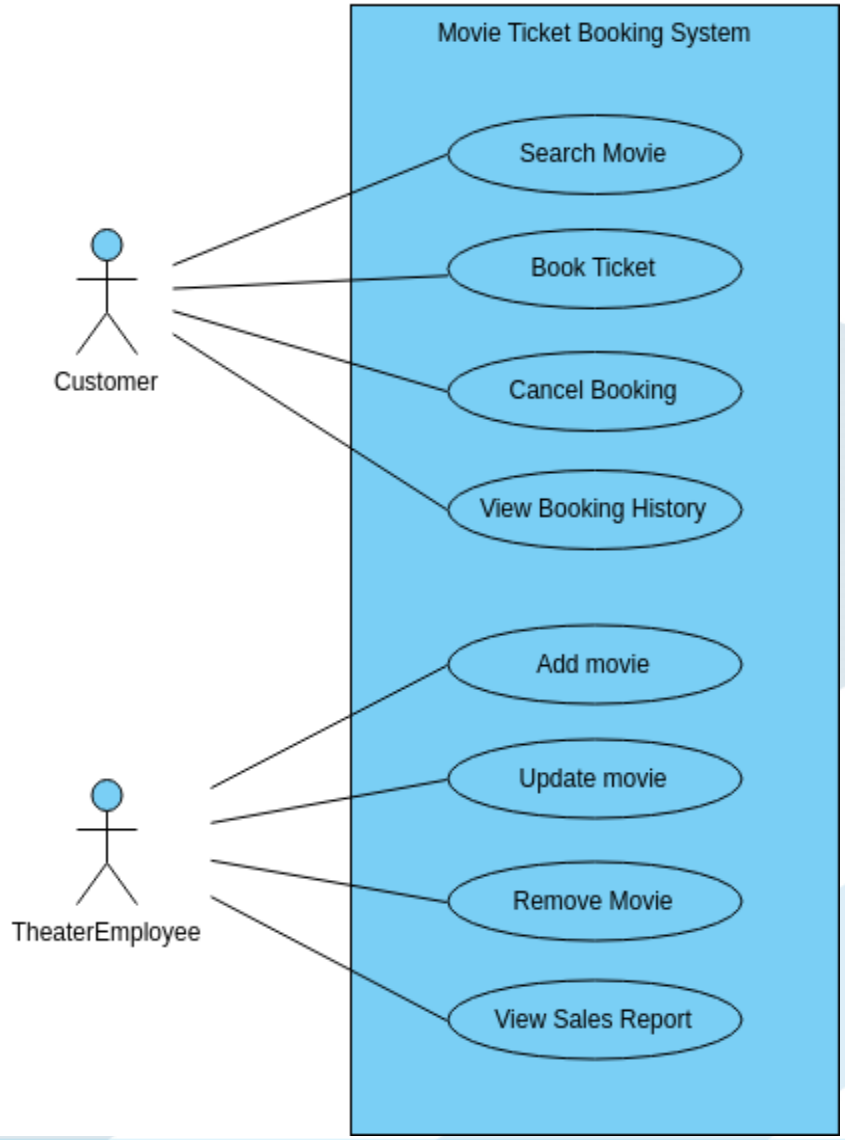
- مخطط حالة الاستخدام هو نوع من مخططات لغة النمذجة الموحدة UML، يُمثل التفاعل بين الجهات الفاعلة (المستخدمين أو الأنظمة الخارجية) مع النظام قيد الدراسة.
- يُعطي مخطط حالة الاستخدام نظرة شاملة على ما يفعله النظام أو جزء منه دون الخوض في تفاصيل التنفيذ.
- تُساعد مخططات حالة الاستخدام على تصوّر كيفية تفاعل مختلف المستخدمين مع النظام.

Use Case Diagram Notations



System boundary حدود النظام: حدود النظام هي تمثيل مرئي لحدود النظام الذي يُنمذج. تُساعد الحدود على التمييز بوضوح بين العناصر التي تُشكل جزءاً من النظام وتلك التي تقع خارجه. عادةً ما تُمثل حدود النظام بمستطيل يُحيط يضم جميع حالات الاستخدام فيه. ولا يمكن للنظام أن يمتلك وظائف لا نهائية.

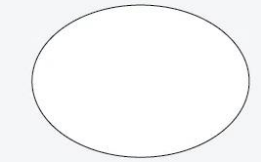
حالات الاستخدام Use Cases



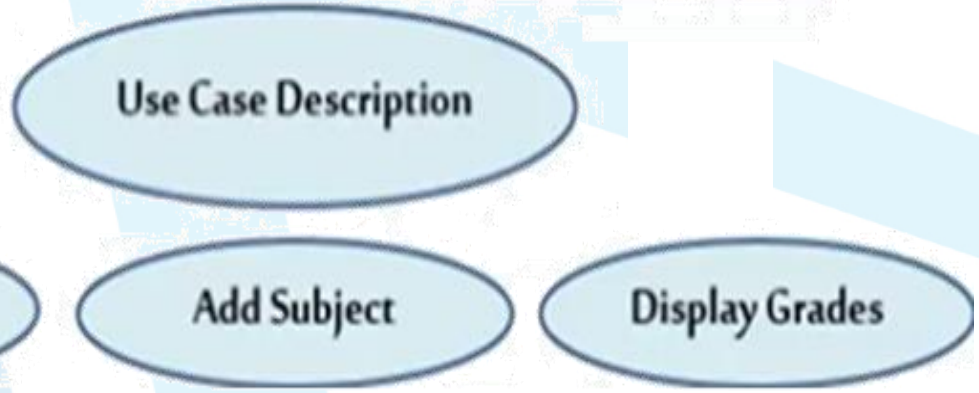
وهي العمليات داخل النظام او المهام التي يقوم بها النظام

يجب ان يكون وصفها بفعل لأنها عملية (وظيفه)

تمثل بشكل بيضاوي يوضع داخلها وصف وظيفة حالة الاستخدام



Use cases are represented by ovals.

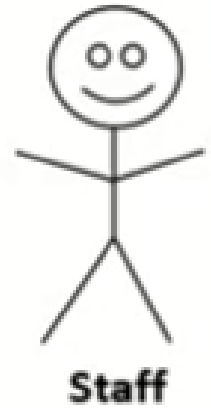


Actors

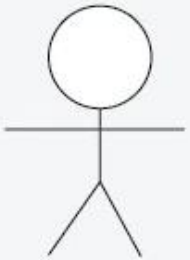
Actors are external entities that interact with the system.
These can include users, other systems, or hardware devices.
In the context of a Use Case Diagram, actors initiate use cases and receive the outcomes.

- *Actor* is someone interacting with use case (system function). Named by noun.
- Actor *triggers* use case.
- Actor has responsibility toward the system (inputs), and Actor have expectations from the system (outputs).

Actors may be connected to use cases by associations, indicating that the actor and the use case communicate with one another using messages.



Actor



Actors are typically represented by stick figures

Actor – anything that needs to interact with the system to exchange information.

- ***Could be*** a **human**, an **organization**, **another information system**, an **external device**, or even **time**.

Ex. **College student enrolling for the fall semester's course**

Actor: student

Use case : enrolling in course

(business event) … ???

The trigger for the use case – the **event** that causes the use case to **begin**.

➢Temporal Event

a system event triggered by time.

- ❑ The actor is time.
- ❑ The billing systems for a credit card company **automatically generates** its bills on the 5th day of the month

  - **actor: Billing Date**

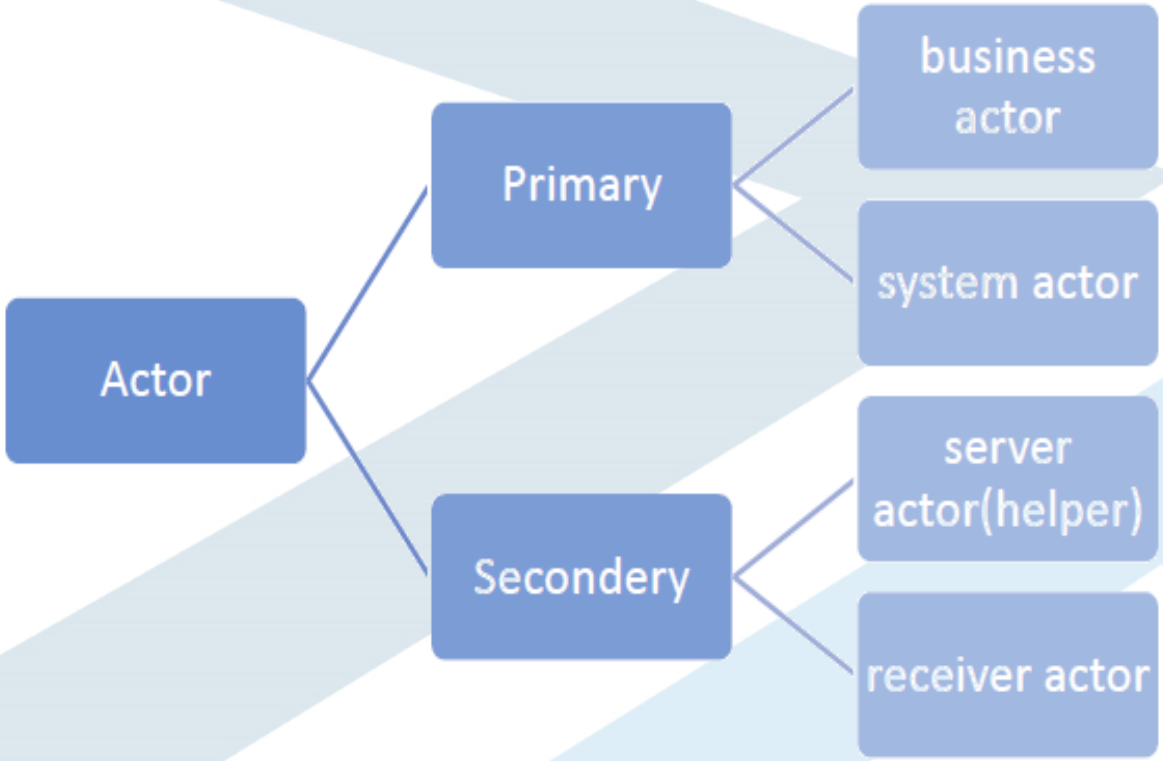
في **مخطط حالة الاستخدام، تُمثل الجهات الفاعلة كيانات خارجية تتفاعل مع النظام قيد النمذجة. تُصنف الأنواع الرئيسية للجهات الفاعلة بناءً على دورها وتفاعلها مع النظام.**

الجهات الفاعلة الأساسية، والمعروفة أيضًا بالجهات الفاعلة النشطة، هي كيانات خارجية تُطلق حالة استخدام لتحقيق هدف محدد. وهي عادةً المستخدمون الرئيسيون أو الأنظمة الخارجية التي تستفيد مباشرةً من وظائف النظام. ومن الأمثلة على ذلك عميل يُجري طلبًا عبر نظام تجارة إلكترونية، أو مستخدم يُحوّل أموالًا عبر تطبيق مصرفي، أو طالب يُسجل في دورة تدريبية عبر بوابة جامعية.

الجهات الفاعلة الثانوية، والمعروفة أيضًا بالجهات الفاعلة السلبية أو الجهات الفاعلة الداعمة، هي كيانات تُساعد النظام في إكمال حالة استخدام ولكنها لا تُطلقها بنفسها.

تُوفر هذه الجهات الخدمات أو الموارد اللازمة لعمل النظام، مثل بوابة دفع تتحقق من المعاملات، أو خادم بريد إلكتروني يُرسل رسائل تأكيد، أو قاعدة بيانات تُخزن البيانات وتسترجعها. يستخدم النظام الجهات الفاعلة الثانوية ولا يتفاعل معها بشكل مُستقل.

يصنف ال Actors على النحو التالي :



سيناريو -1-

اقرأ السيناريو التالي :

في مركز سيرياتيل يطلب الزبون من الموظف المسؤول عن الدور -بطاقة دور- له حتى يتمكن من دفع الفاتورة , يقوم الموظف عن طريق النظام الآلي بإخراج بطاقة الدور و إعطاءها للزبون , عندما يحين دور الزبون , يتواصل مع موظف آخر من أجل دفع الفاتورة وهذا الموظف يتعامل مع النظام الآلي استجابةً لطلب الزبون .

يوجد في هذا السيناريو منظومتين برمجيتين اثنتين :

- منظومة الدور.

- منظومة خدمات الشركة.

المنظومة الأولى تعامل معها كلاً من الموظف و الزبون .

المنظومة الثانية تعامل معها الموظف والزبون ومنظومة الدور (حيث لا يصل الزبون لمنظومة الخدمات حتى يقطع الدور) .

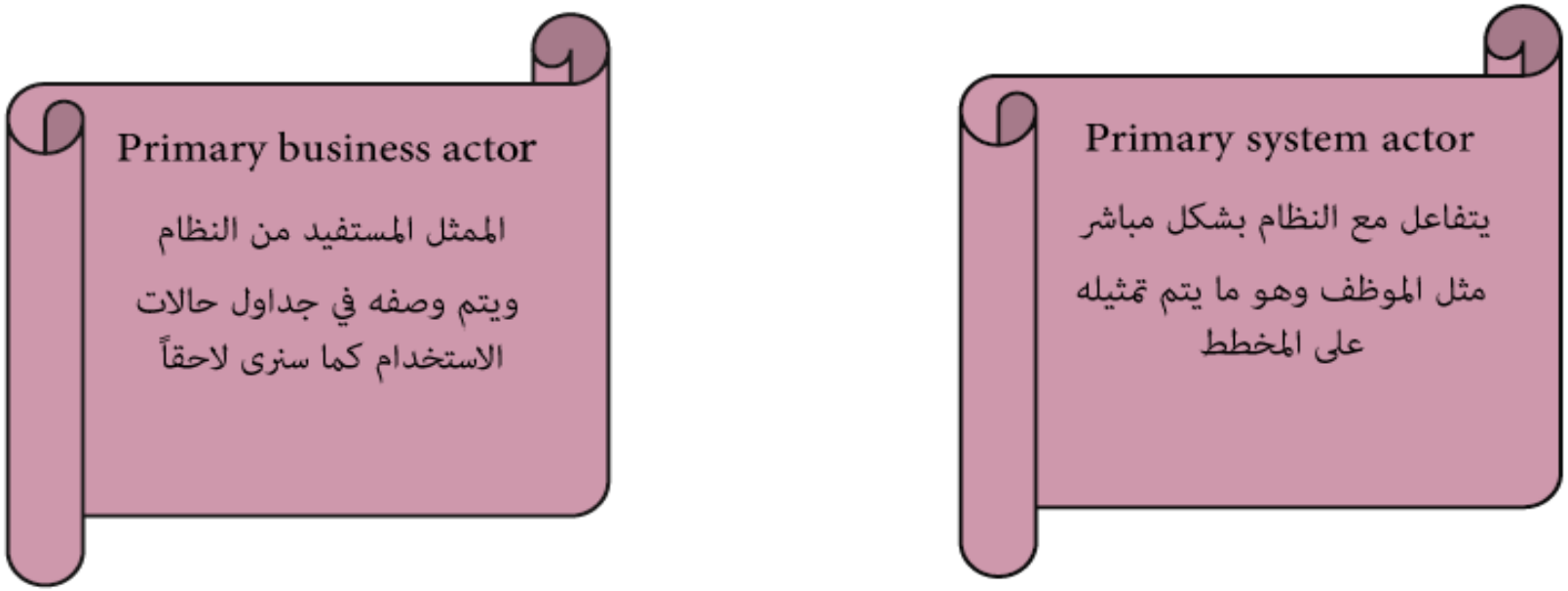
في منظومة الدور تفاعل الموظف بشكل مباشر مع النظام , في منظومة الخدمات تفاعل الموظف ومنظومة الدور بشكل مباشر مع النظام , وكان الزبون هو المستفيد من خدمات كلا المنظومتين والذي قام بتفعيل الموظف كل مرة ليتعامل مع النظام .

إذاً من هو ال actor في هذا النظام ؟

لاحظ أن الزبون لم يتعامل مباشرةً مع النظام الآلي , وبالتالي لا يعتبر ال actor في المخطط, كما أن الموظف لا يعتبر جزء من النظام حتى وإن كان عاملاً في الشركة , وبالتالي يقع خارج حدود النظام الآلي وكونه من قام بالتعامل المباشر مع النظام فيكون هو ال actor على المخطط.

في هذا التحليل قمنا بإلغاء الزبون رغم أنه المفعّل الحقيقي للوظائف , ولا يمكن وضعه على المخطط لأنه لا يوجد لدينا علاقة actor to actor في ال use case diagram فمالحل؟

*يقسم ال primary actor (الذي يستفيد أو يتفاعل مع النظام مباشرةً) إلى :



بعض الأنظمة يكون فيها ال Primary system actor هو نفسه ال Primary business actor كما في نظام ال ATM.

سيناريو -2-

يقوم الزبون بفتح حساب على منتدى معين ليتمكن من الاستفادة من خدماته , فتظهر رسالة للزبون بإرسال طلبه إلى admin المنتدى , ويُرسل طلب الانتساب إلى ال admin كإشعار وهو بدوره يوافق على الطلب أو يرفضه .

إذا اعتبرنا المنتدى منظومة برمجية , يكون الزبون هو ال System actor وال business actor لأنه تواصل مع المنتدى ولأنه سيستفيد من الخدمة بنفس الوقت ,أما الadmin فإنه تفاعل مع النظام لرفض الطلب أو قبوله ,

ولكنه ليس مستفيد من خدمات النظام وإنما مجرد صانع قرار تابع لطلب الزبون , ووجوده مرهون بوجود الزبون أي لامعنى لتفاعله مع النظام دون أن يرسل الزبون طلب انضمام فنسميه secondary actor ولأنه استجاب لطلب الزبون وأرسل رفضه أو موافقته للنظام(استقبل data و أرسل data) فإنه يصنف ك server actor or helper.

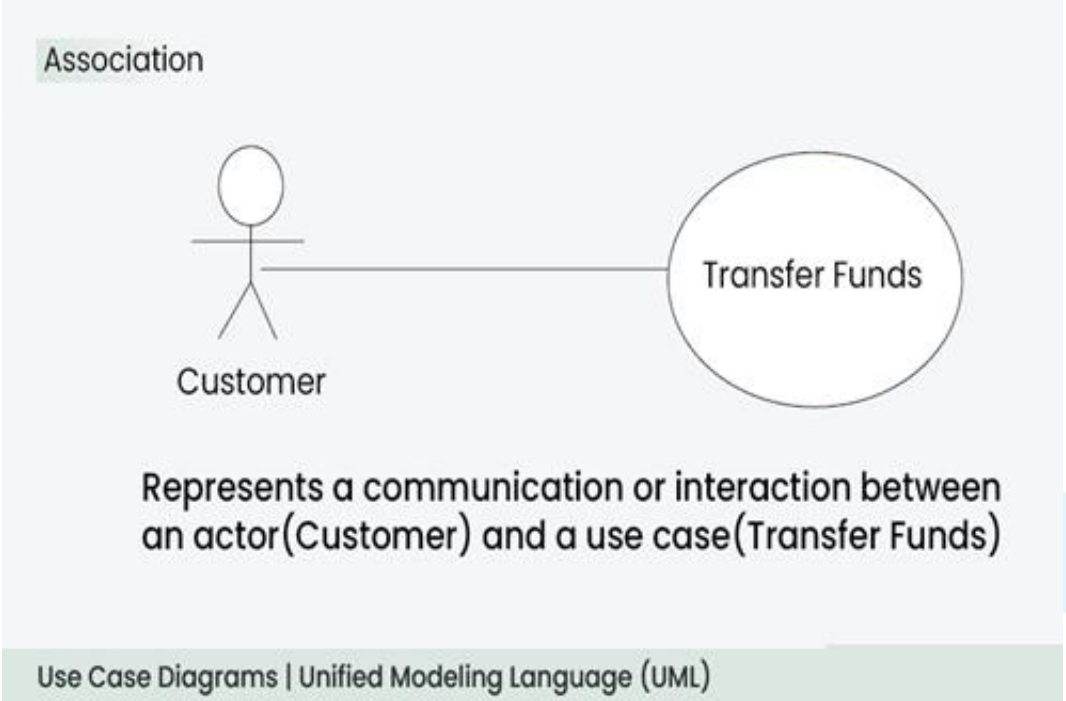
Use Case Diagram Relationships

1. Association Relationship

تمثل علاقة الارتباط تواصلًا أو تفاعلًا بين فاعل وحالة استخدام. ويُمثَّل بخط يربط الفاعل بحالة الاستخدام.
تدل هذه العلاقة على أن الفاعل مُشارك في الوظيفة الموصوفة في حالة الاستخدام.

Example: Online Banking System

- **Actor:** Customer
- **Use Case:** Transfer Funds
- **Association:** A line connecting the "Customer" actor to the "Transfer Funds" use case, indicating the customer's involvement in the funds transfer process.



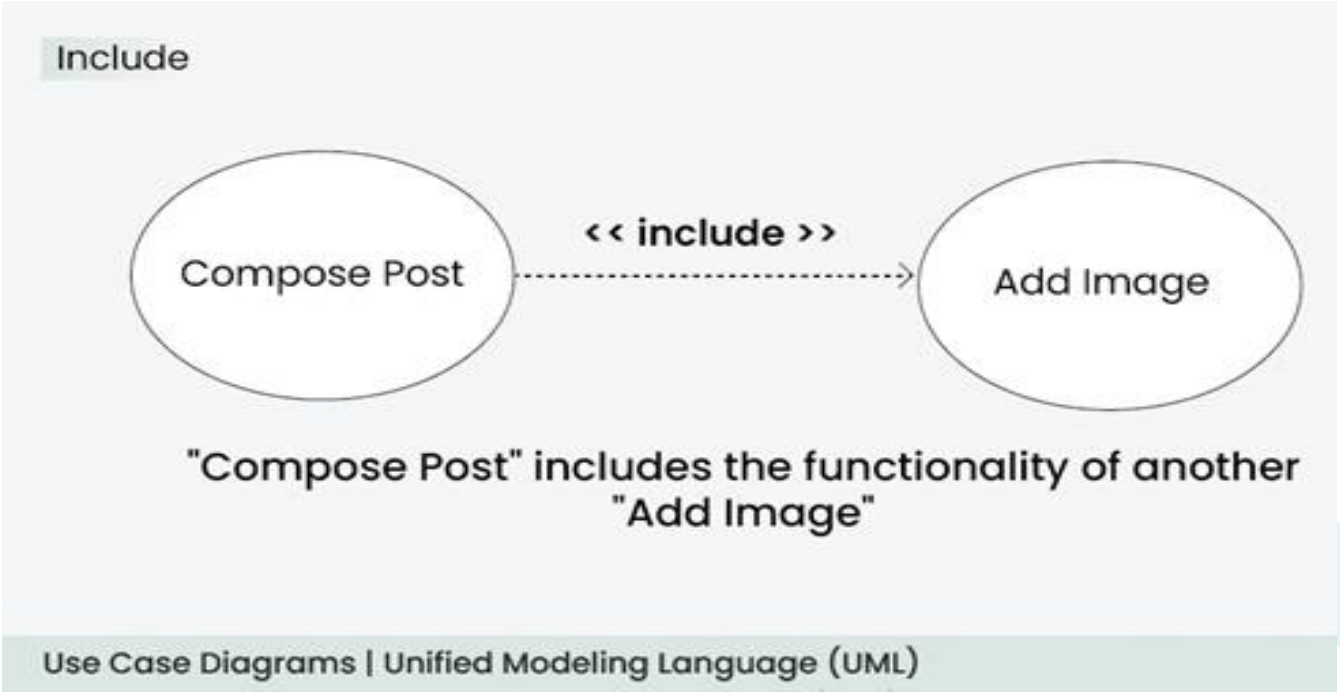
2- Include Relationship

a use case includes the functionality described in another use case as a part of its business process flow. A key here is that the included use case cannot stand alone

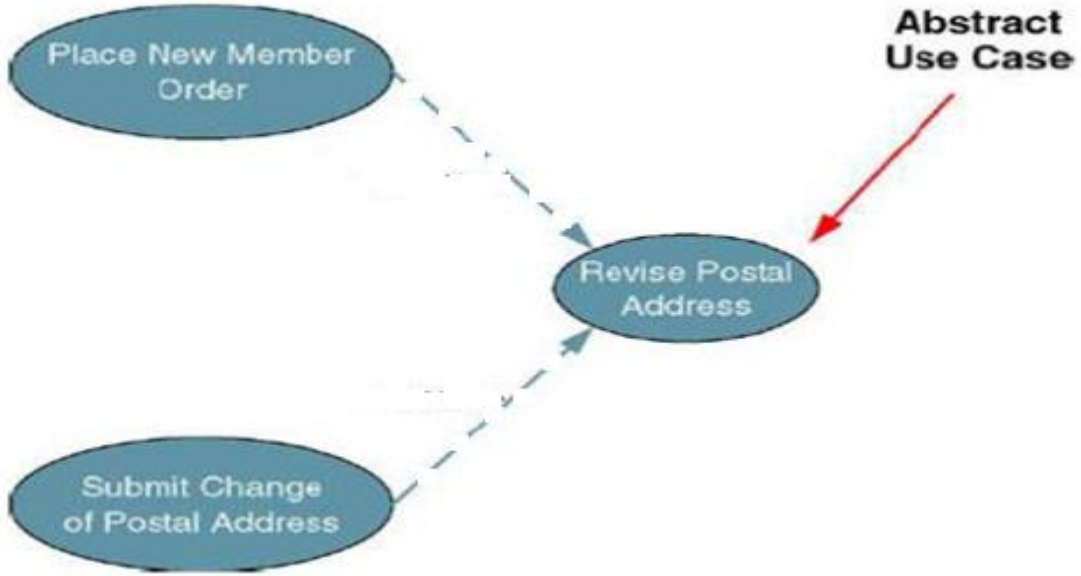
The Include Relationship indicates that a use case includes the functionality of another use case.

It is denoted by a dashed arrow pointing from the including use case to the included use case.

This relationship promotes modular and reusable design



- الخطوة 3 و 4 مكررة عند A,B لذلك نضع use case افتراضية نسميها C خطواتها 3,4 بحيث :



عندما يلزم أكثر من use case نفس المجموعة من الخطوات الجزئية للحصول على الخدمة نقوم باشتقاق use case افتراضية تجريدية خطواتها هي مجموعة الخطوات المكررة وترتبط مع ال use case التي تحتاجها للحصول على الخدمة بعلاقة <<include>> or <<uses>> , وهي علاقة إجبارية بمعنى أنه لا يمكن إنجاز الخدمة الأساسية دون إنجاز خطوات هذه ال use case الافتراضية , وهذه ال use case غالباً تكون غير موجودة ضمن متطلبات النظام بالأصل (ليست خدمة) والهدف منها تقليل التكرار .

اتجاه السهم المعبر عن علاقة include من الحالة الأساسية إلى المشتقّة , مثال :

لدينا الحالتين A , B بالخطوات التالية :

A

1
2
3
4

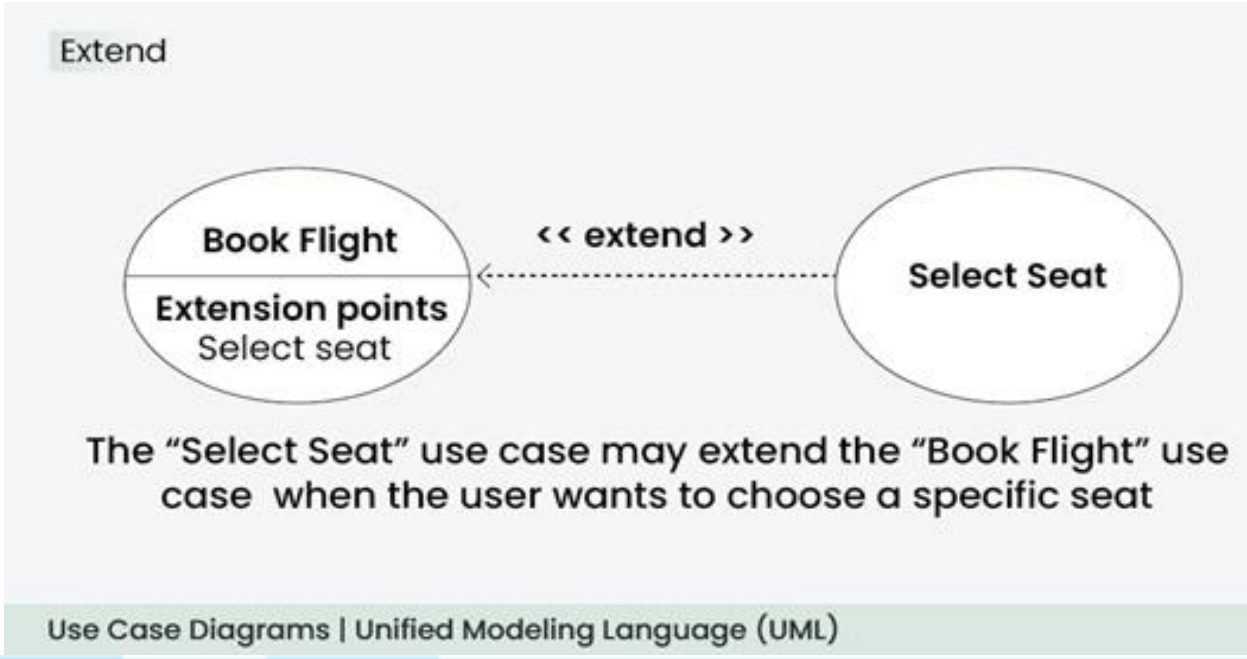
B

a
b
3
4

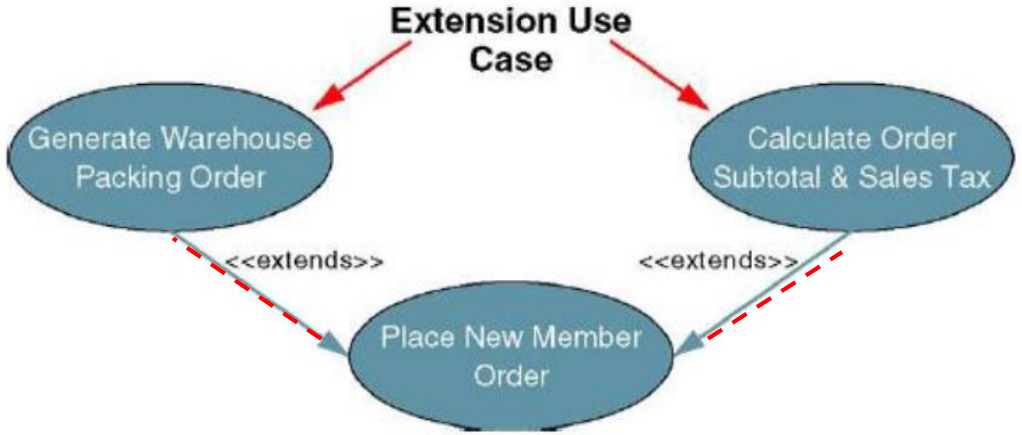
3. Extend Relationship

The Extend Relationship illustrates that a use case can be extended by another use case under specific conditions. It is represented by a dashed arrow with the keyword "extend." This relationship is useful for handling optional or exceptional behavior.

- حالة استخدام توسع عمل حالة استخدام أخرى , تستخدم في الحالات التالية
- كجزء اختياري من سلوك حالة ما , لا وجود لحالة الاستخدام الموسعة دون وجود حالة الاستخدام الأساس و لكن حالة الاستخدام الأساس ممكن أن يوجد دون حالة الاستخدام الموسعة
- كحالة يمكن تنفيذها فقط في ظروف معينة (الانتقال من حالة إلى أخرى وفق شرط معين)
- يمكن لحالة الاستخدام الأساس أن يكون لديها عدة حالات استخدام موسعة
- يمكن أن نقوم بتوسعة حالة الاستخدام الموسعة



- نقوم بتوسعة A ب extention use case نسميها C خطواتها هي 5,6 , ترتبط معها بعلاقة <<extend>> واتجاه السهم فيها من المشتق إلى الأساسي أي من C إلى A ويمكن للزبون أن يختار تنفيذ C أو لا بعد الحصول على A ونقول إن



وهي علاقة تربط بين خدمة أساسية وخدمة أخرى تشكّل توسعة لها بحيث يكون تنفيذها أو عدم تنفيذها لايؤثر على الحصول على الخدمة الأساسية , كطلب الزبون لفاتورة تفصيلية بعد حصوله على الفاتورة الأساسية .

والخدمة المشتقة ليست خدمة أساسية قائمة بحد ذاتها , ولا تتحقق دون تحقق الخدمة الأساسية المرتبطة بها و دون طلب الزبون لها .

مثال :

لدينا ال use case المسماة A والتي يلزمها الخطوات التالية لتنفيذها :

1

2

3

هنا تم الحصول على الخدمة الأساسية 4

إضافة على الخدمة 5

إضافة على الخدمة 6

جَـامعة
المَـنارة
MANARA UNIVERSITY