HY 351: Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων
CS 351: Information Systems Analysis and Design

Functional Modeling

Lecture : 8
Date : 20-10-2005

Yannis Tzitzikas
University of Crete, Fall 2005

Outline

- Why we model ?
- Functional Modeling
- UML Diagrams for functional modeling
  - Activity Diagrams
  - Use Case descriptions and Use Case Diagrams

Τι είναι μοντέλο και γιατί μοντελοποιούμε

- Μοντέλο: Μια αφαίρεση (απλούστευση) της πραγματικότητας
  - εστιάζει στα σημαντικά, κρύβει τις άσχετες πλευρές και τις δευτερεύουσας σημασίας λεπτομέρειες
- Γιατί μοντελοποιούμε;
  - Ένα μοντέλο μας επιτρέπει την καλύτερη κατανόηση ενός συστήματος
  - Συνήθως φτιάχνουμε μοντέλα σύνθετων συστημάτων τα οποία δεν μπορούμε να κατανοήσουμε στην πληρότητα τους (ένεκα των περιορισμένων μας αντιληπτικών και διανοητικών ικανοτήτων)
  - Μοντελοποιώντας περιορίζουμε το πρόβλημα εστιάζοντας σε επιμέρους πλευρές του συστήματος (διαίρει και βασίλευε) και κλίμακες αφαίρεσης.

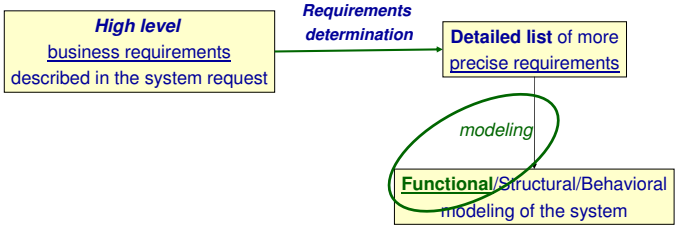
Βασικές Αρχές Μοντελοποίησης

- Η επιλογή του τύπου μοντέλου καθορίζει τον τρόπο μελέτης του συστήματος και τη μορφή της λύσης που θα επιτευχθεί.
  - Αντί αρχιτεκτονικού σχεδίου, μαθηματικές φόρμουλες πίεσης στους πυλώνες
- Κάθε μοντέλο μπορεί να παρασταθεί σε διαφορετικά επίπεδα ακρίβειας
- Καλά μοντέλα είναι εκείνα που συνδέονται με την πραγματικότητα
- Κανένα μοντέλο από μόνο του δεν είναι επαρκές. Κάθε μη τετριμμένο σύστημα προσεγγίζεται καλύτερα από ένα (σχετικά μικρό) σύνολο ανεξάρτητων μοντέλων από διαφορετικές σκοπιές.

Μοντελοποίηση στην Ανάλυση και Σχεδίαση Πλ. Συστημάτων

- Μοντελοποίηση στην Ανάλυση και Σχεδίαση Πλ. Συστ.:
  - Βοηθά την οπτικοποίηση ενός (υπαρκτού ή προς κατασκευή) συστήματος
  - Βοηθά την προδιαγραφή της δομής ή συμπεριφοράς ενός συστήματος
  - Αποτελεί οδηγό για την κατασκευή ενός συστήματος
  - Τεκμηριώνει τις αποφάσεις που έχουμε πάρει

Modeling in IS Analysis and Design



What is Functional Modeling?

Its objective is to describe:

- the business processes and
- the interaction of an information system with its environment

How we model functions in OO Analysis and Design?

Usually we employ two types of (UML) models:

- ***Activity Diagrams***
  - allow the logical modeling of business processes and workflow
- ***Use Cases*** and ***Use Case Diagrams***
  - allow the logical description of the basic functions of the information system w.r.t. its environment

Remarks:

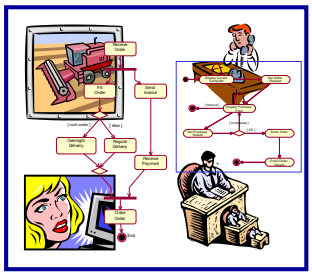
- They are both "**logical**", I.e. they do not specify how the modeled functions will be designed and implemented in the real system
- We could use them in order to describe not only the new system, but also the existing one.

What is the difference between *Activity Diagrams* and *Use Cases*?

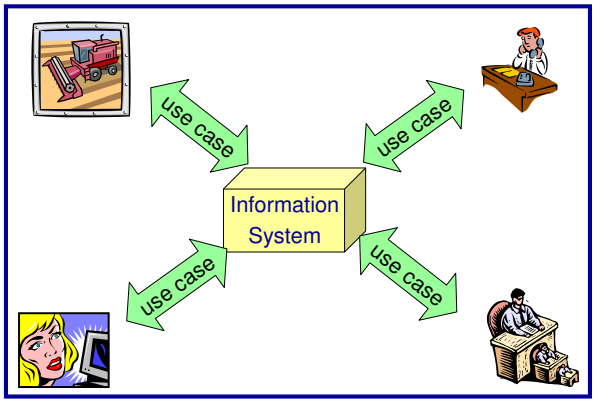
- Activity Diagrams
  - we use them to model the internal functional view of the system
    - how the business system operates
- Use Cases
  - we use them to model the external functional view of the system
    - how it interacts with its environment (users)

An organization

Activity Diagrams can model the business processes of the organization



Use Cases can model the interaction between the IS and its environment



Activity Diagrams

What can we model with Activity Diagrams?

- They could be used to model any process
  - from a high level business process
  - to the flow of control of a method
- They are like *flow charts* with the extra ability to represent parallelism, concurrency and complex decisions.
- They combine ideas from several techniques (not from 3 amigos) like
  - event diagrams, SDL state modeling, workflow modeling, Petri Nets

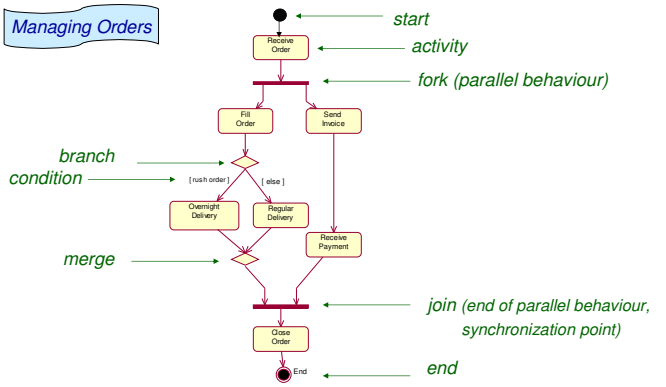
Activities and Activity Diagrams

Activity (or activity state)
- is a state of doing something: either a real-world process (typing a letter) or the execution of a software routine
- it could represent a *manual* or *computerised* activity
- each activity has a name
  - usually: **Verb + Noun** (e.g. "Set Appointment", "Receive Order")

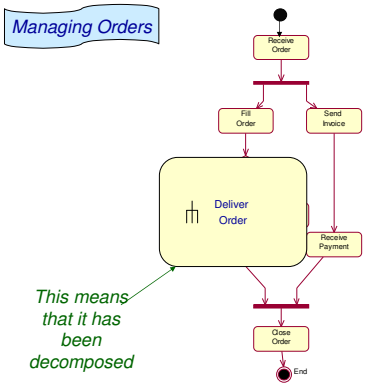
Activities vs Actions
- Actions: non decomposable
- Activities: can be decomposed into a set of activities and/or actions.

Activity Diagram
- describes the sequencing of activities with support of both conditional and parallel behaviour

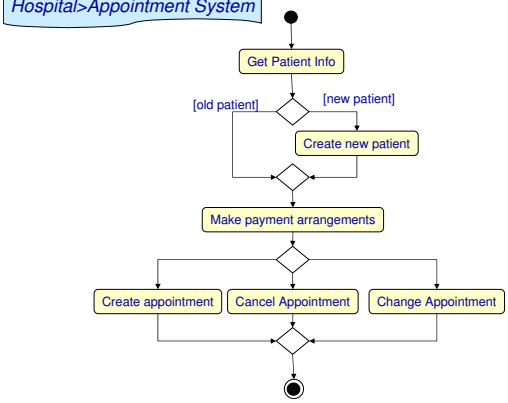
An Activity Diagram



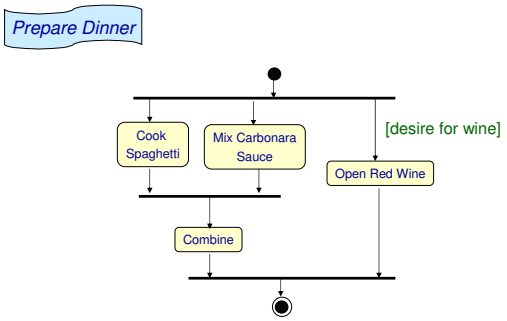
An Activity Diagram



Another Activity Diagram



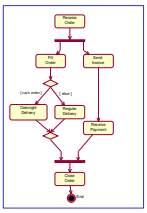
Another Example with conditional thread



Swimlanes

Activity diagrams describe what happens but not who does what

- it does not say which agent (or class) is responsible for what

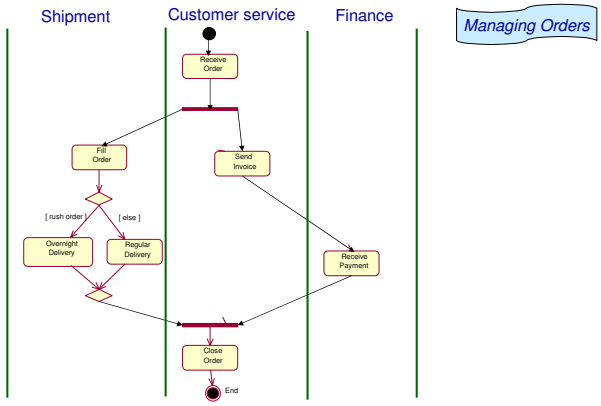


?

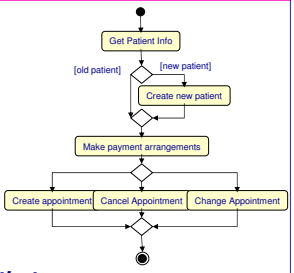
=> **Swimlanes**

- arrange the activity diagram in vertical zones. Each zone represent the responsibilities of a particular agent or class.
  - V.2: Partitions

Swimlanes (example)



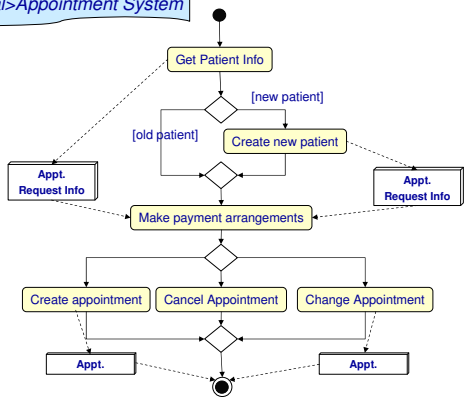
Object flow



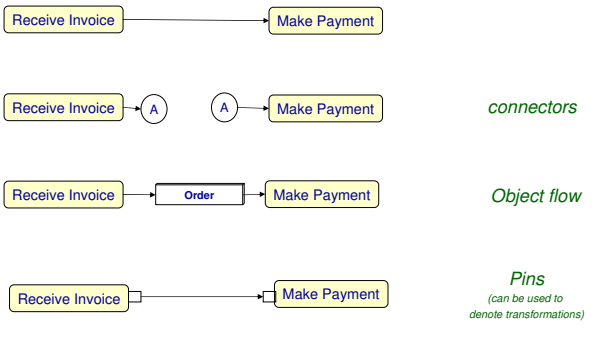
How can also show the objects that are in involved in the flow of control ?

Adding Object Flow

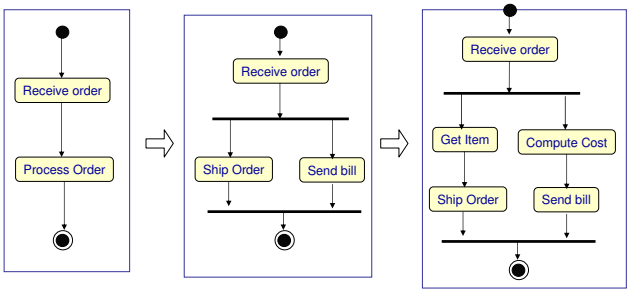
Hospital>Appointment System



Different but equivalent ways to show an edge

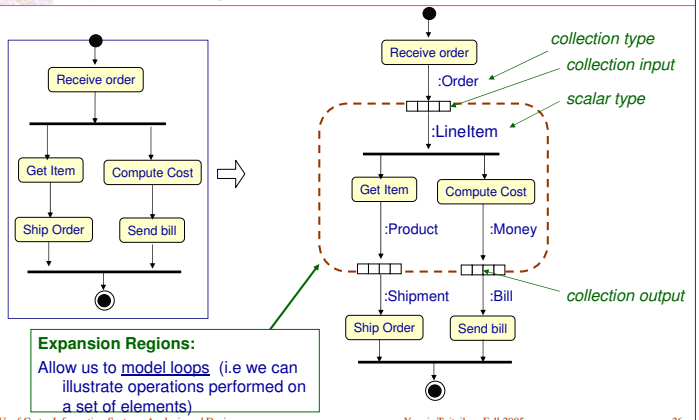


Examples of analyzing an activity



Something is missing here!

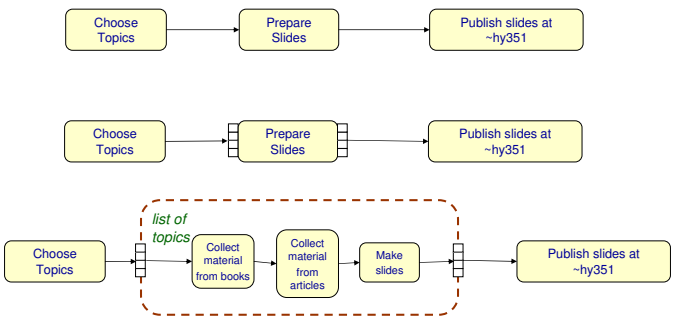
Examples of analyzing an activity
Expansion Regions



Expansion Regions:
Allow us to model loops (i.e we can illustrate operations performed on a set of elements)

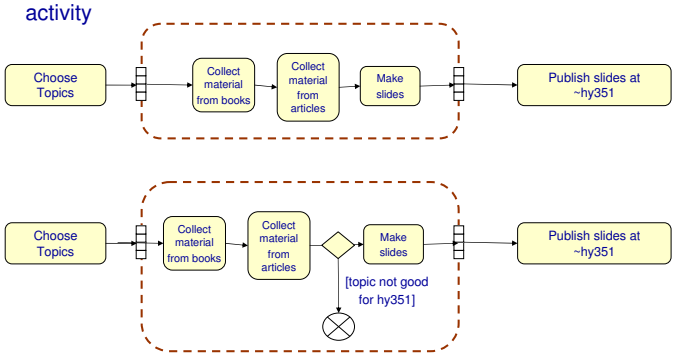
Another example with expansion regions

Prepare HY351



Expansion Regions and Flow finals

Indicate the end of a particular flow without terminating the whole activity

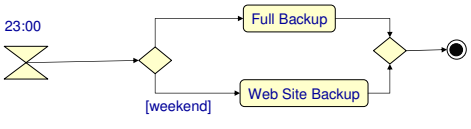


Signals

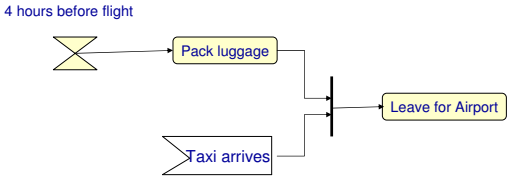
- Activity diagrams have a clearly defined start point
  - which could be the invocation of a program or routine
- However, we could also have *actions that respond to signals.*

Types of Signals
- **Time signal**
  - may indicate the end of a month, the end of a financial period, or the end of each microsecond in a real time controller
- **Accept signal**
- **Send signals**

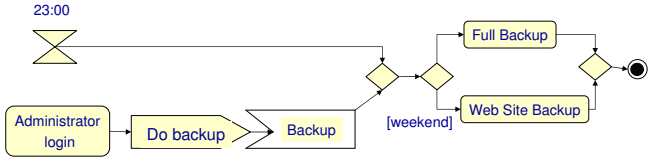
Example with Time Signal



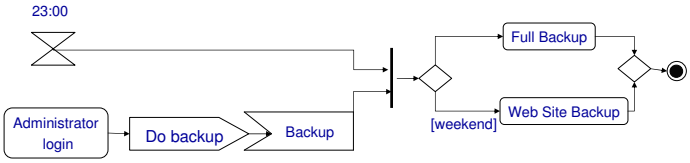
Example with Time and Accept signals



Example with Time and Send Signal



Example with Time and Send Signal (2)

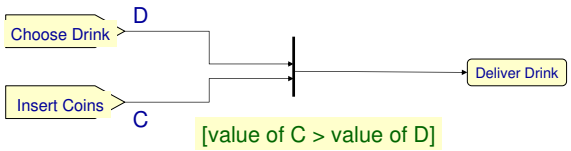


What's the difference?

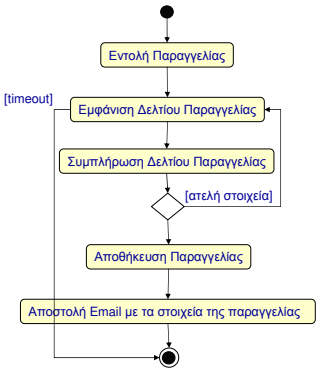
Join specifications

Join specifications

Boolean expressions attached to joins



Παράδειγμα



Business Process Modeling with Activity Diagrams

Activity Diagrams and IS Development Methodologies

- Object-Oriented Methodologies
  - Activity diagrams are used to model the behaviour in a business process independent of the objects
- Structured Methodologies
  - They could be considered as *sophisticated data flow diagrams* (that are used in conjunction with structured analysis)

- *Αρχικά η Αντικειμενοστρεφής Αν&Σχ δεν περιελάμβανε Διαγράμματα Δραστηριοτήτων (αλλά μόνο Περιπτώσεις Χρησης)*
- *Σήμερα η τάση είναι να φτιάχνουμε μερικά από τις απαιτήσεις αφού συχνά βοηθούν την επικοινωνία με τον πελάτη*
  - *Δεν πρέπει όμως να υπερβάλλουμε και να καταλήξουμε να κάνουμε αναλυτική αποσύνθεση λειτουργιών (functional decomposition)*

In most cases each activity will be associated with a use case

When to use and use not Activity Diagrams

- When to use activity diagrams
  - for understanding and illustrating business workflow
  - for illustrating the flow of activities in a use case
  - for modeling multithreaded programming
  - for modeling a complicated sequential algorithm (flow chart)
- When use not activity diagrams
  - for describing how objects collaborate (use Interaction Diagrams instead)
  - for describing how an object behaves over its lifetime (use a State Diagram instead)
  - for representing complex conditional logic (use a Truth Table instead)

Guidelines for Creating Activity Diagrams

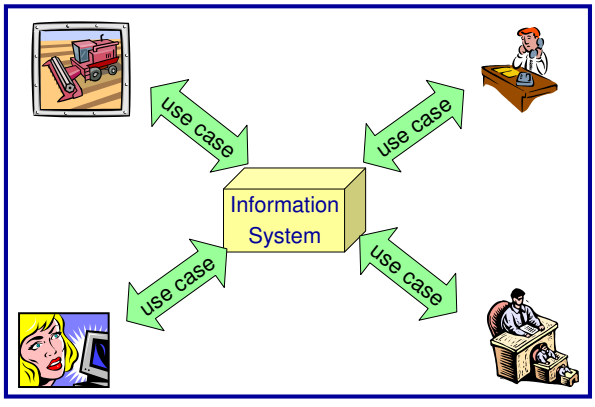
1/ Determine the scope of the diagram
2/ Set the title of the diagram
3/ Identify activities and control flows that occur between activities
4/ Identify any decisions that are part of the process being modeled
5/ Identify any prospects for parallelism in the process
6/ Draw the activity diagram

Other tips:

- *try to minimize line crossings and enhance readability*
- *verify the correctness of miracle activities (those with no input) and black hole activities (those with no output)*

Use Cases

Use Cases



What we can model with Use Cases?

- They model the external functional view of the system
  - how it interacts with its environment (e.g. with the users of the system)
  - they describe what the user can do and how the system responds
- They have become primary element in project development and planning

Distinctions
- Use Cases
- Use Case Diagrams

What is a Use Case?

- **Use Case** = a set of scenarios tied together by a common user goal
- **Scenario** = a sequence of steps describing an interaction (user vs system)
  - ***E.g.: The customer browses the catalog and adds desired items to the shopping basket. When the customer wishes to pay, the customer describes the shipping and credit card information and confirms the sale. The system checks the authorization on the credit card and confirms the sale both immediately and with a follow-up email.***

Contents and Format of a Use Case

- A UC usually contains:
  - a common all-goes-well scenario
  - alternatives that may include things going well
  - alternative ways that things go well
- The format for a UC
  - sequence of numbered steps
  - alternatives as variations of that sequence
  - [add a line for preconditions (that should be true when the UC can start)]
- ***How detailed a UC should be ?***

  The more risk the more detail we need

An example Use Case description

Buy a Product
1. Customer browses through catalog and selects items to buy
2. Customer goes to check out
3. Customer fills in shipping information (address, next-day or 3-day delivery)
4. System presents full pricing information, including shipping
5. Customer fills in credit card information
6. System authorizes purchase
7. System confirms sale immediately
8. System sends confirming email to customer

Alternative: *Authorization Failure*
At step 6, system fails to authorize credit purchase
Allow customer to re-enter credit card information and re-try

Alternative: *Regular Customer*
3a. System displays current shipping information, pricing information, and last 4 digits of credit card information
3b. Customer may accept or override these details
Return to primary scenario at step 6

Types of Use Cases

Overview vs Detail
- Overview Use Cases:
  - high level, specified at the beginning of the project, very brief
- Detailed Use Cases
  - refine and specify exactly overview use cases

Essential vs Real
- Essential Use Cases (implementation independent)
  - describe the minimum essential issues necessary to understand the functionality
  - (else called Business Use Cases)
- Real Use Case
  - describe specific set of steps
    - (e.g. the secretary can add an MS Office doc to the repository and then search ..)
  - (else called System Use Cases)

Elements of a Use-Case Description

Use Case Name:	ID:	Importance Level:
Primary Actor:	Use Case Type:	
Stakeholders and Interests:		
Brief Description:		
Trigger:		
Relationships: (Association, Include, Extend, Generalization)		
Normal Flow of Events:		
Subflows:		
Alternate/Exceptional Flows:		

Taken from Dennis et al. 2005

Παράδειγμα: Κλείσιμο Ιατρικού Ραντεβού (1/3)

Όνομα Περίπτωσης Χρήσης: ***Κλείσιμο Ραντεβού*** Αριθμός: **2** Σπουδαιότητα: **Υψηλή**
Κύριος Δράστης: ***Ασθενής*** Τύπος Περ. Χρήσης: ***Αναλυτική, Ουσιώδης***
Εμπλεκόμενοι και Στόχοι: ***Ασθενής: θέλει να ορίσει, αλλάξει ή ακυρώσει ένα ιατρικό ραντεβού***
Σύντομη Περιγραφή: ***Ο ασθενής θέλει να ορίσει, αλλάξει ή ακυρώσει ένα ιατρικό ραντεβού***
Ερέθισμα (συμβάν ενεργοποίησης): ***Ο ασθενής συνδέεται και επιλέγει PANTEBOY*** Τύπος ερεθίσματος: ***Εξωτερικό***
Σχέσεις: Συσχέτιση (association) : ***Ασθενής*** Περιλαμβάνει (include) : ***«Διευθέτηση Τρόπου Πληρωμής»*** Επεκτείνει (extend) : ***«Δημιουργία Νέου Ασθενούς»*** Εξειδικεύει (generalizaiton) :

Παράδειγμα: Κλείσιμο Ιατρικού Ραντεβού (2/3)

Φυσιολογική ροή γεγονότων:
1. Ο ασθενής συνδέεται με τον ιστόχωρο *2. Ο ασθενής πληκτρολογεί το όνομα του και τον αριθμό μητρώου του* *3. Το σύστημα επιβεβαιώνει ότι ο ασθενής είναι καταγεγραμμένος στη βάση* *4. Το σύστημα εκκινεί την περίπτωση χρήσης «Διευθέτηση Τρόπου Πληρωμής»* *5. Ο ασθενής επιλέγει τι θέλει να κάνει:* *αν θέλει νέο ραντεβού, δες P1* *αν θέλει ακύρωση ραντεβού, δες P2* *αν θέλει αλλαγή ραντεβού δες P3* *6. Το σύστημα εμφανίζει τα αποτελέσματα του 5 στον χρήστη*
Συνιστώσες ροές γεγονότων
P1: Νέο ραντεβού *1. Το σύστημα ρωτά τον χρήστη για την επιθυμητή ημερομηνία και ώρα* *2. Το σύστημα προγραμματίζει το νέο ραντεβού λαμβάνοντας υπόψη τις επιθυμίες του ασθενή και τις διαθέσιμες ημερομηνίες*

Παράδειγμα: Κλείσιμο Ιατρικού Ραντεβού (3/3)

P2: Ακύρωση ραντεβού *1. Το σύστημα ρωτά τον χρήστη την ημερομηνία του προς ακύρωση ραντεβού* *2. Το σύστημα βρίσκει και διαγράφει την αντίστοιχη εγγραφή από το πρόγραμμα των ραντεβού* *P3: Αλλαγή ραντεβού* *1. Το σύστημα εκτελεί το P2* *2. Το σύστημα εκτελεί το P1*
Εναλλακτικές Ροές - Εξαιρέσεις *3α: Το σύστημα εκτελεί την περίπτωση χρήσης «**Δημιουργία Νέου Ασθενούς**»* *P1.2α1: Το σύστημα προτείνει εναλλακτικές ημερομηνίες/ώρες* *P1.2α2: Ο ασθενής επιλέγει μια από αυτές ή αποφασίζει να μην κλείσει ραντεβού*

Guidelines for Creating Use Case Descriptions

- *Write each step in "Subject-Verb-Object" form*
- *Clarify who is the initiator in each step*
- *Describe the steps as if you were an independent external observer*
- *Write at the same level of abstraction*
- *Ensure that there is a sensible set of steps, like:*
  - *the primary actor initiates the dialog and sends data to the system*
  - *the system checks the validity of the data/request*
  - *the system processes the request*
  - *the system returns to the actor the results of the processing*
- *Apply KISS principle liberally*
- *Write repeating instructions after the set of steps to be repeated.*

Adapted from Dennis et al. 2005

Use Case Diagrams

Use Case Diagrams

Purpose:

- For visualising several Use Cases (now part of UML)
- However they are not necessary (in order to use UC)

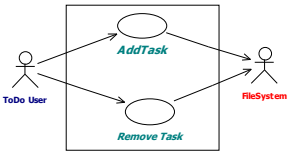
A Use Case Diagram is a graph having as nodes:

- **Actors**
- **Use Cases**

and edges:

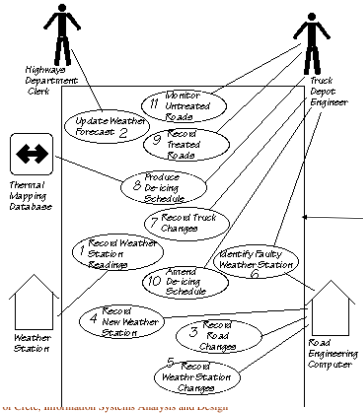
- between Actors and Use Cases, and
- between Use Cases
  - ***include, ISA, extend***

Use Case Diagrams > Actors



- **Actors** carry out Use Cases
- Actor = role a user plays with respect to the system
  - A user may play a lot of roles
  - Actors are not always humans:
    - can be an external system that needs some information from the current system
- Hints
  - The number of actors is subjective
  - A method that helps us identify use cases is to think about the external events (from outside world) to which we want to react

Use Case Diagram



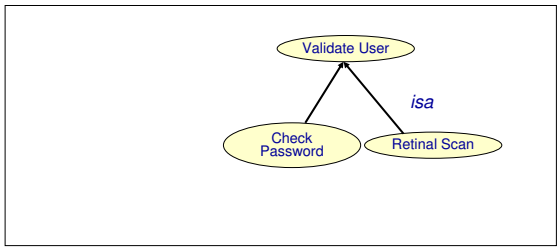
Boundary of the system

Taken from Volere Requirements Specification Document

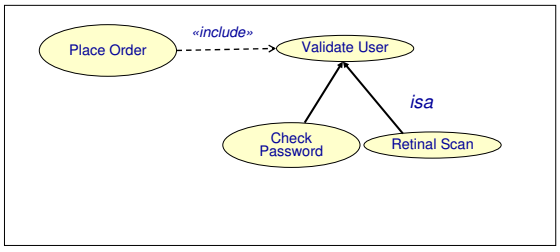
Use Case Diagrams > UC Relationships

- Include $\subseteq$ **UC** $\times$ **UC**
  - to avoid describing the same chunk of behaviour many times, or to avoid copy-paste
- ISA $\subseteq$ **UC** $\times$ **UC**
  - the specialized can override any part of the base UC, although it should still be about the same essential user goal
- extend $\subseteq$ **UC** $\times$ **UC**
  - like ISA. The extending UC may add additional behaviour but only on declared "***extension points***" of the base UC

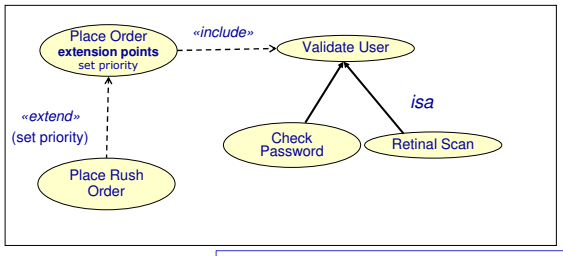
Example: Generalization

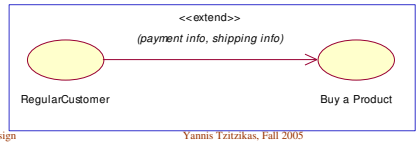


Example: Generalization + Include

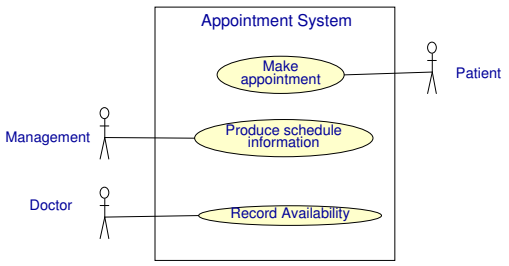


Example: Generalization + Include + Extend

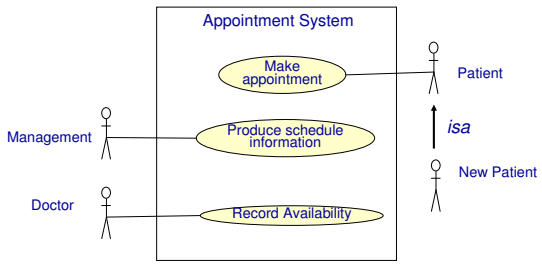




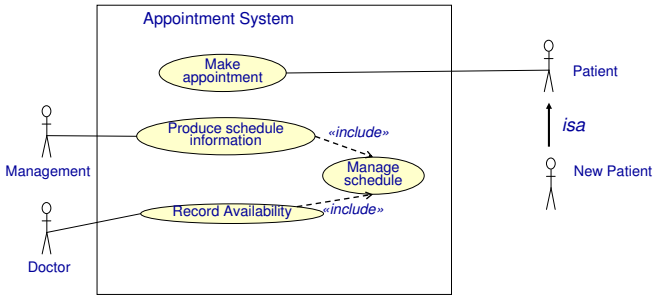
Use Case Diagram for an Appointment System



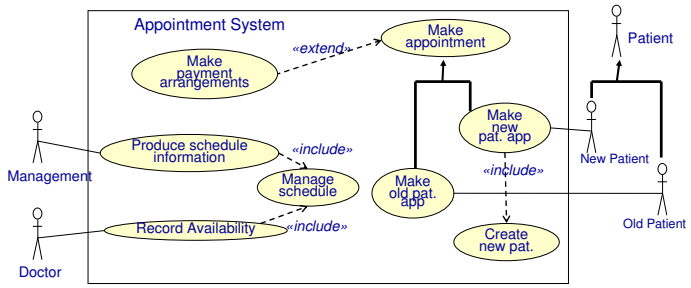
Use Case Diagram for an Appointment System Specializing an actor



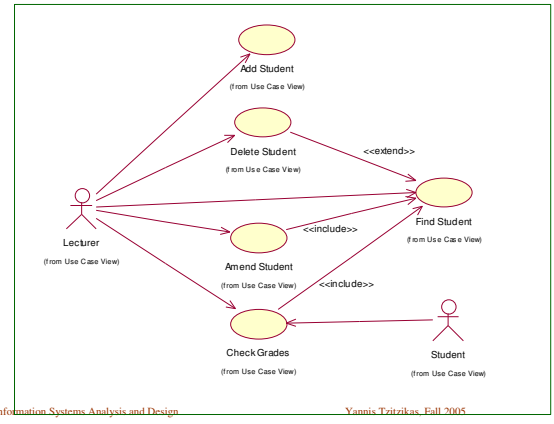
Use Case Diagram for an Appointment System adding «include»



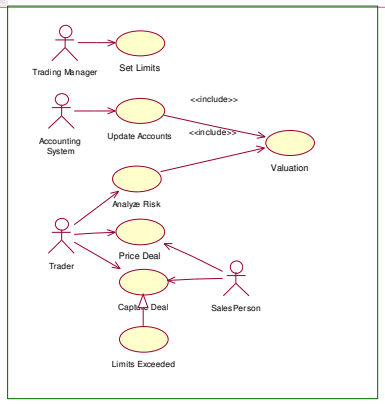
Use Case Diagram for an Appointment System adding «Isa» and «extend»



Another Example



Another example



Creating
Use Cases and Use Case Diagrams

Methodological Issues

- One could first start by defining the Use Case Diagram and then describe each Use Case
- How many Use Cases should we define ?
  - For one10-person year project ≈ 12 Use Cases

Steps in Writing Use Cases

A/ Identify the major UCs	1 Review the Activity Diagram 2 Identify the system's boundaries 3 List the primary actors and their goals 4 Identify and write the major use cases 5 Carefully review use cases
B/ Expand the major UCs	6 Choose one major use case to expand 7 Fill in details on the use-case template 8 Fill in the steps of the normal flow of events 9 Normalize the size of each step 10 Describe alternate or exceptional flows 11 Simplify and organize as necessary
C/ Confirm the major UCs	12 Review the current set (check semantics, involve the users) 13 Iterate the entire set of steps
D/ Create the UC diagram	14 Start with system boundary 15 Place elements in order to be easy to read 16 Place actors on the diagram, connect them with UCs

Some Advantages of Use Cases

- Instead of having to model or understand the functioning of the entire organization, we try to understand and study each Use Case separately
- We can exploit Use Cases in order to estimate the time and effort needed to design and implement the system
- Project managers can control the progress of the project by inspecting the progress of each individual Use Case

Reading and References

- **Systems Analysis and Design with UML Version 2.0** (2nd edition) by A. Dennis, B. Haley Wixom, D. Tegarden, Wiley, 2005. CHAPTER 6
- **UML Distilled: A Brief Guide to the Standard Object Modeling Language** (3rd Edition) by Martin Fowler, Addison Wesley, 2004. Chap. 11 (2nd edition: Chapter 9.)
- **The Unified Modeling Language User Guide** (2nd edition) by G. Booch, J. Rumbaugh, I. Jacobson, Addison Wesley, 2004, Chap 20