

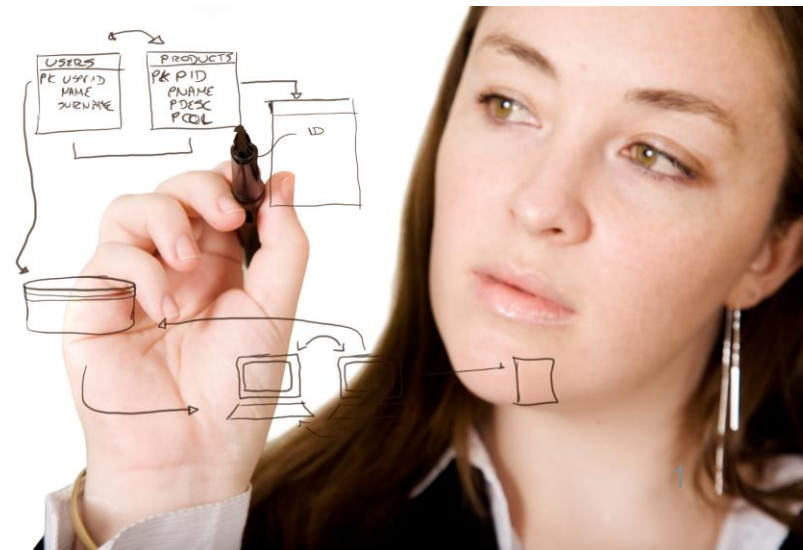
מודלים לפיתוח מערכות תוכנה

Software Systems Modeling

קורס 12003
סמסטר ב' תשע"ו

UML 3.1

ד"ר ראובן יגל
robi@post.jce.ac.il



השבוע

- UML
 - ריענון
 - תרחישי שימוש
- סיום תרגיל 1
- תרגיל 2 מידול בסיסי – חלק א'

מקורות להיום

- Modeling
 - [RIT Class](#), Wei Le
- UML
 - UML Specification <http://www.omg.org/spec/UML/>
 - Fowler, "UML Distilled: A Brief Guide to the Standard Object Modeling Language"
 - Ambler, [Introduction to Object-Orientation and the UML](#)
 - Cockburn [Writing Effective Use Cases](#)
 - se-class requirement lecture <http://jce-il.github.io/se-class/lecture/se03-requirements.pdf>

Remindaer: Challenges

Create Software Models

- Modeling languages:
https://en.wikipedia.org/wiki/Modeling_language
 - General purpose and domain-specific languages
 - Formalism
 - Level of abstraction
- Models for software running in different platforms
 - Model-driven architecture
 - Views: PIM (computation), CIM (environment), PSM
- Models for software consistently changing at runtime (agent)
- Modularity, separate concerns

Manage Software Models

- Find information from the models (query)
- Correctness of the models:
 - Model consistencies
 - Model checking models
- Transformations
 - Decomposition
 - Composition
 - Between models
- Evolutions of models

Use Software Models

- Generate code
- Monitor runtime software behavior
(interacting with environments, adaptation)
- Testing (model-based testing criteria and test input generation)

UML Modeling - Overview

UML Modeling

- A language: syntax and semantics
- Capture ideas, relations, decisions, requirements in a well-defined notations

AgileData.org: ... all developers should have a basic understanding of the industry-standard [Unified Modeling Language \(UML\)](#). A good starting point is to understand what I consider to be the [core UML diagrams](#) – [use case diagrams](#), [sequence diagrams](#), and [class diagrams](#) – although as I argued in [An Introduction to Agile Modeling and Agile Documentation](#) you must be willing to learn more models over time.

UML Diagrams

- Structural : relations of objects (class diagram, component diagram)
- Behavioral : sequence of actions (activity diagram, sequence diagram)

UML Views

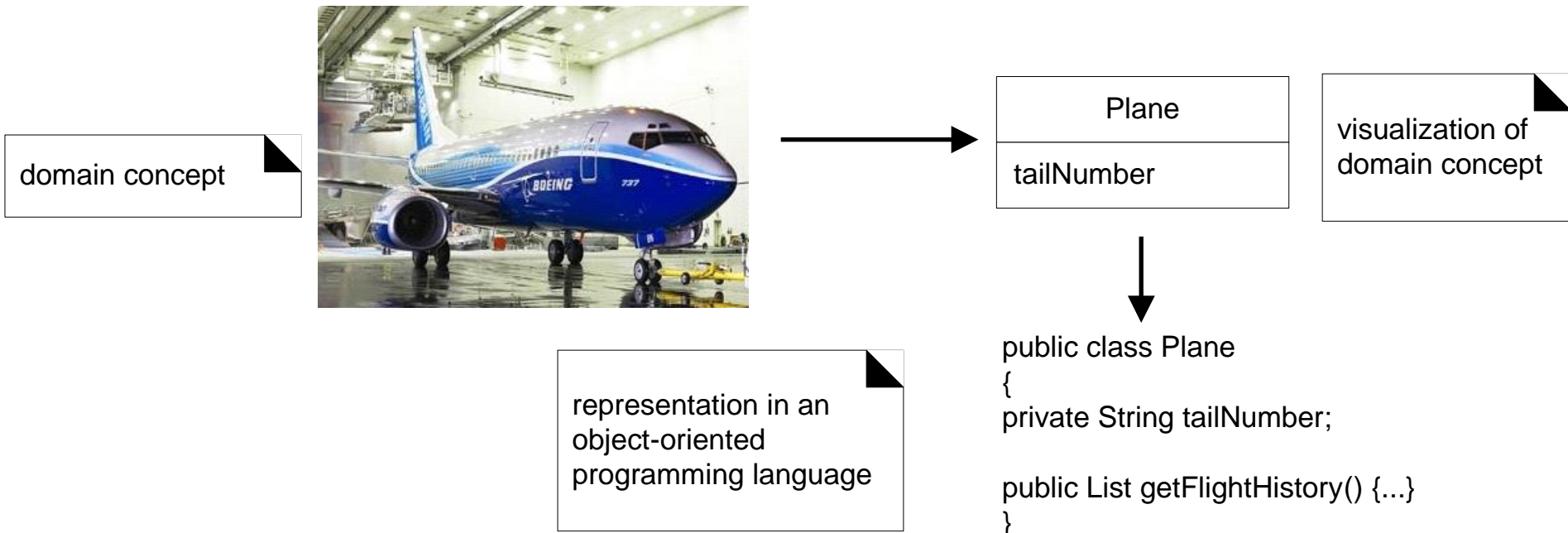
One concept can be expressed in different graphs, choose one to express for your purposes, for your audiences

- Design – class, structure
- Deployment – configure, install,
- Implementation – state chart, interaction
- Process – performance, runtime behavior
- Use case

Domain concept

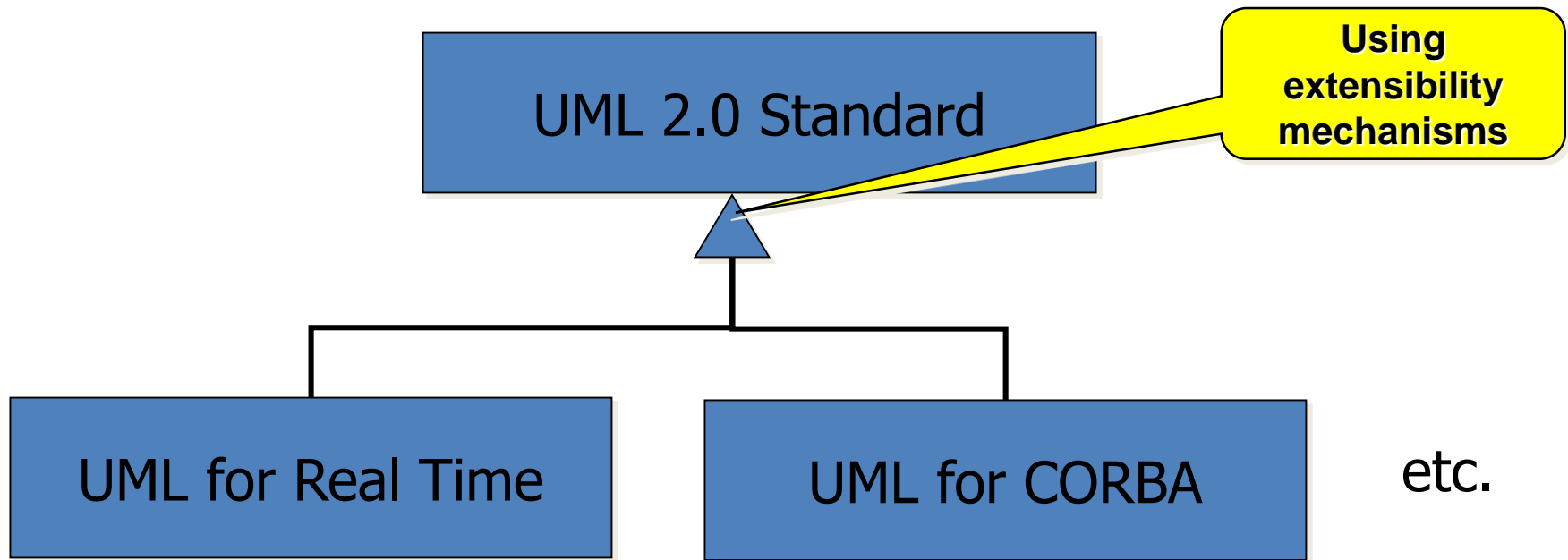
Design representation

Code representation



UML as a “Family of Languages”

- The standard can be specialized for different domains



Use case and use case diagram

Steps Before Coding

Phase	Action	Results
Initiation	Raise a business need	domain model, business use cases
Requirement	What to accomplish (abstract)	use case, activity diagrams
Design	How the system works (more details: software architecture, components, data types, algorithms)	component, class, sequence diagrams, formal specifications

Source of Requirements

- Initial requirements come from the customer, by:
 1. Documents, such as RFI/RFP
 2. Meetings, reports
- Advanced requirements come from the analysts, after studying scope and price
 1. Feasibility (technological, organizational etc)
 2. Prototypes
- Final requirements are stabilized in an iterative process.

Types of Requirements

Visible Functional Requirements

“The system will deliver cash to the customer”

“Cash will be delivered after card was taken out”

Qualitative Requirements

“The authorization process will take no more than 1 sec”

“The user interface will be easy to use”

Hidden Requirements

“Database maintenance processes will occur every night”

Intro: Use Case and Use Case Diagram

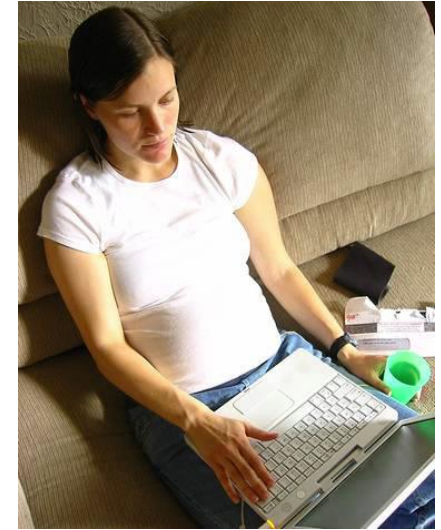
Use Cases as Means of Communication



Customer



Designer



User

The use case should stimulate a discussion about **what the system should do, mainly with people who are outside of the development team.**

Use Case

A use case is a **contract of an interaction between the system and an actor**.

Use Case Diagram: an integration of use cases

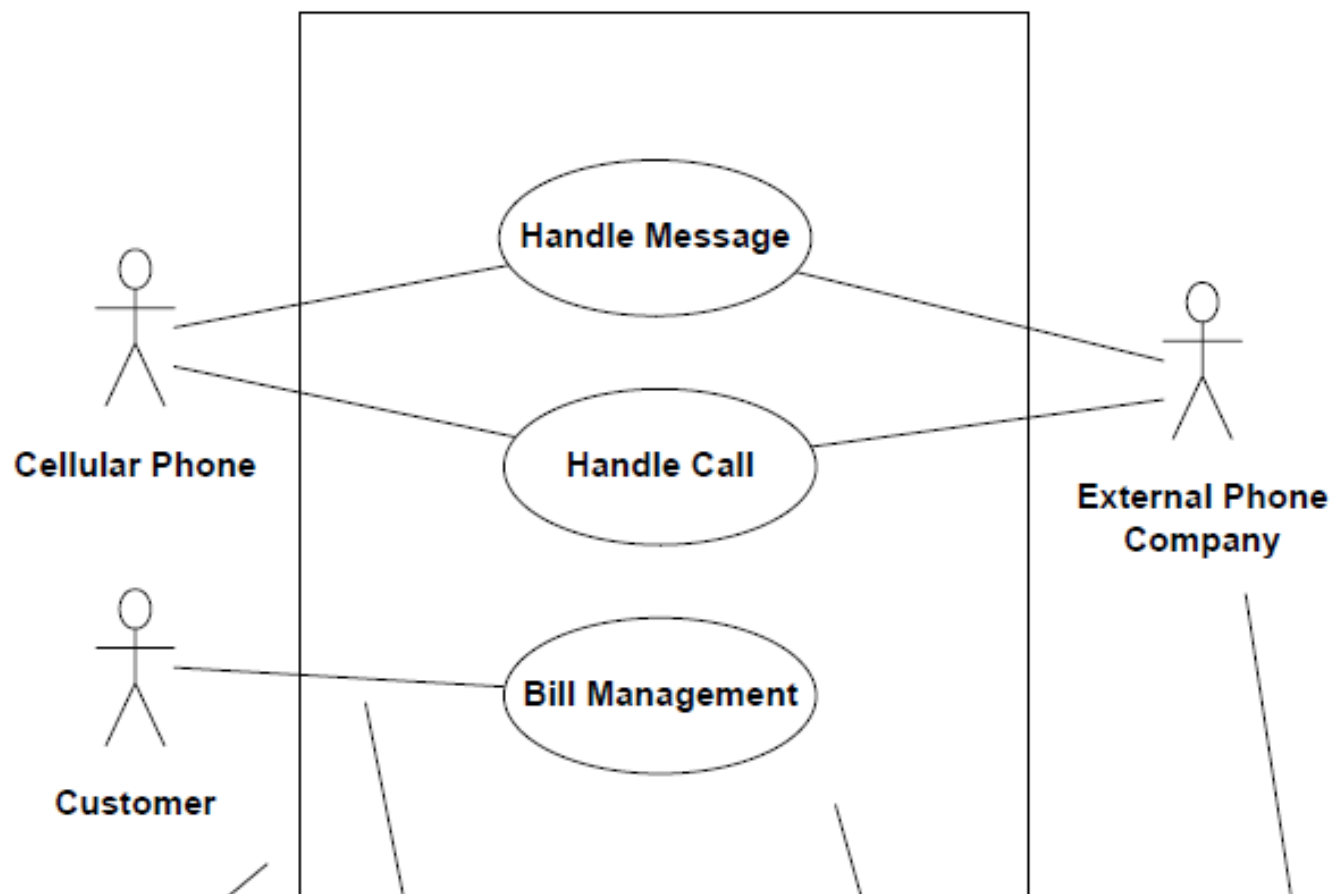
Use Case Diagram

A use case diagram illustrates a set of use cases for a system, the actors, and the interactions between actors and use cases.

A graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

Use Case Diagram Objectives

1. Create a semi-formal model of the functional requirements
2. Analyze and define:
 - Scope
 - External interfaces
 - Scenarios and reactions



System
boundary

Association

Use Case

Actors

Example

What makes a good Use Case Diagram?

Lack of ambiguity

- Each requirement must be interpreted in a single manner.

Completeness

- The collection of all use cases is everything that can be done to/with the system.

Consistency

- Requirements should not conflict with each other. If there are, tradeoffs must be detected and discussed.

Avoid design

- Requirements should raise a need, not answer it.

Construct a Use Case Diagram

Finding actors

External objects that produce/consume data:

1. Must serve as sources and destinations for data
2. Must be external to the system
3. **Actors vs. Stakeholders**

Humans

Machines

External systems

Sensors

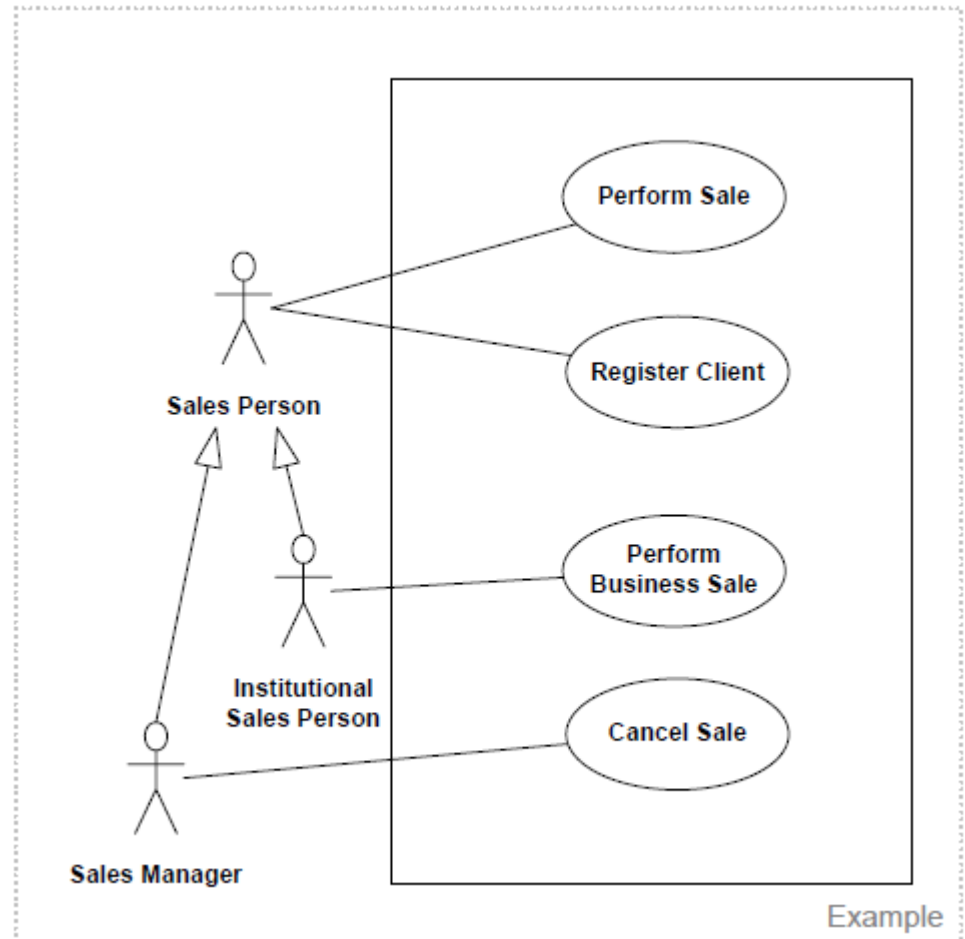
Actor Relationships – Generalization/Specialization

Define hierarchy for actors

Notation

The child actor inherits all use-cases associations

Should be used if (and only if), the specific actor has more responsibility than the generalized one (i.e., associate with more use-cases)



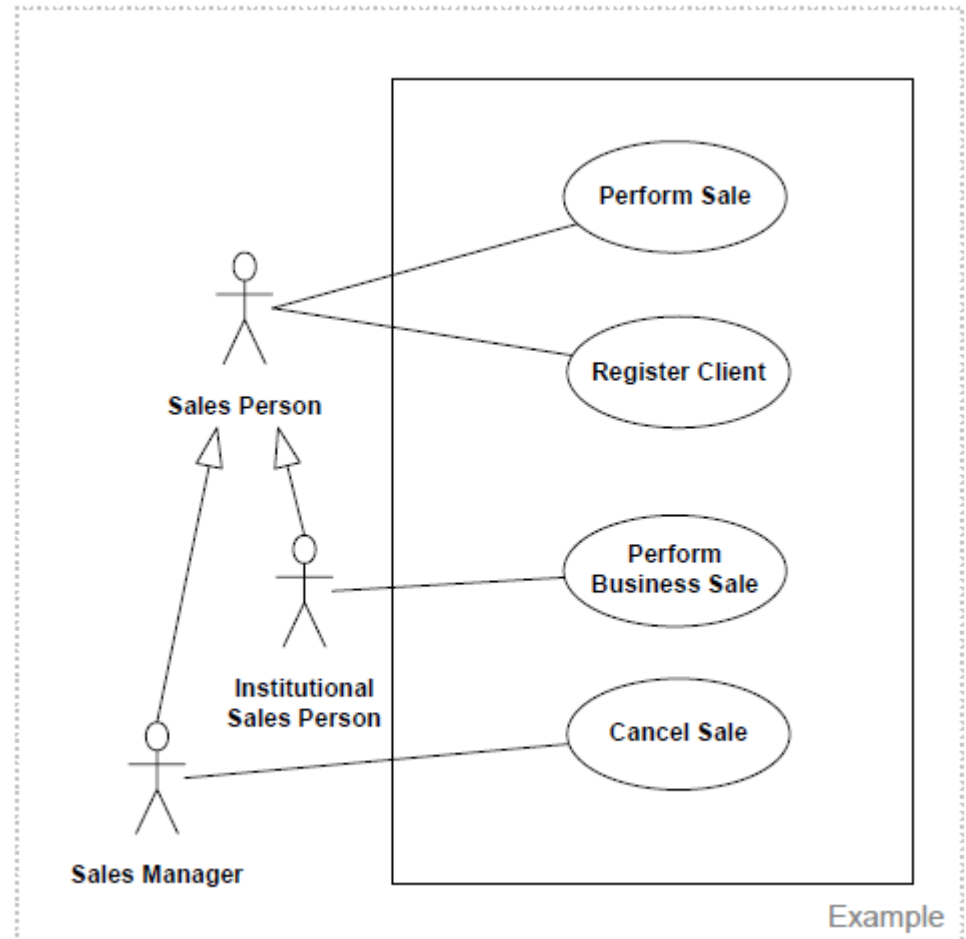
Association: Actor and Use Case

Solid line:

Interaction between actors
and use case

Arrowhead (optional)

- Control flow
- Initial invocation, primary actor

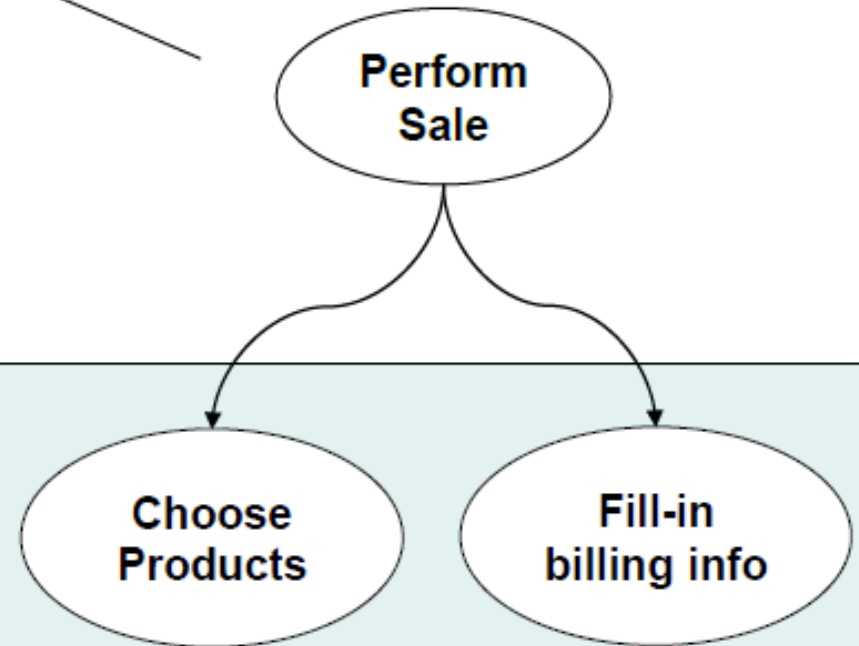
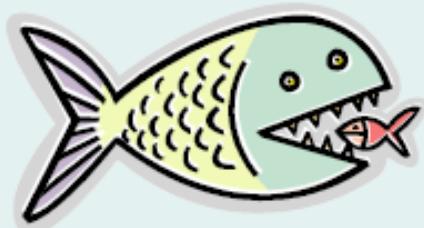


Use Case Levels

Base Use Case:
Used directly by
the user

↑ User goals

↓ Sub-functionality



Alistair Cockburn "Writing Effective Use Cases"

Use Case Relationships

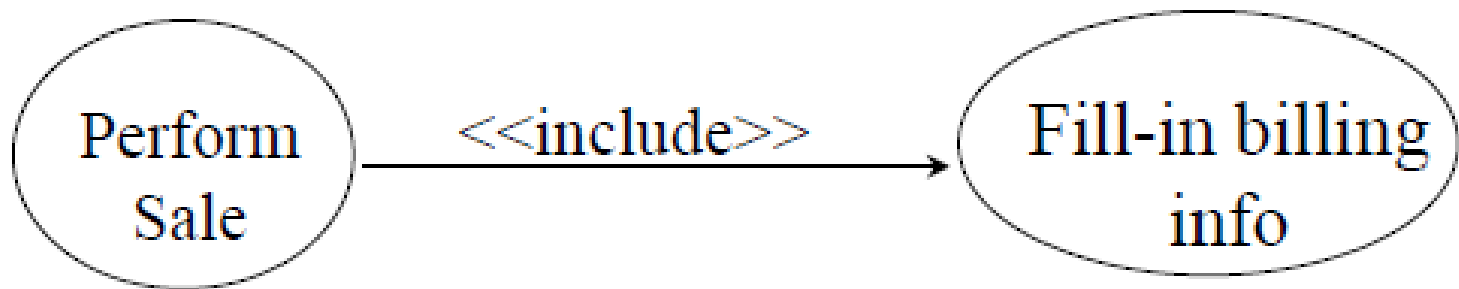
- Goal: enable flexibility in requirements specification
 1. Isolating functionality
 2. Enabling functionality sharing
 3. Breaking functionality into manageable chunks
- Relationships
 1. Include
 2. Extend
 3. Generalization

Include

Goal:

1. Decomposing complicated behavior
2. Centralizing common behavior

the behavior of the included use case is inserted into the behavior of the including use case - The first use case often depends on the outcome of the included use *case*.

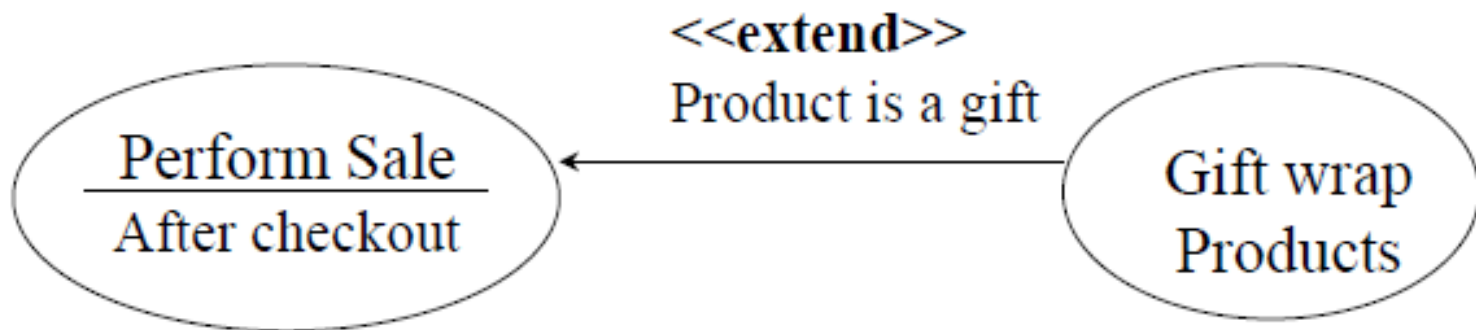


Extend

the behavior of the extension use case may be **inserted** in the extended use case under some conditions

Note the direction of the arrow

The base use-case does not know which use-case extends it



Example

Detailed Use-case

See se-class...

תרחיש שימוש (פורמלי) – דוגמא

שם	הזמנת ספר
שחקן ראשי	קורא
מטרה	קורא מעוניין לשריין ספר מתוך הקטלוג המקוון
הקף	מערכת הספרייה
רמה	משתמש (cockburn)
בעלי עניין ואינטרסים	קורא – לשריין ספר בעל הספרייה – שרות מורחב לרווחת הלקוחות
טריגר	הקורא נכנס למערכת
תנאי-קדם	הקורא עבר את מסך ההזדהות (login) ונחת בעמוד הבית
תנאי סיום מוצלח	הספר שמור עבור הקורא (האם זה תנאי מוצלח?)
תנאי כישלון	הספר אינו שמור

תרחיש שימוש-המשך

<ol style="list-style-type: none"> 1. הקורא לוחץ בתפריט על הזמנת ספר 2. המערכת מציגה קטלוג עם מסך חיפוש 3. הקורא מזין את שם הספר 4. המערכת מציגה התאמות עם מיקומם 5. הקורא בוחר התאמה ובקשה לשמירה 6. המערכת מאשר את ההזמנה ומציגה את הקטלוג בחזרה 	<p>תרחיש הצלחה עיקרי</p>
<ol style="list-style-type: none"> 2א. פג תוקף ה-login 2א.1. המערכת מחזירה את הקורא למסך הכניסה 2א.2. הקורא מתייחס או מנסה שוב 4א. המערכת אינה מוצאת את הספר 5א.1. ... 	<p>הרחבות (שגיאות)</p>
<ol style="list-style-type: none"> 3. הקורא מזין מחבר או נושא 	<p>תרחישים חלופיים</p>

Example: Amazon

Actors?

Base Use Cases?

Include?

Extend?

תרגיל 2 - חלק א'

- תרחיש שימוש לאמזון
- שימוש ב-CASE ליצירת קובץ בתבנית סטנדרטית XMI (ArgoUML, Eclipse/EMF)
- שכפול תבנית התרגיל והגשה עם PR
 - כל חלק בנפרד
 - חלק א' – שבועיים, כל שבוע עוד חלק

סיכום

• UML

– תרחישי שימוש