UNIVERSITY of HOUSTON

DEPARTMENT OF COMPUTER SCIENCE

COSC 4351 Fall 2023

Software Engineering

M & W 4 to 5:30 PM

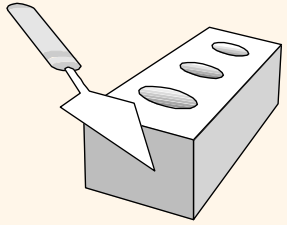
Prof. Victoria Hilford

PLEASE TURN your webcam ON

NO CHATTING during LECTURE

COSC 4351

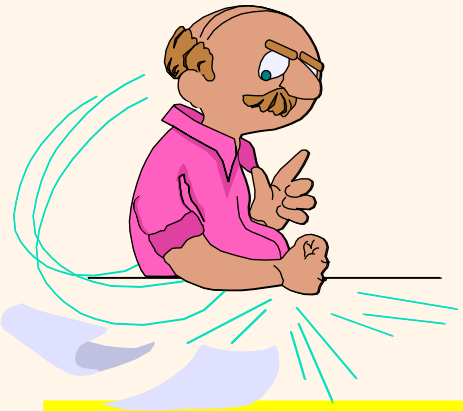
4 to 5:30



PLEASE

LOG IN

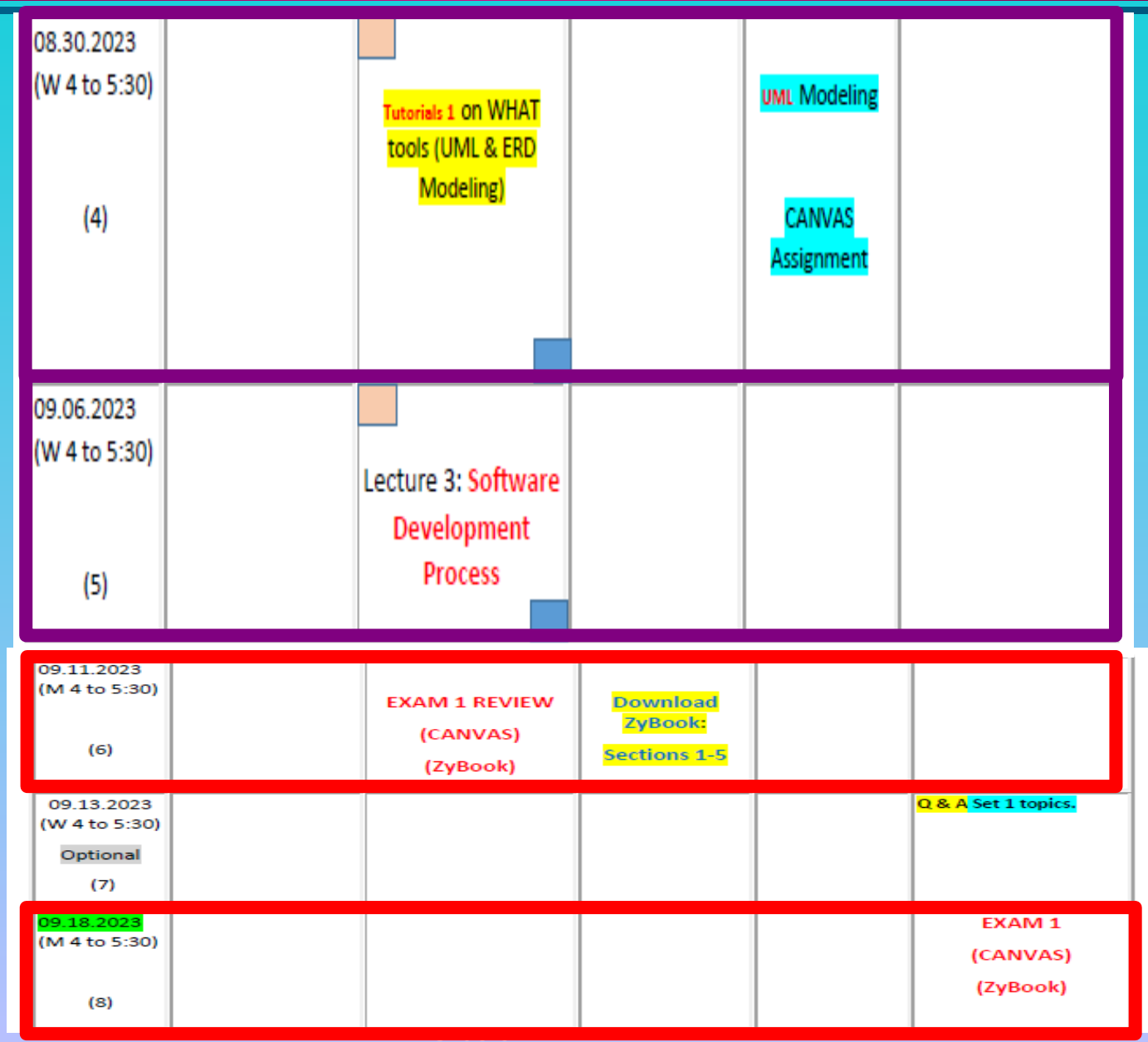
CANVAS



Youyi [A-L]

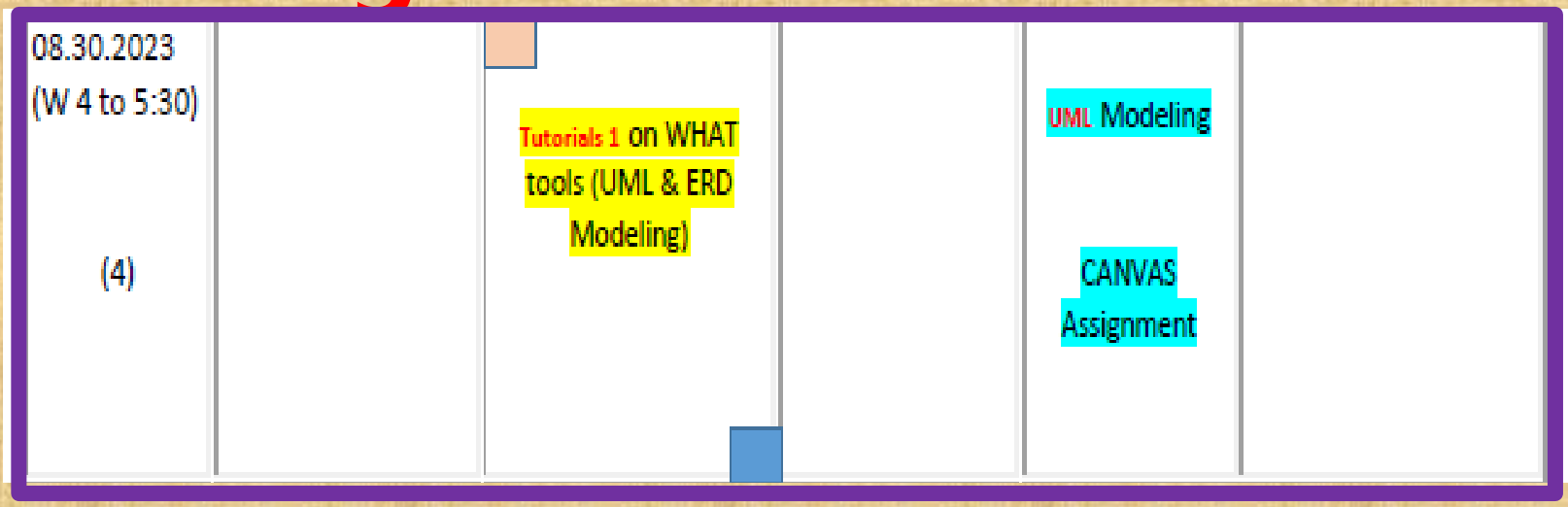
Kevin [M-Z]

Please close all other windows.


08.30.2023
(W 4 to 5:30)
(4)
Tutorials 1 on WHAT tools (UML & ERD Modeling)
UML Modeling
CANVAS Assignment
09.06.2023
(W 4 to 5:30)
(5)
Lecture 3: Software Development Process
09.11.2023
(M 4 to 5:30)
(6)
EXAM 1 REVIEW
(CANVAS)
(ZyBook)
Download ZyBook:
Sections 1-5
09.13.2023
(W 4 to 5:30)
Optional
(7)
Q & A Set 1 topics.
09.18.2023
(M 4 to 5:30)
(8)
EXAM 1
(CANVAS)
(ZyBook)

Class 4

TUTORIAL 1 ON UML & ERD Modeling



08.30.2023 (W 4 to 5:30) (4)		Tutorials 1 on WHAT tools (UML & ERD Modeling)		UML Modeling CANVAS Assignment

CLASS PARTICIPATION 20 points — 20% of Total

UML Modeling PRACTICE

CLASS PARTICIPATION 20% Module | **Available until** Aug 30 at 4:00pm | **Due** Aug 30 at 4pm | 100 pts

Please complete as much as possible. Consider it the first iteration.

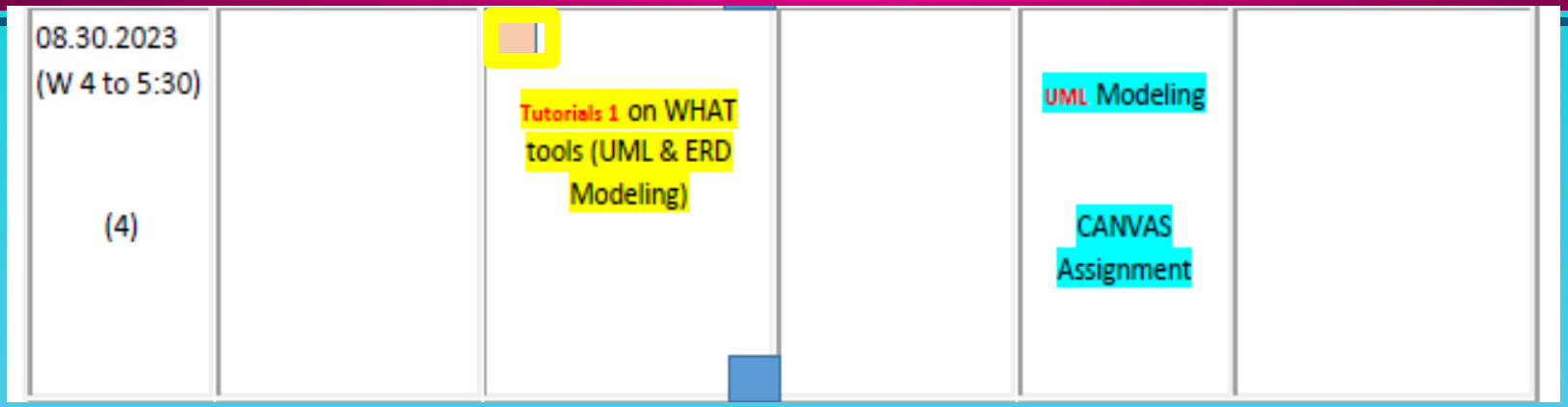
You **must use Microsoft Word** Case Tool.

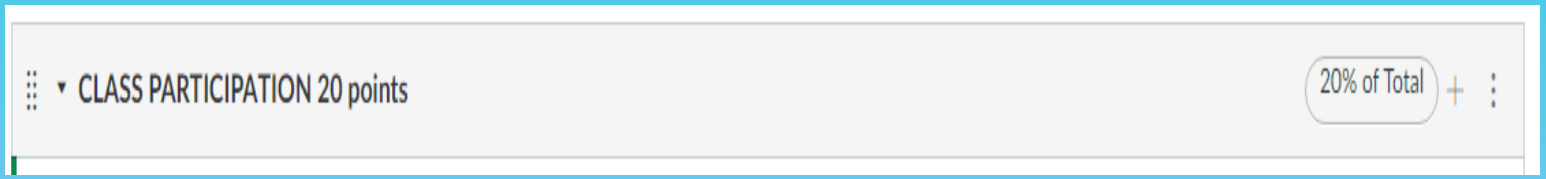
You must attach both the .docx and .pdf.

TA will not grade unless both files are submitted.

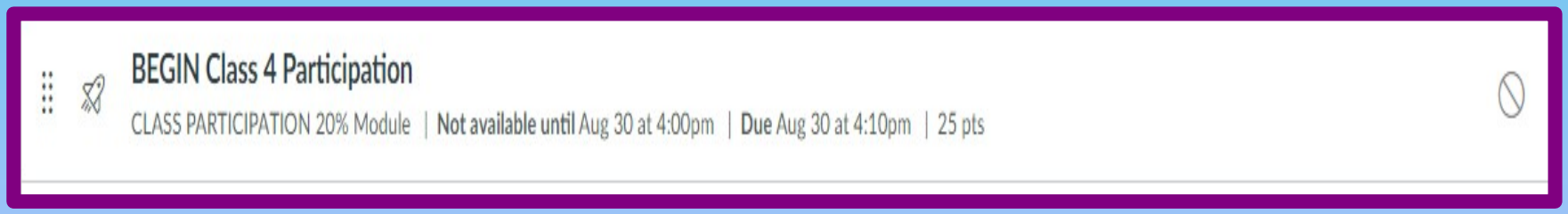
- UML Modeling - WHAT Practice.docx

From 4:00 to 4:10 – 10 minutes.



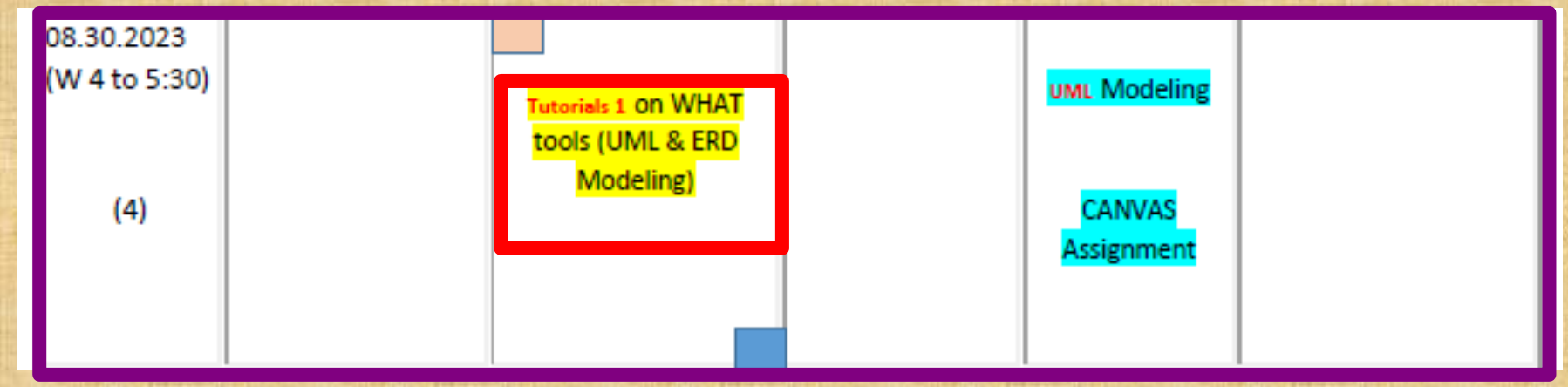


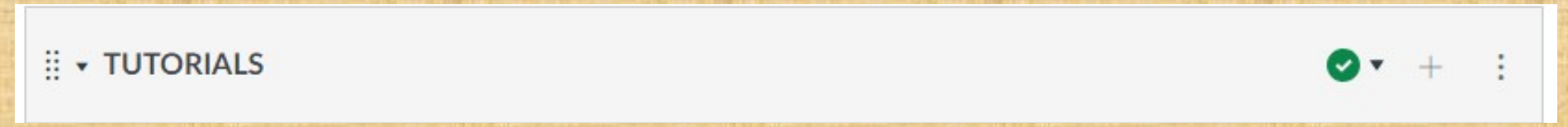
PASSWORD: IN TEAMS

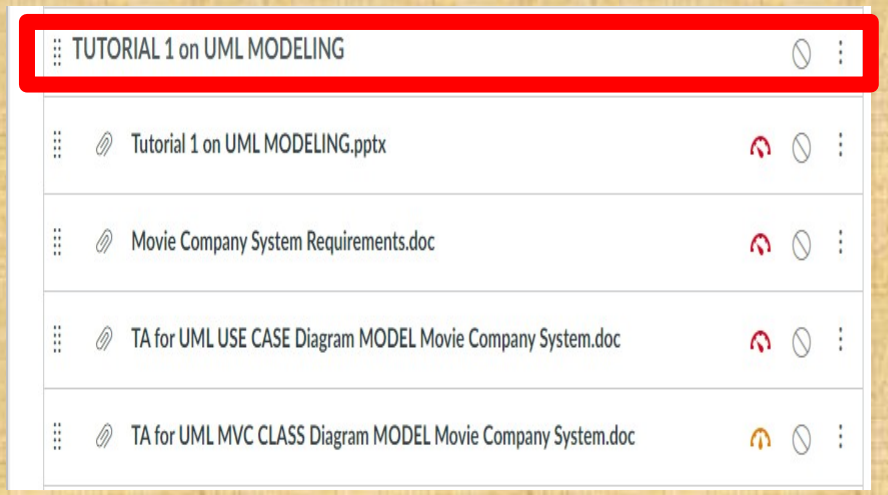


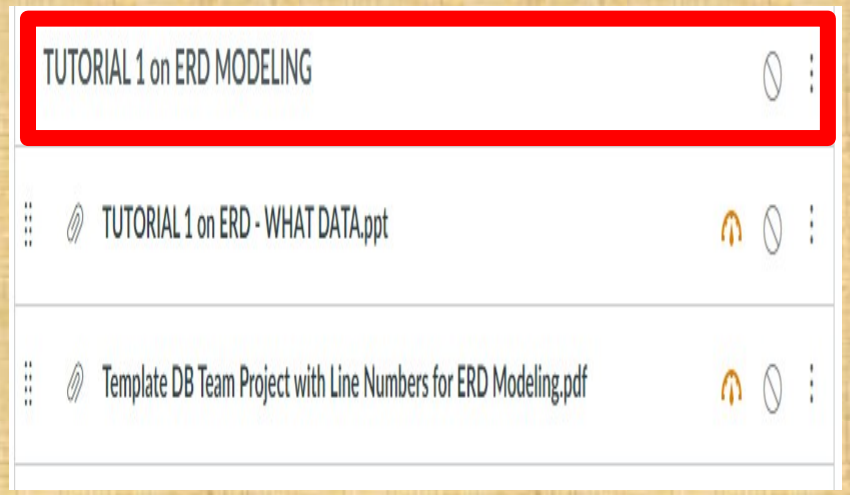
VH, publish.

From 4:10 to 5:00 PM – 50 minutes.



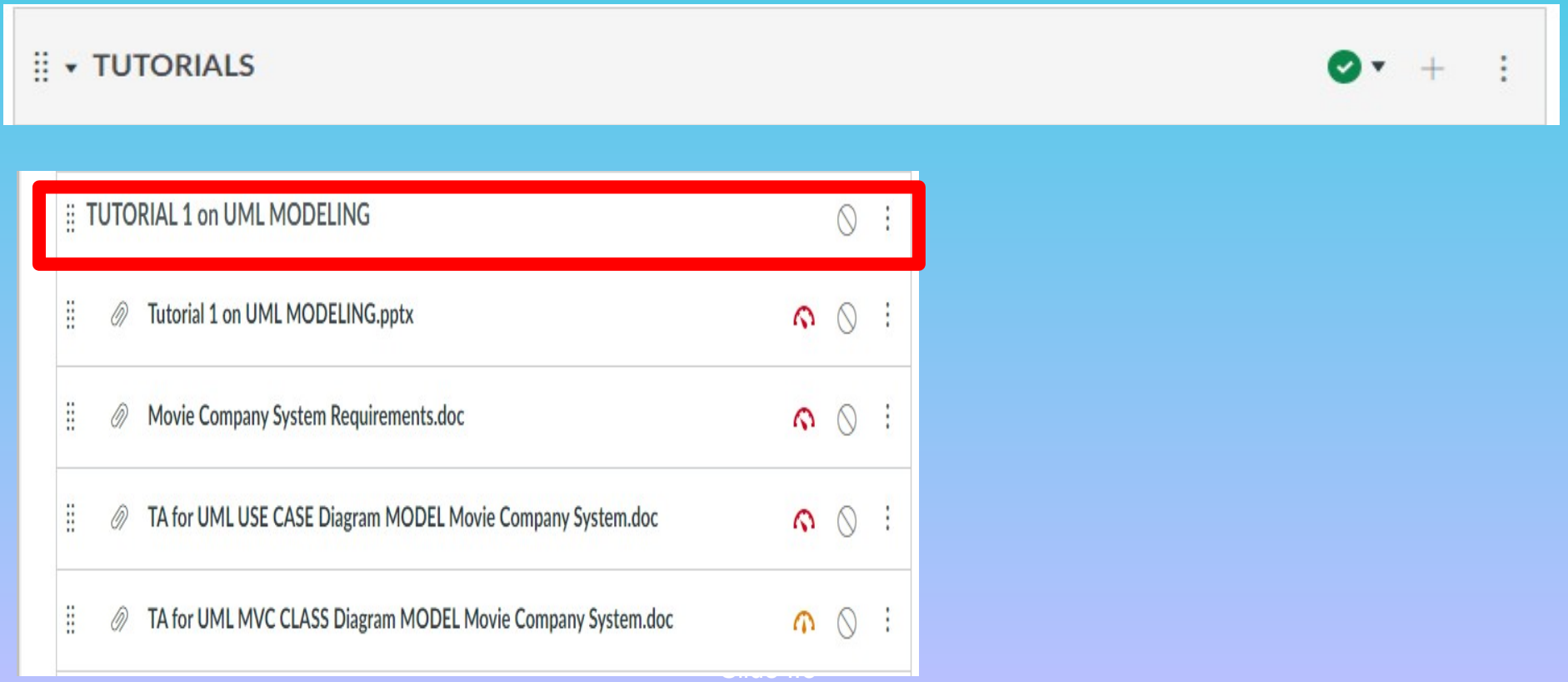






From 4:10 to 4:40 PM – 30 minutes.

UML Modelling



HealthCare.gov October 21, 2013

http://www.newyorker.com/online/blogs/elements/2013/10/why-the-healthcaregov-train-wreck-happened-in-slow-motion.html

Early in a project, there is a phase in which the client and the contractor work together to create a description of what is to be built. This is called the specification and building a complex software product without a clear, fixed set of specifications is impossible. The *Times* reported that

UML Modeling

> the biggest contractor, CGI Federal, was awarded its $94 million contract in December 2011. But the government was so slow in issuing specifications that the firm did not start writing software code until this spring.... As late as the last week of September, officials were still changing features of the Web site.

This is like being told to build a skyscraper without any blueprints, while the client keeps changing the desired location of things like plumbing and wiring.

The *Times* also quoted an "insurance executive working on information technology," who said that "we foresee a train wreck" because "we don't have the I.T. specifications." He also said that "the political people in the administration do not understand how far behind they are." I've been a software contractor for the better part of fifteen years, and no one spends that long developing software without being involved in a few troubled projects. One thing they all have in common is that "train wrecks" are never a surprise to anyone working on them. They are not discrete events; they are part of drawn out processes. We only saw the wreckage of Healthcare.gov on October 1st, but the contractors have been working on a wreck for almost two years.

To Victoria Hilford
Cc
Sent Wednesday, July 1, 2015 9:41 PM
Subject Thank You!

UML Modeling

Hi my name is Jackson and I was one of your students in your COSC 1320 Summer 2015 class and I just wanted to let you know how grateful I am to have taken your class. For the first time I have actually been very motivated to learn more and more about programming as before I actually did rather poorly in my other computer classes. Programming language just seem very alien to me and whenever I tried to practice coding I would sit there not having a clue as to what kind of programs to make, so it made it very hard for me to improve and stay motivated. Now I can't stop thinking about code. One time when I walked into my favorite restaurant and saw their rewards system and immediately my brain started to think about how to write the program in C++ or JAVA and even ways to optimize it and make it better.

One of the skills that I learned in your class that I found very helpful was the UML Modeling. When you first introduced it I found it to be very tedious and useless. It wasn't until I started doing the programming assignments that I realized how useful it actually was. Before, when I did the programming assignments in my other computer courses I was just overwhelmed with details as I tried to take in everything at once and on top of that when I started to actually write my program I had no idea where to start, overall I just felt very lost. After I learned UML Modeling every time I saw a program I would subconsciously break down the program into a UML model and it helped me tremendously in understanding what was going on. I never had that feeling of being lost again. I always knew what I should be doing and how things were working together.

There is still a lot I need to learn and a lot of things I need to relearn, but I am very eager to study these things throughout the rest of my summer break.

Thank you for giving me the motivation to learn more and more about Computer Science.

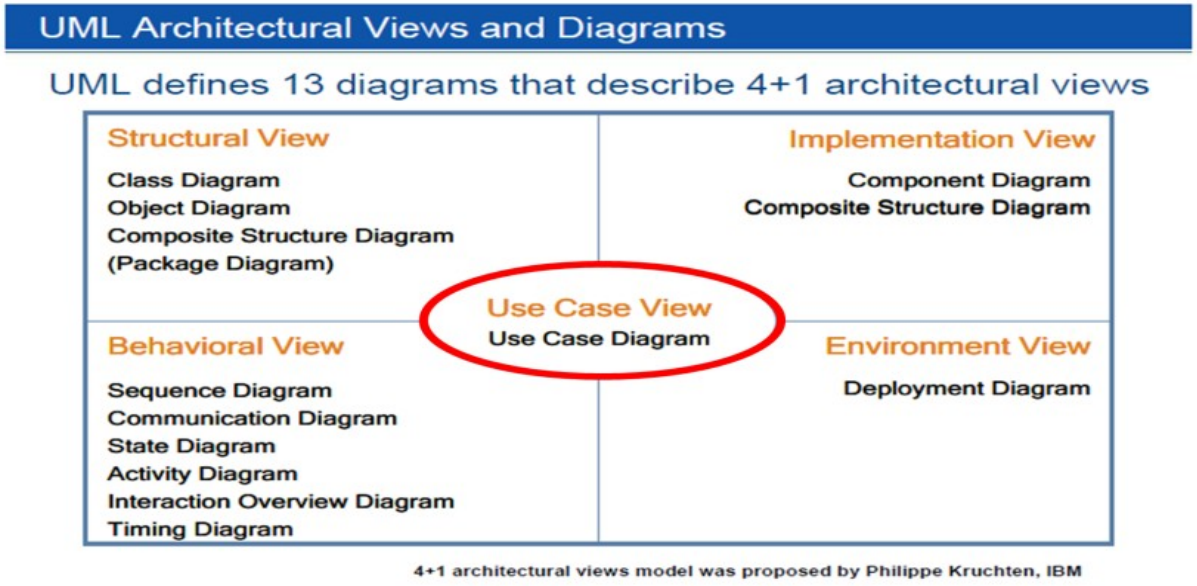
The Unified Modeling Language

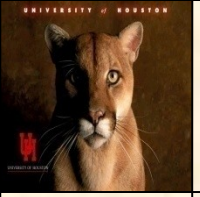
The **Unified Modeling Language** (**UML**) has become a **standard** notation for **Object Oriented software Analysis (OOA) & Design (OOD – "blue print")**

It is **OO Language independent**

It includes various types of **UML Diagrams** (**Models**) which use **specific icons and notations**

However, it is **flexible** – the details you include in a given **Diagram** (**Model**) depend on what you are trying to capture and communicate

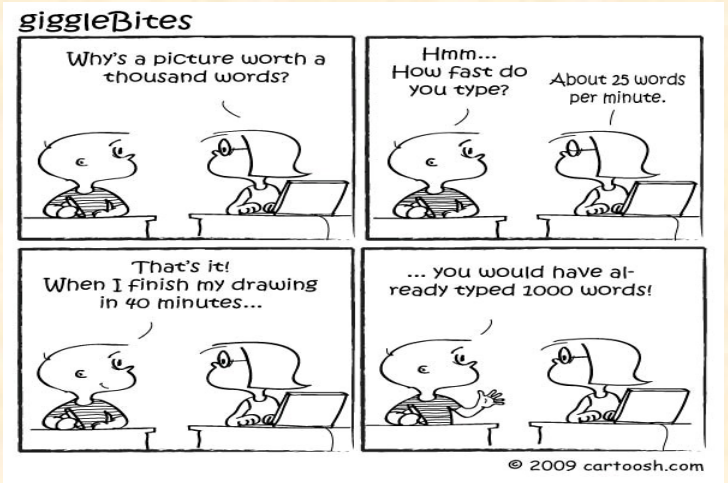


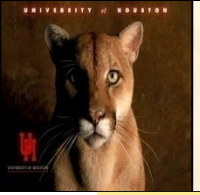


Start with OOA&OOD or Coding?

- The heart of Object Oriented Paradigm **problem** solving is the construction of **a Model**.
- Architects **design** buildings. **Builders** use the **designs** to create **buildings**.
- **Blueprints** are the standard graphical language that both **architects** and **builders** **must learn** as part of their trade.
- Writing **software** is not unlike constructing a **building**.
- Writing **software** begins with the construction of **a Model**. **A Model** is an **abstraction** of the underlying **problem**.
- Unified Modeling Language (**UML**) is a pictorial language used to make **software blueprints**.

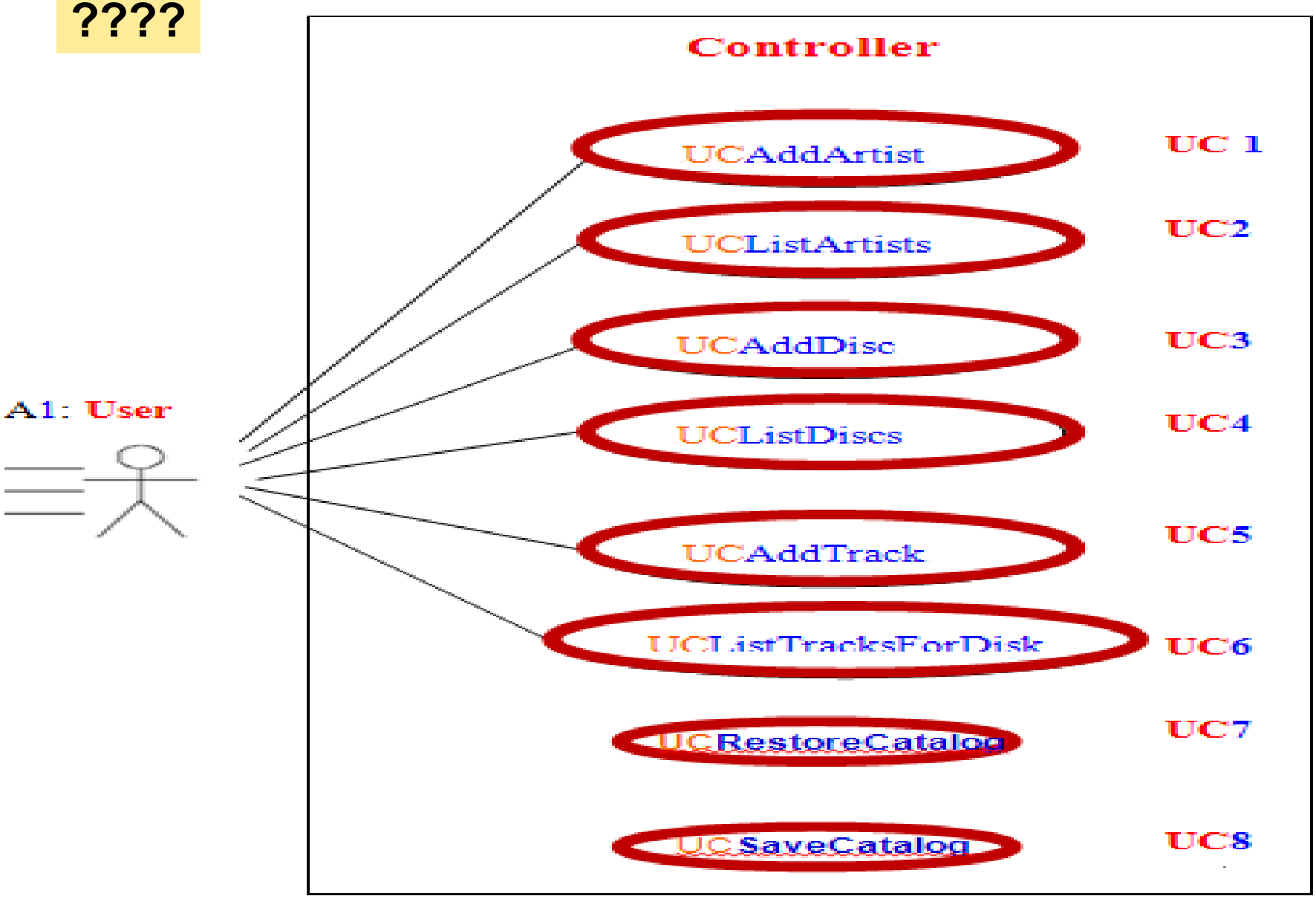
"A picture is worth a thousand words"





UML USE CASE DIAGRAM MODEL

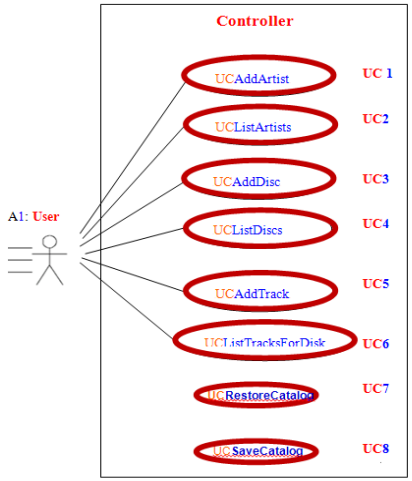
????



UML USE CASE Description

UC2 UCListArtists Description

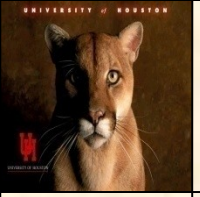
Name:	**UCListArtists**
Actor:	**User**
Description:	**This use case describes the process used by User to List all Artists**
Successful Completion:	**User requests Listing Artists** **1. Movie Company System displays all the Artists by traversing the artists array**
Alternative:	
Pre-Condition:	**User requests List Artists**
Post-Condition:	**Artists displayed**
Assumptions:	**None**



User Story

Step 4: UML Use Cases Description

UML MVC CLASS DIAGRAM MODEL



OO Analysis & Design: UM

OpenAI

Whoa there! You might need to wait a bit.

eiving more requests than we are are comfortable with! To try your request again, come back in a short while and reload this page.

TECHNICAL DETAILS

Hide details for this error

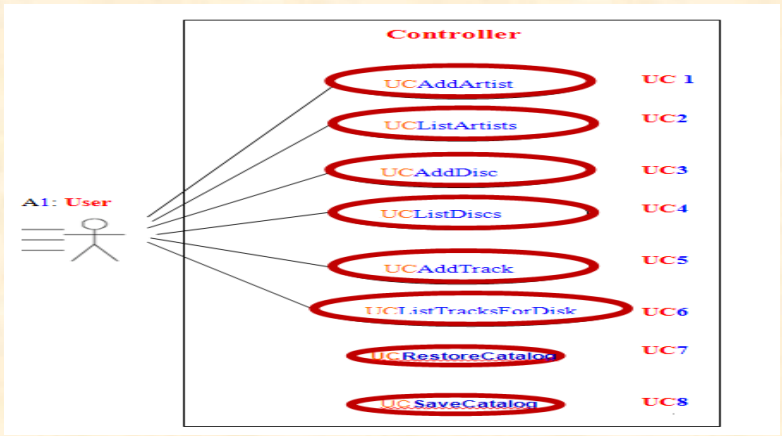
SUPPORT

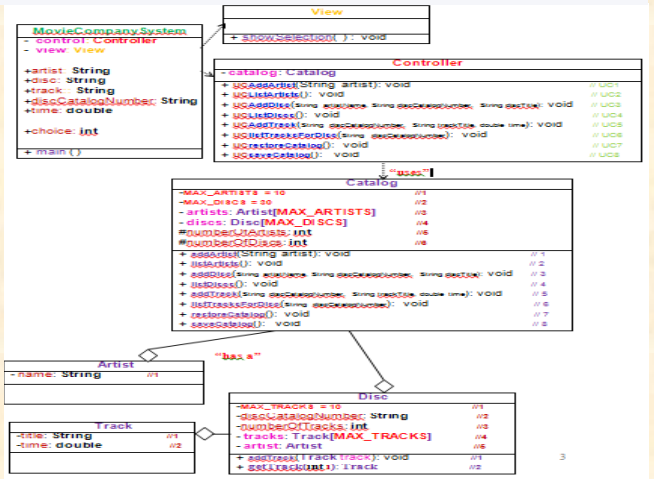
Get Help

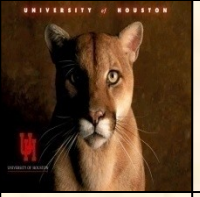
support@openai.com

Global rate limit exceeded

TRACKING ID: d8f9495fd6ee1f4c95d5





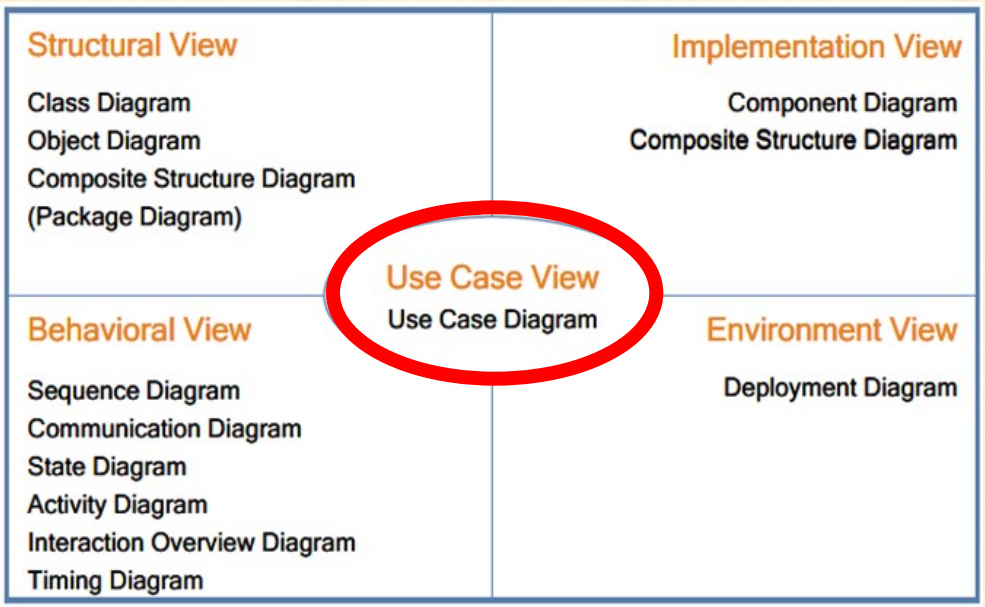


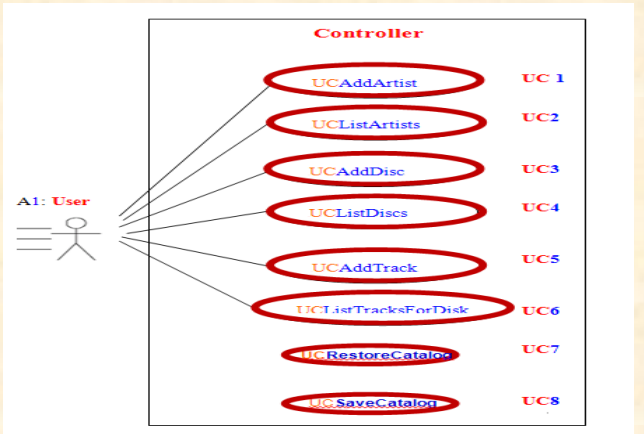
Unified Modeling Language

- **UML** provides structure for problem solving.
- A **method** for **M**odeling **O**bject **O**riented **programs**.
  - A means for visualizing, constructing and documenting **software projects**.
  - Promotes **component Reusability**.

You **will not understand what your algorithm** does **two weeks after you wrote it**. But if you have the **Model**, you will **catch up fast and easily**.

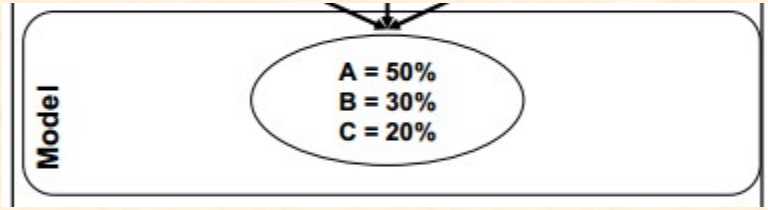
- Teamwork for parallel development of large **systems**.
- **UML** includes 13 **Diagrams** (**Models**).

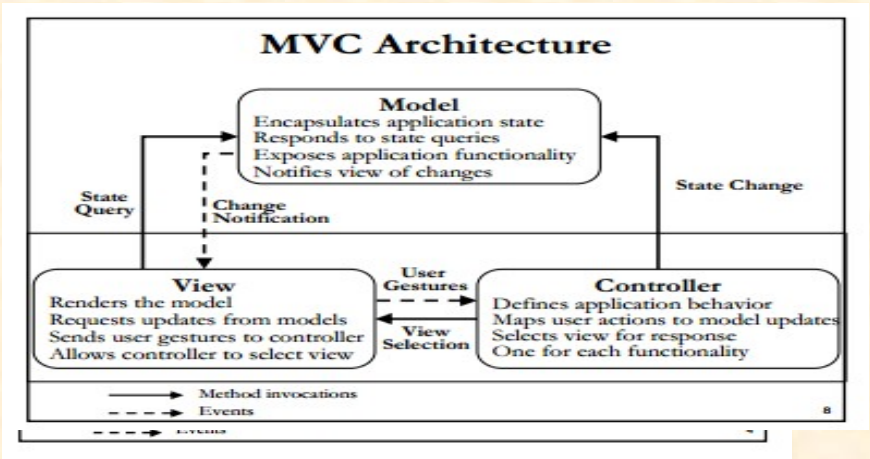


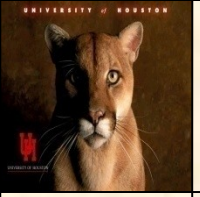


MVC Model View Controller

- **Design Pattern** that separates:
  - **Presentation** Logic – **View Class**
  - Business **Model** – **Model Class**
  - **Business** Logic – **Controller Class**

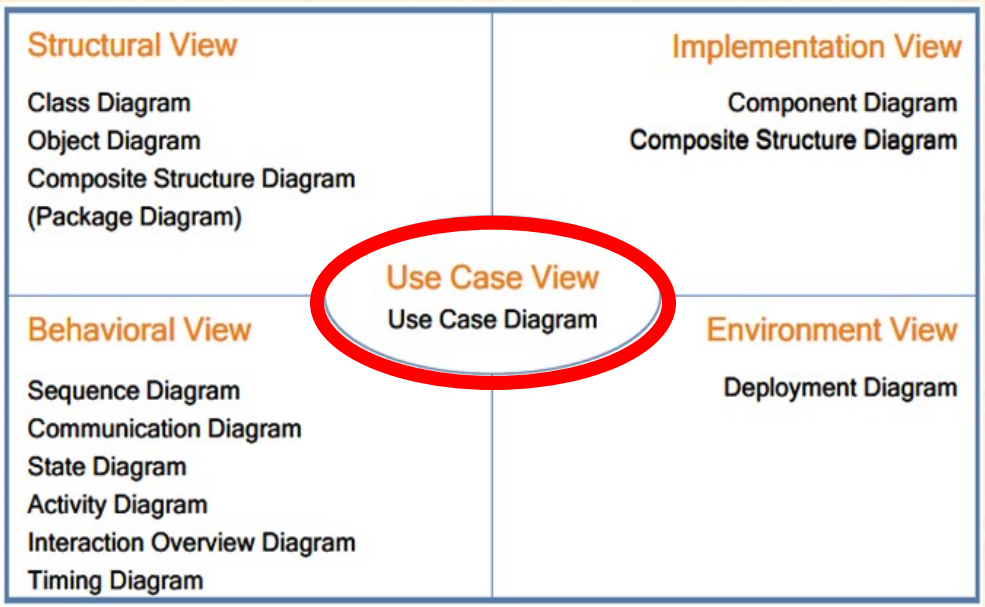


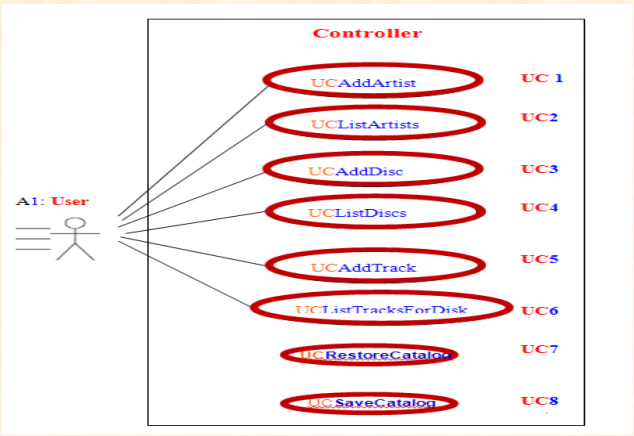


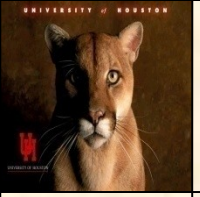


Unified Modeling Language

- UML provides structure for problem solving.







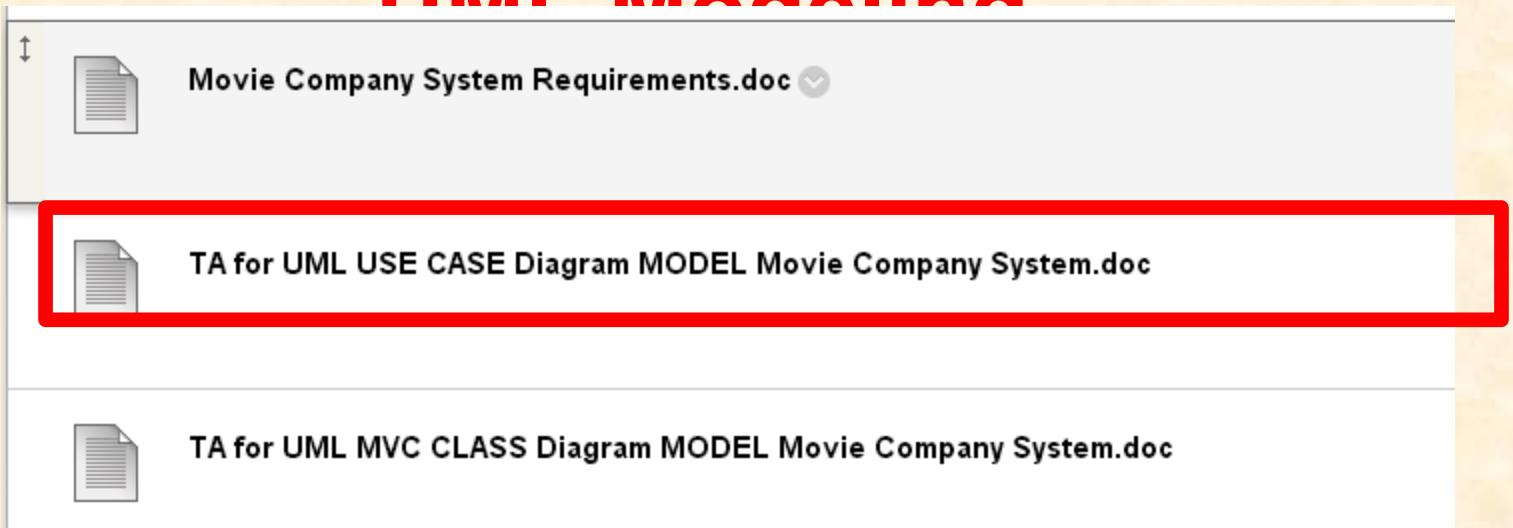
Movie Company System

Case Study

Requirements

Domain/Textual Analysis

UML Modeling



Movie Company System

Requirements

The **Movie Company** wants to keep a **catalog** of its artists and their discs (maximum 30).

Artists have a name. The **Movie Company** can ***add an artist*** or get a ***listing of all its artists***.

Each artist produces music **Disc**s. A Disc is associated with an Artist. A Disc is cataloged with a disc catalogNumber. The **Movie Company** can ***add a disc*** or get a ***listing of all its discs***.

A Disc has a number of **Track**s. The Track have a **title** and a duration **time**. A Disc has a catalogNumber , a title, the number of tracks. The **Movie Company** can ***add a track*** or get a ***listing of all tracks for a given disc***.

Movie Company System

Requirements

The **Movie Company** wants to keep a **catalog** of its artists and their discs (maximum 30).

Artists have a name. The **Movie Company** can ***add an artist*** or get a ***listing of all its artists***.

Each artist produces music **Disc**s. A Disc is associated with an Artist. A Disc is cataloged with a disc catalogNumber. The **Movie Company** can ***add a disc*** or get a ***listing of all its discs***.

A Disc has a number of **Track**s. The Track have a **title** and a duration **time**. A Disc has a catalogNumber , a title, the number of tracks. The **Movie Company** can ***add a track*** or get a ***listing of all tracks for a given disc***.

Movie Company System

Requirements

SAMPLE

The **company** has the following affiliated Artists (maximum 10):

"Miles Davis", with 3 Discs:

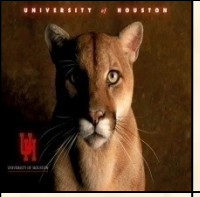
Disc 1: **Title** "Miles Ahead" with Track "Miles Ahead" that is 3.34 minutes

Disc 2: Title "Tutu with Tracks: "Tutu", 4.56, "Portia", 3.34, "Tomaas", 3.56

Disc 3: Title "Kind of Blue" with Tracks "So What", 4.54, "Freddie Freeloader, 5.25, "Blue in Green", 2.56

"John Coltrane", with 1 Disc:

Disc 1: Title "Blue Train" with Blue Train" that is 5.15 minutes; "Moments Notice", 5.52, "Locomotion", 6.18



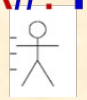
UML USE CASE MODELING

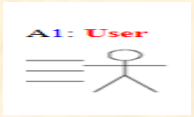
Requirements

Domain/ Textual Analysis for First UML Use Cases Diagram + Use Cases Description

Step 1: Identify Actors

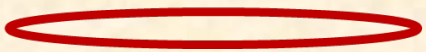
A#: **Role**





Step 2: Identify Use Cases

UC#: **Verb + Attribute**



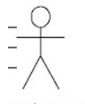
Step 3: Draw UML Use Cases Diagram

CUT & PASTE

REARANGE

Step 4: UML Use Cases Description

A1: User



Movie Company System

UML USE CASE MODEL

The **Movie Company** wants to keep a **catalog** of its artists and their discs (maximum 30).

Artists have a name. The **Movie Company** can ***add an artist*** or get a ***listing of all its artists***.

Each artist produces music **Disc**s. A Disc is associated with an Artist. A Disc is cataloged with a disc catalogNumber. The **Movie Company** can ***add a disc*** or get a ***listing of all its discs***.

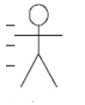
A Disc has a number of **Track**s. The Track have a **title** and a duration **time**. A Disc has a catalogNumber , a title, the number of tracks. The **Movie Company** can ***add a track*** or get a ***listing of all tracks for a given disc***.

Step 1: Identify Actors

Movie Company System

UML USE CASE MODEL

A1: User



The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

UC1: UCAddArtist

UC2: UCListArtists

Artists have a name. The Movie Company can add an artist or get a listing of all its artists

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

UC3: UCAddDisc

UC4: UCListDiscs

disc catalogNumber. The Movie Company can add a disc or get a listing of all its discs

A Disc has a number of Tracks. The Track have a title and a duration time. A Disc has a

UC5: UCAddTrack

catalogNumber , a title, the number of tracks. The Movie Company can add a track or get a

UC6: UCListTracksForDiscs

UC7: UCrestoreCatalog

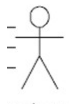
UC8: UCsaveCatalog

listing of all tracks for a given disc

Step 2: Identify Use Cases

CUT & PASTE

A1: User



Movie Company System

UML USE CASE MODEL - WHAT

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

UC1: UCAddArtist

UC2: UCListArtists

Artists have a name. The Movie Company can add an artist or get a listing of all its artists

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

UC3: UCAddDisc

UC4: UCListDiscs

disc catalogNumber. The Movie Company can add a disc or get a listing of all its discs

A Disc has a number of Tracks. The Track have a title and a duration time. A Disc has a

UC5: UCAddTrack

catalogNumber , a title, the number of tracks. The Movie Company can add a track or get a

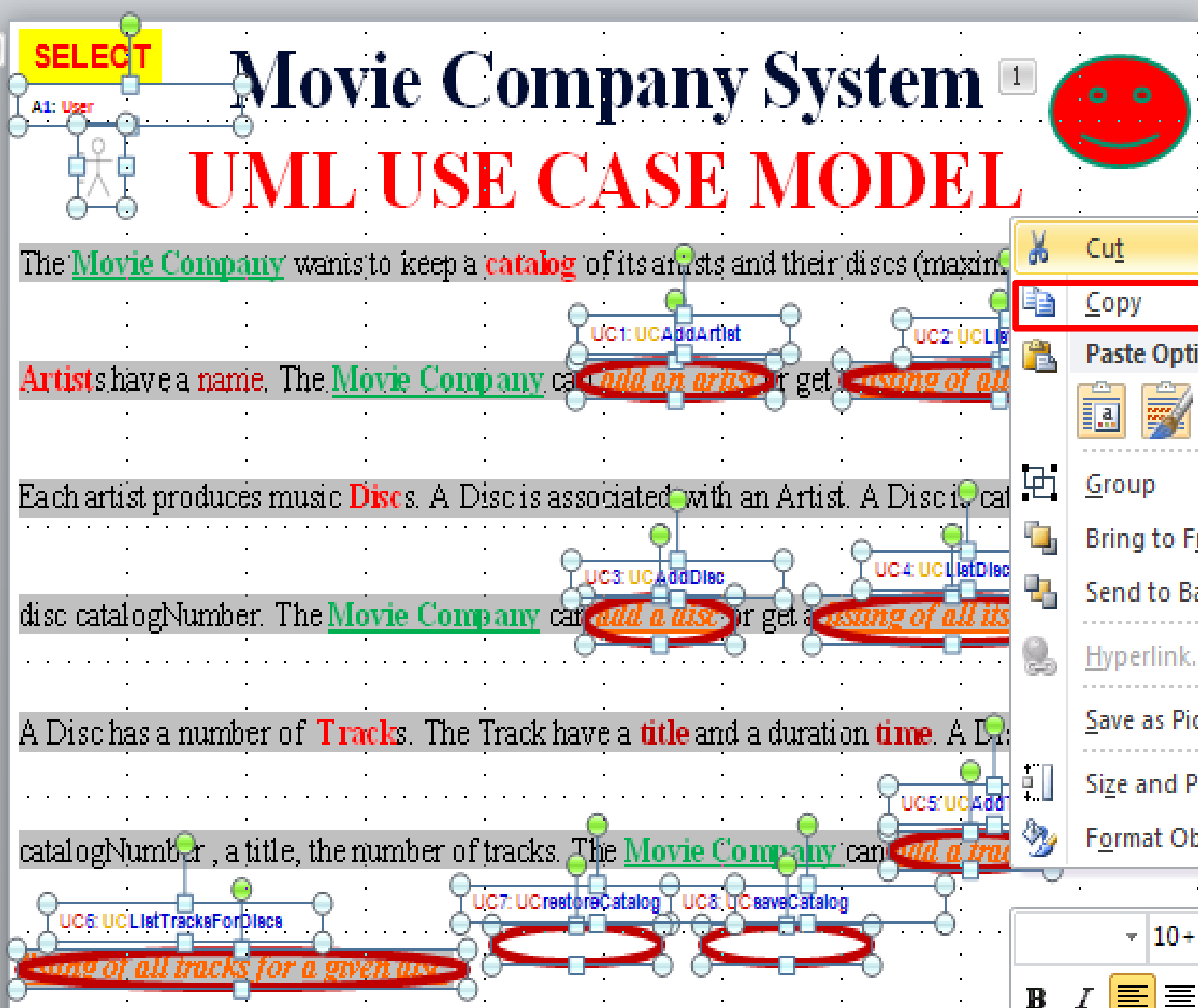
UC6: UCListTracksForDiscs

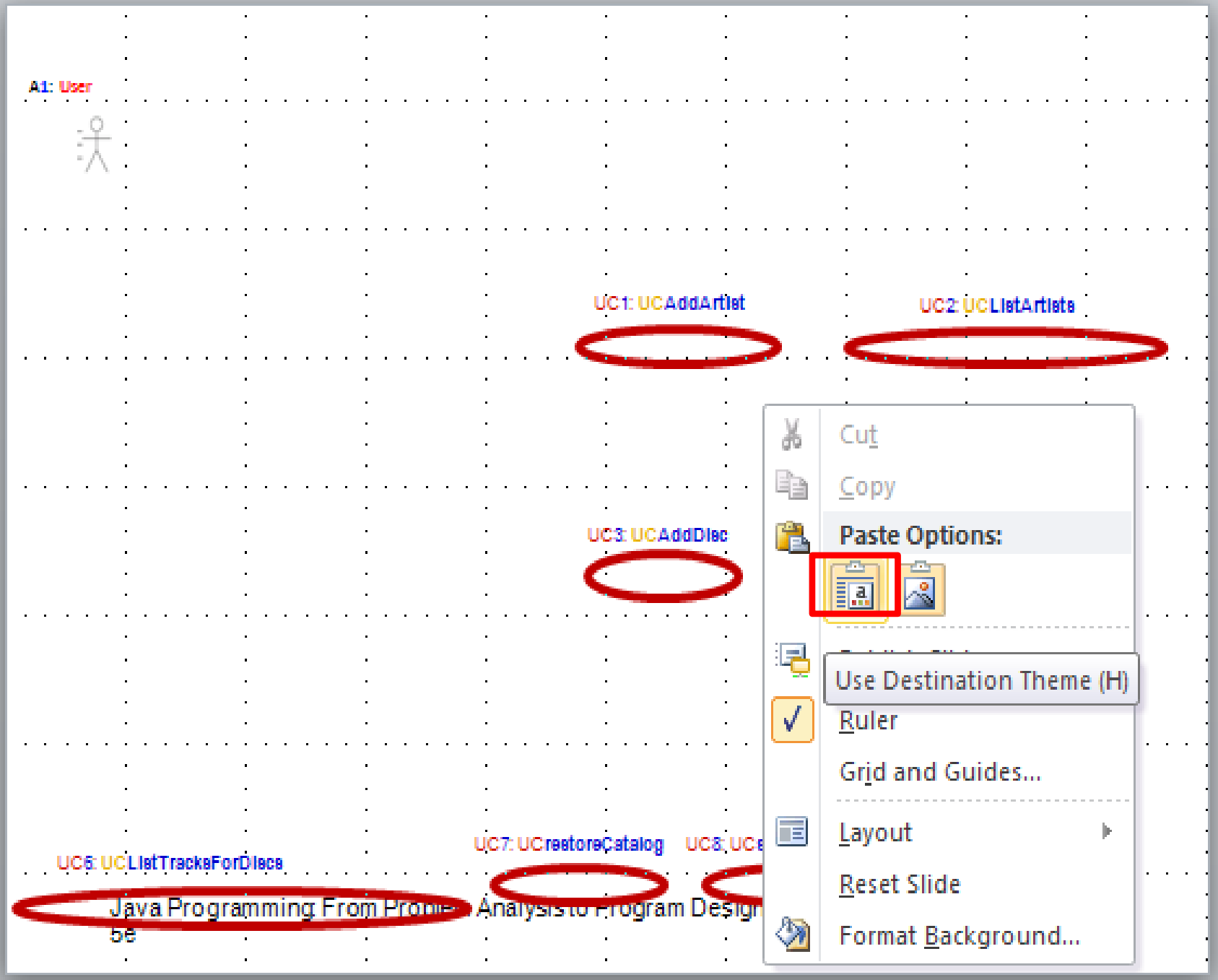
UC7: restoreCatalog

UC8: saveCatalog

listing of all tracks for a given disc

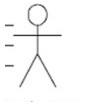
Step 3: Draw UML Use Cases Diagram


SELECT
A1: User
Movie Company System
1
UML USE CASE MODEL
The Movie Company wants to keep a catalog of its artists and their discs (maxim
UC1: UCAddArtist
UC2: UCLis
Artists have a name. The Movie Company ca
r get
Each artist produces music Discs. A Disc is associated with an Artist. A Disc is ca
UC3: UCAddDisc
UC4: UCListDisc
disc catalogNumber. The Movie Company ca
r get a
A Disc has a number of Tracks. The Track have a title and a duration time. A Di
UC5: UCAdd
catalogNumber , a title, the number of tracks. The Movie Company can
UC6: UCListTracksForDiscs
UC7: UCrestoreCatalog
UC8: UCsaveCatalog
Cut
Copy
Paste Options:
Group
Bring to Front
Send to Back
Hyperlink...
Save as Picture...
Size and Position...
Format Object...
10+


A1: User
UC1: UCAddArtist
UC2: UCListArtists
UC3: UCAddDisc
UC6: UCListTracksForDiscs
UC7: UCrestoreCatalog
UC8: UCe
Java Programming From Proble
Analysis to Program Design
5e
Cut
Copy
Paste Options:
Use Destination Theme (H)
Ruler
Grid and Guides...
Layout
Reset Slide
Format Background...

Your deliverable will contain Steps 1 and 2 on one page like this

A1: User



CUT & PASTE

REARANGE

The **Movie Company** wants to keep a **catalog** of its artists and their discs (maximum 30).

UC1: **UCAddArtist**

UC2: **UCListArtists**

Artists have a name. The **Movie Company** can *add an artist* or get a *listing of all its artists*

Each artist produces music **Disc**s. A Disc is associated with an Artist. A Disc is cataloged with a

UC3: **UCAddDisc**

UC4: **UCListDiscs**

disc catalogNumber. The **Movie Company** can *add a disc* or get a *listing of all its discs*

A Disc has a number of **Track**s. The Track have a **title** and a duration **time**. A Disc has a

UC5: **UCAddTrack**

catalogNumber , a title, the number of tracks. The **Movie Company** can *add a track* or get a

UC6: **UCListTracksForDiscs**

UC7: **restoreCatalog**

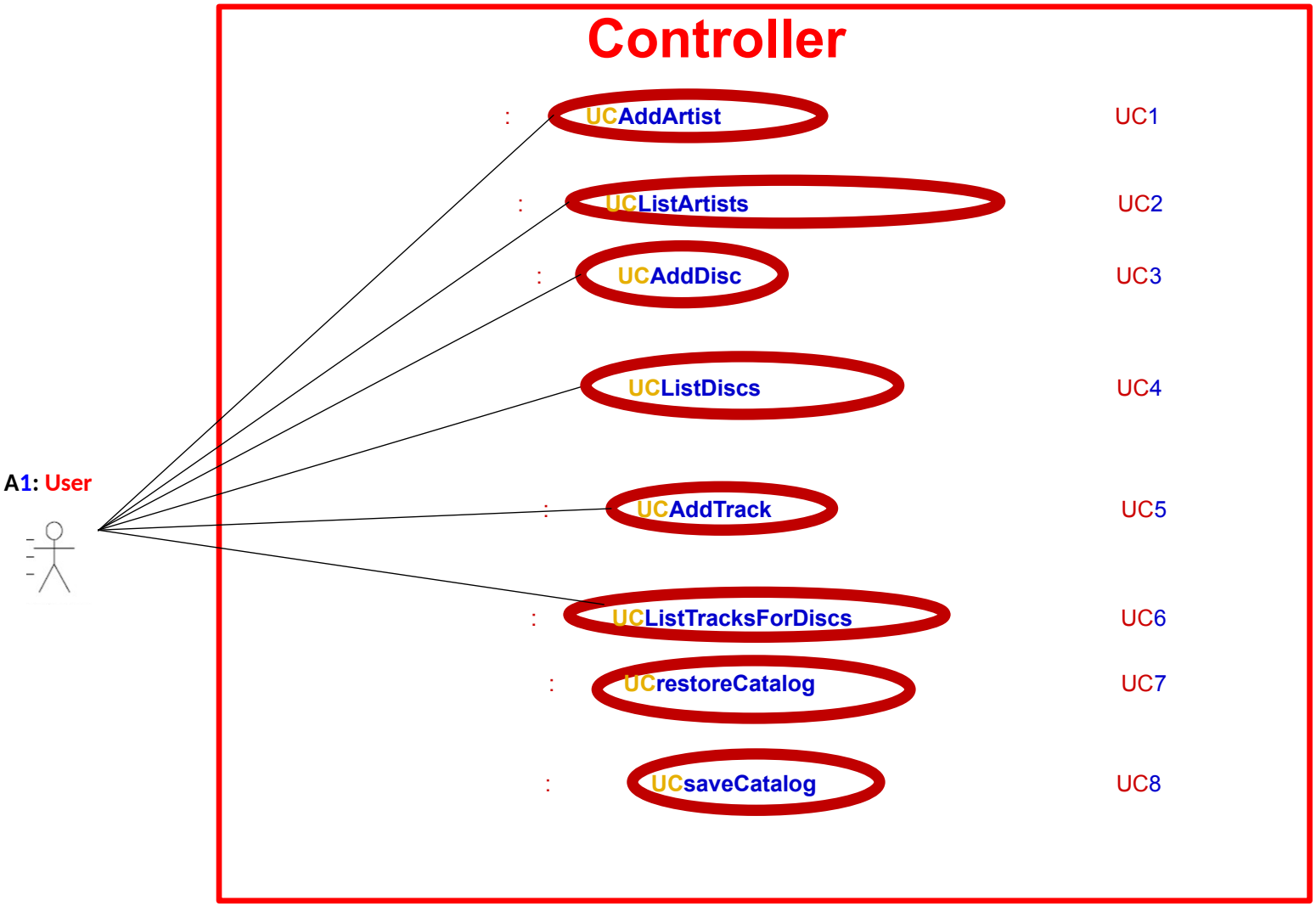
UC8: **saveCatalog**

listing of all tracks for a given disc

Movie Company System

REARRANGE

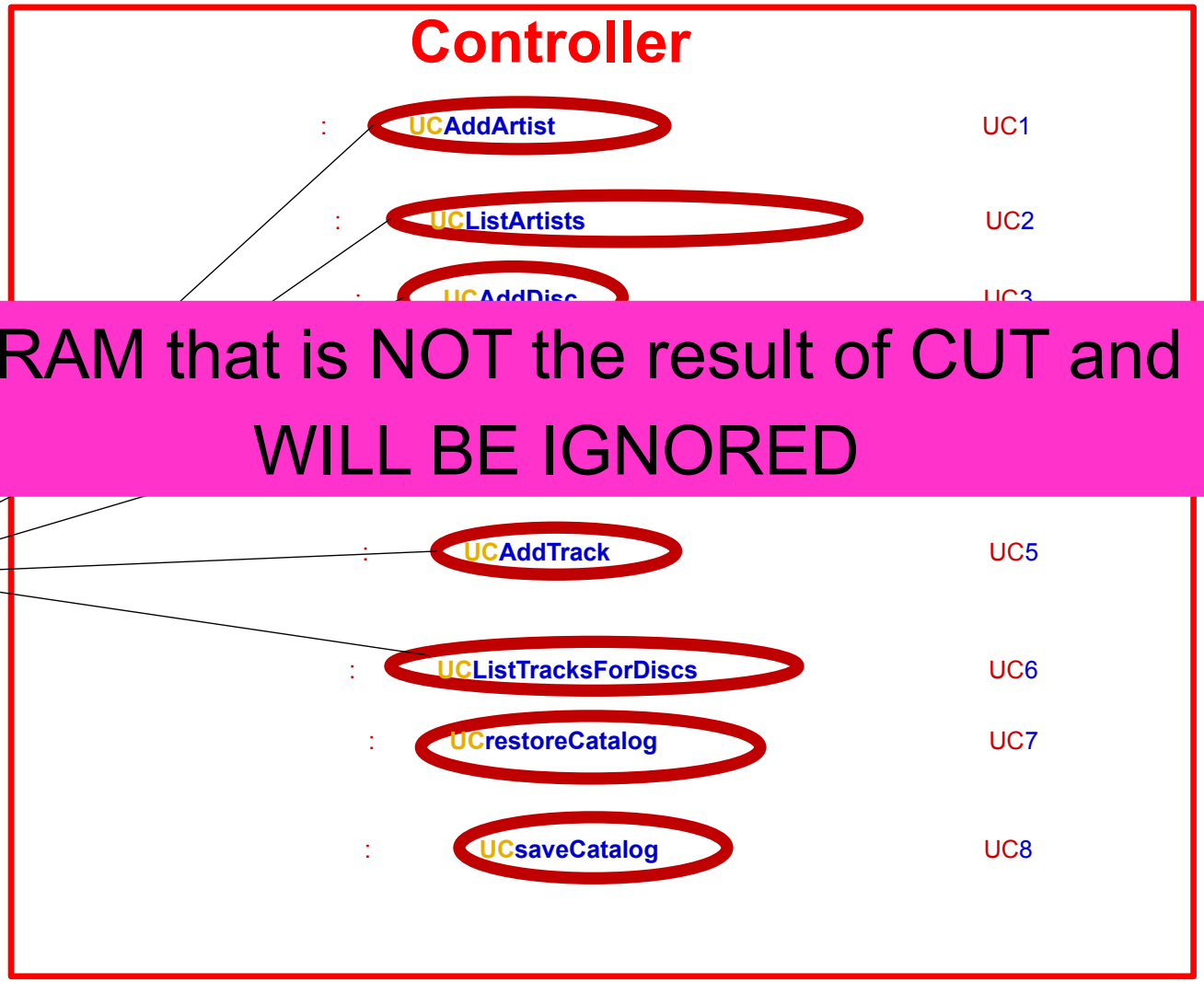
UML USE CASE MODEL



Step 3: Draw UML Use Cases Diagram

Movie Company System

UML USE CASE MODEL



Any DIAGRAM that is NOT the result of CUT and PASTE WILL BE IGNORED

Step 3: Draw UML Use Cases Diagram

Movie Company System

UML USE CASE Description

UC1 UCAddArtist Description

Name:	**UCAddArtist**
Actor:	**User**
Description:	**This use case describes the process used by User to Add another Artist**
Successful Completion:	**User requests Adding Artist by artistName** **1. Movie Company System checks to see if not already there and there is room in the artists array** **2. Artist with artistName is added at the end of the artists array and successful message is sent to the User**
Alternative:	**User requests Adding Artist by artistName** **1. Movie Company System checks to see if not already in the artists array and there is no room** **2. If artistName either ALREADY present or NO MORE ROOM MAX_ARTISTS Artist with artistName is NOT added and UNsuccessful message is sent to the User**
Pre-Condition:	**User requests Add Artist**
Post-Condition:	**Artist with artistName Added successfully or UNsuccessfully**
Assumptions:	**None**

Movie Company System

UML USE CASE Description

UC2 UCListArtists Description

Name:	**UCListArtists**
Actor:	**User**
Description:	**This use case describes the process used by User to List all Artists**
Successful Completion:	**User requests Listing Artists** **1. Movie Company System displays all the Artists by traversing the artists array**
Alternative:	
Pre-Condition:	**User requests List Artists**
Post-Condition:	**Artists displayed**
Assumptions:	**None**

Movie Company System

UML USE CASE Description

UC3 UCAddDisc Description

Name:	**UCAddDisc**
Actor:	**User**
Description:	**This use case describes the process used by User to Add another Disc for a particular artistName**
Successful Completion:	**User requests Adding Disc to artistName** **1. Movie Company System prompts for discCatalogNumber** **2. Movie Company System prompts for disc** **3. Disc discCatalogNumber , disc is Added to the disks array and successful message is sent to the User**
Alternative:	**User requests Adding Disc to artistName** **1. Movie Company System prompts for discCatalogNumber** **2. Movie Company System prompts for disc** **3. Movie Company System checks to see if not already the disks array and there is room in the disks array** **4. If disc either ALREADY present or NO MORE ROOM MAX_DISCS Disc disc is NOT added and UNsuccessful message is sent to the User**
Pre-Condition:	**User requests Add Disc**
Post-Condition:	**Disc discCatalogNumber, disc Added to the disks array successfully or UNsuccessfully**
Assumptions:	**None**

Movie Company System

UML USE CASE Description

UC4 UCListDiscs Description

Name:	**UCListDiscs**
Actor:	**User**
Description:	**This use case describes the process used by User to List all Discs**
Successful Completion:	**User requests Listing Discs** **1. Movie Company System displays all the Discs in the disks array, discCatalogNumber, title, artistName**
Alternative:	
Pre-Condition:	**User requests List Discs**
Post-Condition:	**Discs displayed**
Assumptions:	**None**

Movie Company System

UML USE CASE Description

UC5 UCAddTrack Description

Name:	UCAddTrack
Actor:	User
Description:	This use case describes the process used by User to Add another Track trackTitle, time for a particular discCatalogNumber
Successful Completion:	User requests Adding Track to Disc discCatalogNumber 1. Movie Company System prompts for trackTitle 2. Movie Company System prompts for time 3. Track trackTitle, time is Added to Disc discCatalogNumber and successful message is sent to the User
Alternative:	User requests Adding Track to Disc discCatalogNumber 1. Movie Company System prompts for trackTitle 2. Movie Company System prompts for time 3. Movie Company System checks to see if not already there and there is room 4. If trackTitle either ALREADY present or NO MORE ROOM MAX_TRACKS Track trackTitle is NOT added and UNsuccessful message is sent to the User
Pre-Condition:	User requests Add Track
Post-Condition:	Track trackTitle, time Added to Disc discCatalogNumber successfully or UNsuccessfully
Assumptions:	None

Step 4: UML Use Cases Description

Movie Company System

UML USE CASE Description

UC6 UCListTracksForDisc Description

Name:	**UCListTracksForDisc**
Actor:	**User**
Description:	**This use case describes the process used by User to List all Tracks title for a Disc discCatalogNumber**
Successful Completion:	**User requests Listing Discs** **1. Movie Company System displays all the Tracks title for Disc discCatalogNumber**
Alternative:	
Pre-Condition:	**User requests List Tracks for Disc**
Post-Condition:	**Tracks for Disc displayed**
Assumptions:	**None**

Movie Company System

UML USE CASE Description

UC7 UCRestoreCatalog Description

Name:	**UCRestoreCatalog**
Actor:	**None**
Description:	**This use case describes the process of reading the catalog from a data file (Input). It is called automatically when the Movie Company System starts running**
Successful Completion:	**1. Opens "Catalog.txt" file and reads the catalog**
Alternative:	
Pre-Condition:	**None**
Post-Condition:	**The catalog resides in the Movie Company System memory.**
Assumptions:	**File "Catalog.txt" was read successfully**

Movie Company System

UML USE CASE Description

UC8 UCSaveCatalog Description

Name:	**UCSaveCatalog**
Actor:	**None**
Description:	**This use case describes the process of writing the catalog from Movie Company System memory to data file "Catalog.txt" (Output). It is called automatically before the Movie Company System exits**
Successful Completion:	**1. Opens "Catalog.txt" file and write the catalog**
Alternative:	
Pre-Condition:	**The catalog resides in the Movie Company System memory.**
Post-Condition:	**The catalog resides in the "Catalog.txt" file.**
Assumptions:	**File "Catalog.txt" was**

Completed

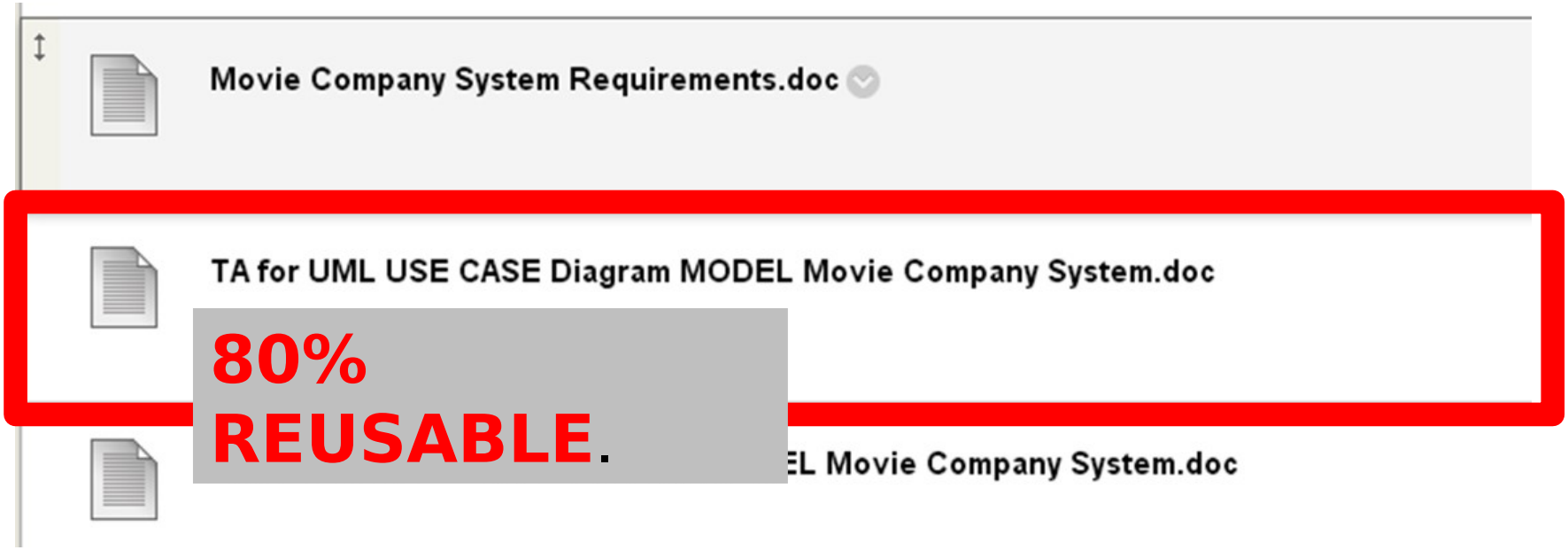
UML USE CASE MODEL

Movie Company System

UML USE CASE MODEL

Please use TEMPLATE

“TA for UML USE CASE Diagram MODEL Movie Company System.doc “



Movie Company System

UML USE CASE MODEL

Complete example.

TA for UML USE CASE Diagram MODEL Movie Company System.doc

TA (Textual Analysis) for UML USE CASE Diagram MODEL for Movie Company System

A1: User

The Movie Company has the following affiliated Artists (maximum 10):

"Miles Davis", with 3 Discs:

Disc 1: Title "Miles Ahead" with Track "Miles Ahead" that is 3.34 minutes

Disc 2: Title "Tutu with Tracks: "Tutu", 4.56, "Portia", 3.34, "Tomaas", 3.56

Disc 3: Title "Kind of Blue" with Tracks "So What", 4.54, "Freddie Freeloader, 5.25, "Blue in Green", 2.56

"John Coltrane", with 1 Disc:

Disc 1: Title "Blue Train" with Blue Train" that is 5.15 minutes; "Moments Notice", 5.52, "Locomotion", 6.18

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30). Artists have a name. The Movie Company can add an artist or get a listing of all its artists.

UC1: UCAddArtist UC2: UCListArtists

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a disc catalogNumber. The Movie Company can add a disc or get a listing of all its discs.

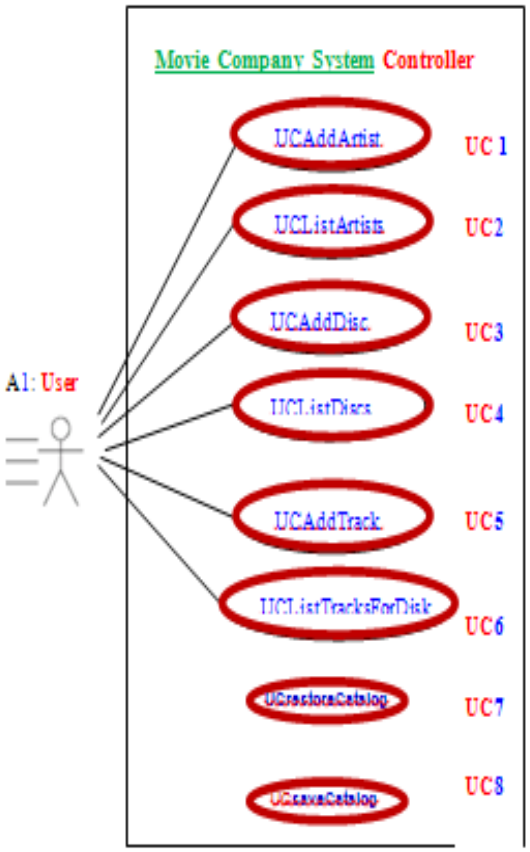
UC3: UCAddDisc UC4: UCListDiscs

A Disc has a number of Tracks. The Track has a title and a duration time. A Disc has a catalogNumber, a title, the number of tracks. The Movie Company can add a track or get a listing of all tracks for a given disc.

UC5: UCAddTrack UC6: UCListTracksForDisc

UC7: UCrestoreCatalog UC8: UCsaveCatalog

All steps on one Requirements document!

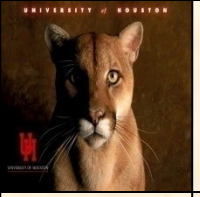


UC1 UCAddArtist Description

Name:	UCAddArtist
Actor:	User
Description:	This use case describes the process used by User to Add another Artist
Successful Completion:	User requests Adding Artist by artistName 1. Movie Company System checks to see if not already there and there is room 2. Artist artistName is added and successful message is sent to the User
Alternative:	User requests Adding Artist by artistName 1. Movie Company System checks to see if not already there and there is room 2. If artistName either ALREADY present or NO MORE ROOM MAX_ARTISTS Artist artistName is NOT added and UNsuccessful message is sent to the User
Pre-Condition:	User requests Add Artist
Post-Condition:	Artist artistName Added successfully or UNsuccessfully
Assumptions:	None

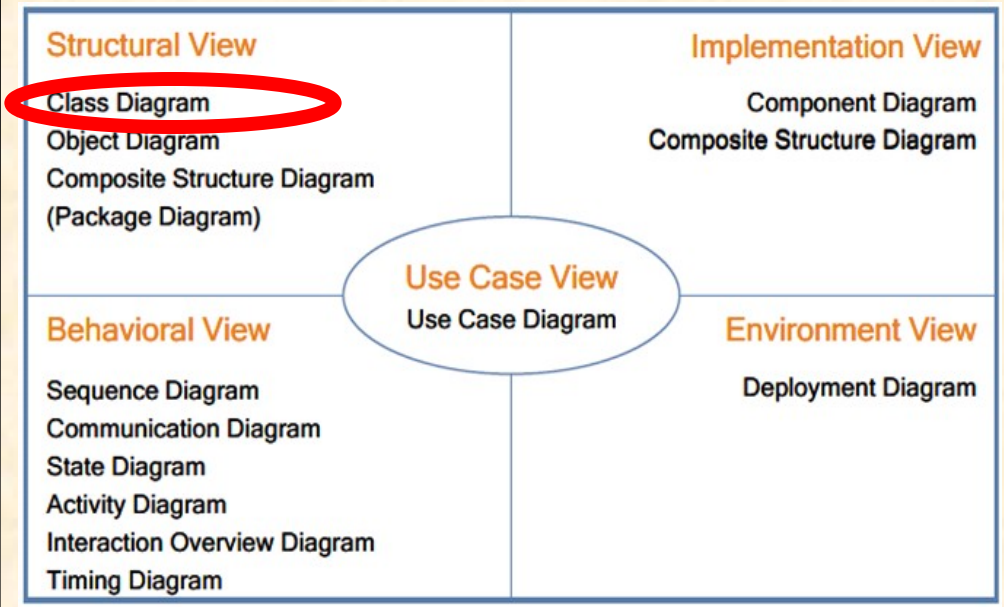
40% of the total effort

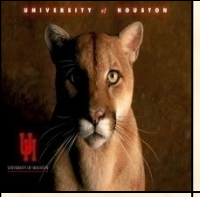
AND SO ON!



Unified Modeling Language

- UML provides structure for problem solving.





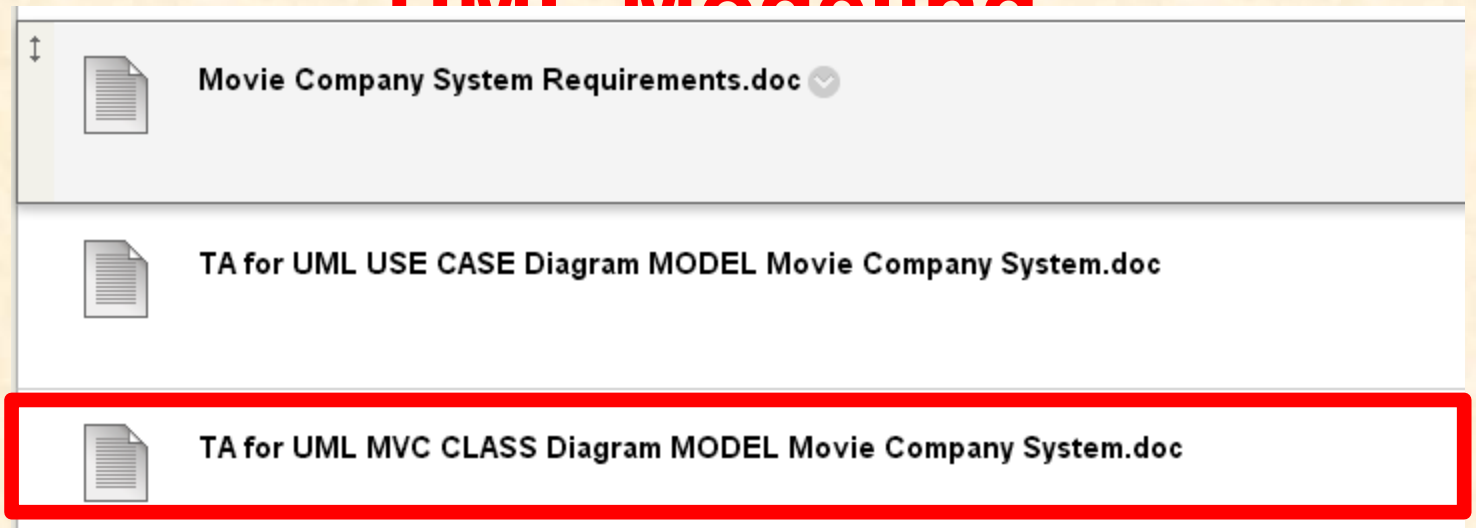
Movie Company System

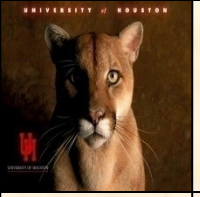
Case Study

Requirements

Domain/Textual Analysis

UML Modeling





UML MVC CLASS MODELING

Analysis/Design

Domain/Textual Analysis for

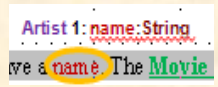
Second UML MODEL: UML Classes Diagram + Methods Pseudocode

Step 1: Identify Classes

Class

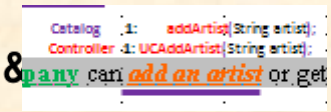
Step 2: Identify Attributes

Class: # - Attribute : Type



Step 3: Identify verbs Methods

Class: # - Method: Verb &



Step 4: Draw UML Classes Diagram

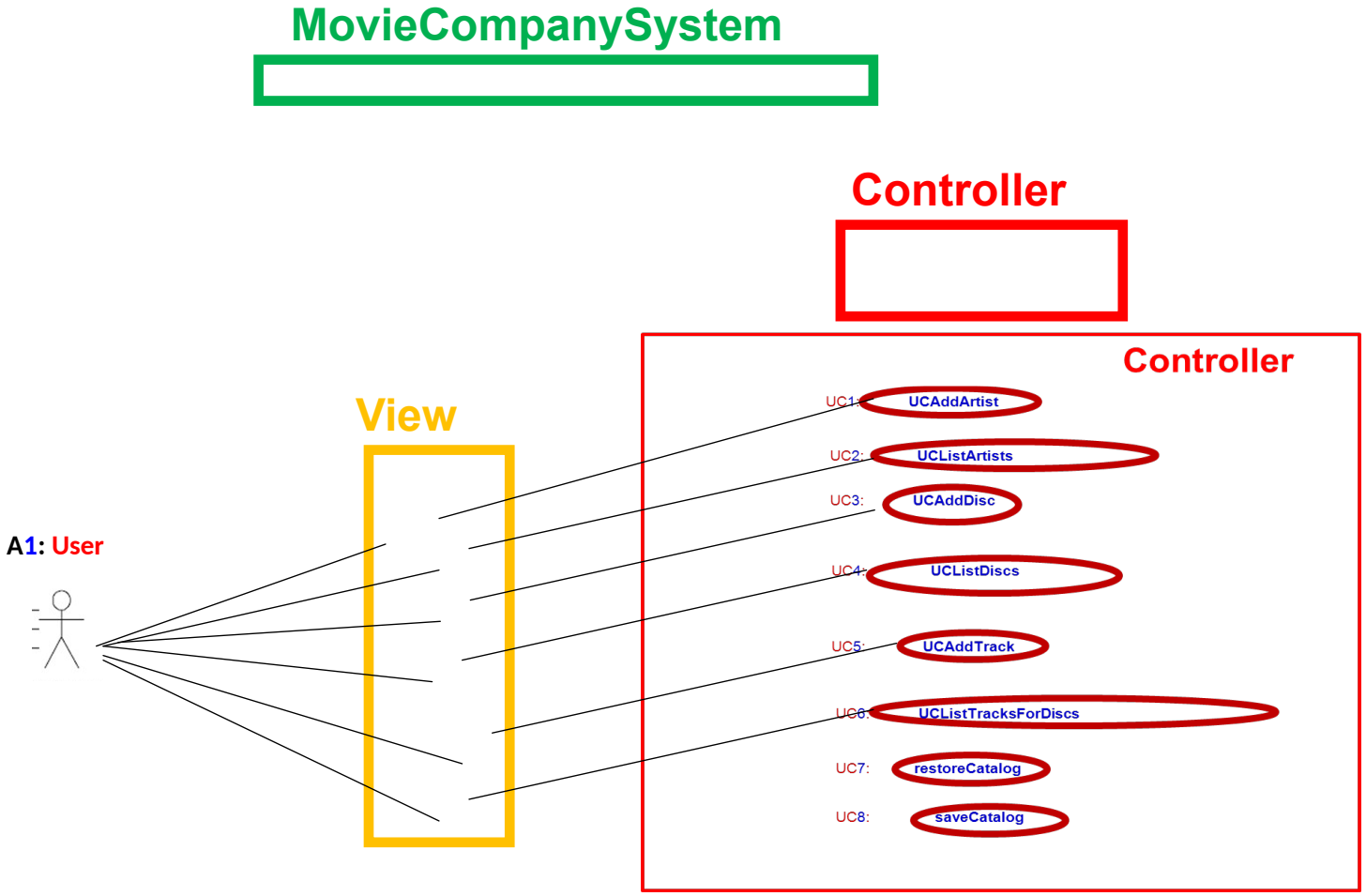
CUT & PASTE

REARANGE

Step 5: Detailed Design Pseudocode for methods

Movie Company System

UML MVC CLASS MODEL



Step 1: Identify Classes

V C

Movie Company System

UML MVC CLASS MODEL

Catalog

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

Artist

Artists have a name. The Movie Company can *add an artist* or get a *listing of all its artists*.

Disc

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

disc catalogNumber. The Movie Company can *add a disc* or get a *listing of all its discs*.

Track

A Disc has a number of Tracks. The Track have a **title** and a duration **time**. A Disc has a

catalogNumber , a title, the number of tracks. The Movie Company can *add a track* or get a

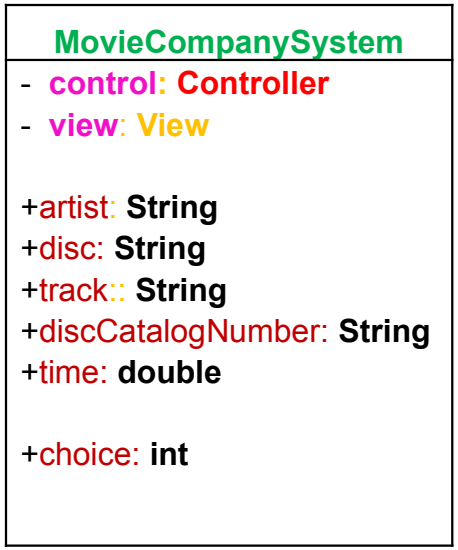
listing of all tracks for a given disc.

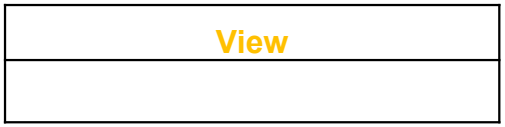
Step 1: Identify Classes

M

Movie Company System

UML MVC CLASS MODEL





Step 2: Identify Attributes V C

Movie Company System

UML MVC CLASS MODEL

Catalog 6: numberOfDiscs : int
Catalog 5: numberOfArtists : int
Catalog 4: Disc[MAX_DISCS] discs
Catalog 3: Artist[MAX_ARTISTS] artists
Catalog 2: MAX_DISCS = 30
Catalog 1: MAX_ARTISTS = 10

Catalog

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

Artist

Artist 1: name : String

Artists have a name. The Movie Company can *add an artist* or get a *listing of all its artists*.

Disc

Disc 5: Artist artist

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

Disc 1: MAX_TRACKS = 10

Disc 2: discCatalogNumber : String

disc catalogNumber. The Movie Company can *add a disc* or get a *listing of all its discs*.

Disc 3: numberOfTracks : int

Track

Track 1: title : String

Track 2: time : double

A Disc has a number of Tracks. The Track have a title and a duration time. A Disc has a

Disc 4: Track[MAX_TRACKS] tracks

catalogNumber , a title, the number of tracks. The Movie Company can *add a track* or get a

listing of all tracks for a given disc.

Step 2: Identify Attributes

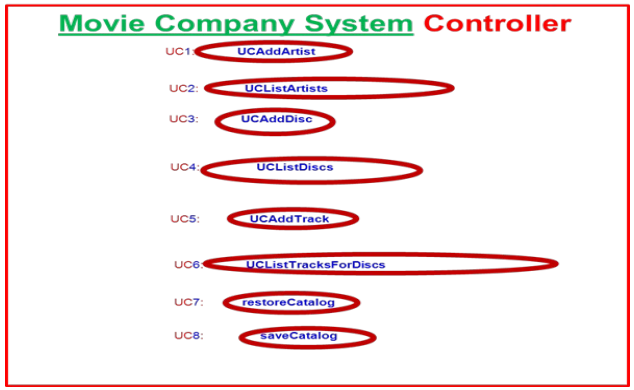
M

Movie Company System

UML MVC CLASS MODEL

MovieCompanySystem
- control: Controller - view: View +artist: String +disc: String +track:: String +discCatalogNumber: String +time: double +choice: int
+ main ( )

View
+ showSelection( ) : void



Controller	
- catalog: Catalog	
+ UCAddArtist(String artist): void	// UC1
+ UCListArtists(): void	// UC2
+ UCAddDisc(String artistName, String discCatalogNumber, String discTitle): void	// UC3
+ UCListDiscs(): void	// UC4
+ UCAddTrack(String discCatalogNumber, String trackTitle, double time): void	// UC5
+ UClistTracksForDisc(String discCatalogNumber): void	// UC6
+ UCrestoreCatalog(): void	// UC7
+ UCsaveCatalog(): void	// UC8

Step 3: Identify verbs Methods V C

Movie Company System

UML MVC CLASS MODEL

Catalog

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

Catalog 1: addArtist(String artist);
Controller 1: UCAddArtist(String artist);

Catalog 2: listArtists();
Controller 2: UCListArtists();

Artist

Artists have a name. The Movie Company can *add an artist* or get a *listing of all its artists*.

Disc

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

Catalog 3: addDisc(String artist, String discCatalogNumber, String disc);
Controller 3: UCAddDisc(String artist, String discCatalogNumber, String disc);

disc catalogNumber. The Movie Company can *add a disc* or get a *listing of all its discs*.

Controller 4: UCListDiscs();
Catalog 4: listDiscs();

Track

A Disc has a number of Tracks. The Track have a **title** and a duration **time**. A Disc has a

Catalog 5: addTrack(String discCatalogNumber, String track, double time);
Controller 5: UCAddTrack(String discCatalogNumber, String track, double time);

catalogNumber , a title, the number of tracks. The Movie Company can *add a track* or get a

Controller 7: UCrestoreCatalog();
Catalog 7: restoreCatalog();

Controller 8: UCsaveCatalog();
Catalog 8: saveCatalog();

listing of all tracks for a given disc.

Controller 6: UCListTracksForDisc(String discCatalogNumber);
Catalog 6: listTracksForDisc(String discCatalogNumber);

Step 3: Identify verbs Methods

M

Movie Company System

UML MVC CLASS MODEL

Catalog 6: numberOfDiscs
Catalog 5: numberOfArtists

CUT & PASTE

Catalog 4: Disc[MAX_DISCS] discs
Catalog 3: Artist[MAX_ARTISTS] artists

Catalog 2: MAX_DISCS = 30

Catalog 1: MAX_ARTISTS = 10

Catalog

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

Artist

Artist 1: name

Artists have a name. The Movie Company can *add an artist* or get a *listing of all its artists*.

Disc

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

Disc 1: MAX_TRACKS = 10

Disc 2: discCatalogNumber

disc catalogNumber. The Movie Company can *add a disc* or get a *listing of all its discs*.

Track

Disc 3: numberOfTracks

Track 1: title

Track 2: time

A Disc has a number of Tracks. The Track have a title and a duration time. A Disc has a

Disc 4: Track[MAX_TRACKS] tracks

Disc 5: Artist artist

catalogNumber , a title, the number of tracks. The Movie Company can *add a track* or get a

listing of all tracks for a given disc.

Step 4: Draw UML Classes Diagram

M

Movie Company System

CUT & PASTE

UML MVC CLASS MODEL

Catalog

The Movie Company wants to keep a catalog of its artists and their discs (maximum 30).

Catalog 1: addArtist(String artist);
Controller 1: UCAddArtist(String artist);

Catalog 2: listArtists();
Controller 2: UCListArtists();

Artist

Artists have a name. The Movie Company can *add an artist* or get a *listing of all its artists*.

Disc

Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged with a

Catalog 3: addDisc(String artist, String discCatalogNumber, String disc);
Controller 3: UCAddDisc(String artist, String discCatalogNumber, String disc);

disc catalogNumber. The Movie Company can *add a disc* or get a *listing of all its discs*.

Controller 4: UCListDiscs();
Catalog 4: listDiscs();

Track

A Disc has a number of Tracks. The Track have a **title** and a duration **time**. A Disc has a

Catalog 5: addTrack(String discCatalogNumber, String track, double time);
Controller 5: UCAddTrack(String discCatalogNumber, String track, double time);

catalogNumber , a title, the number of tracks. The Movie Company can *add a track* or get a

Controller 8: UCsaveCatalog();
Catalog 8: saveCatalog();

Controller 7: UCrestoreCatalog();
Catalog 7: restoreCatalog();

listing of all tracks for a given disc.

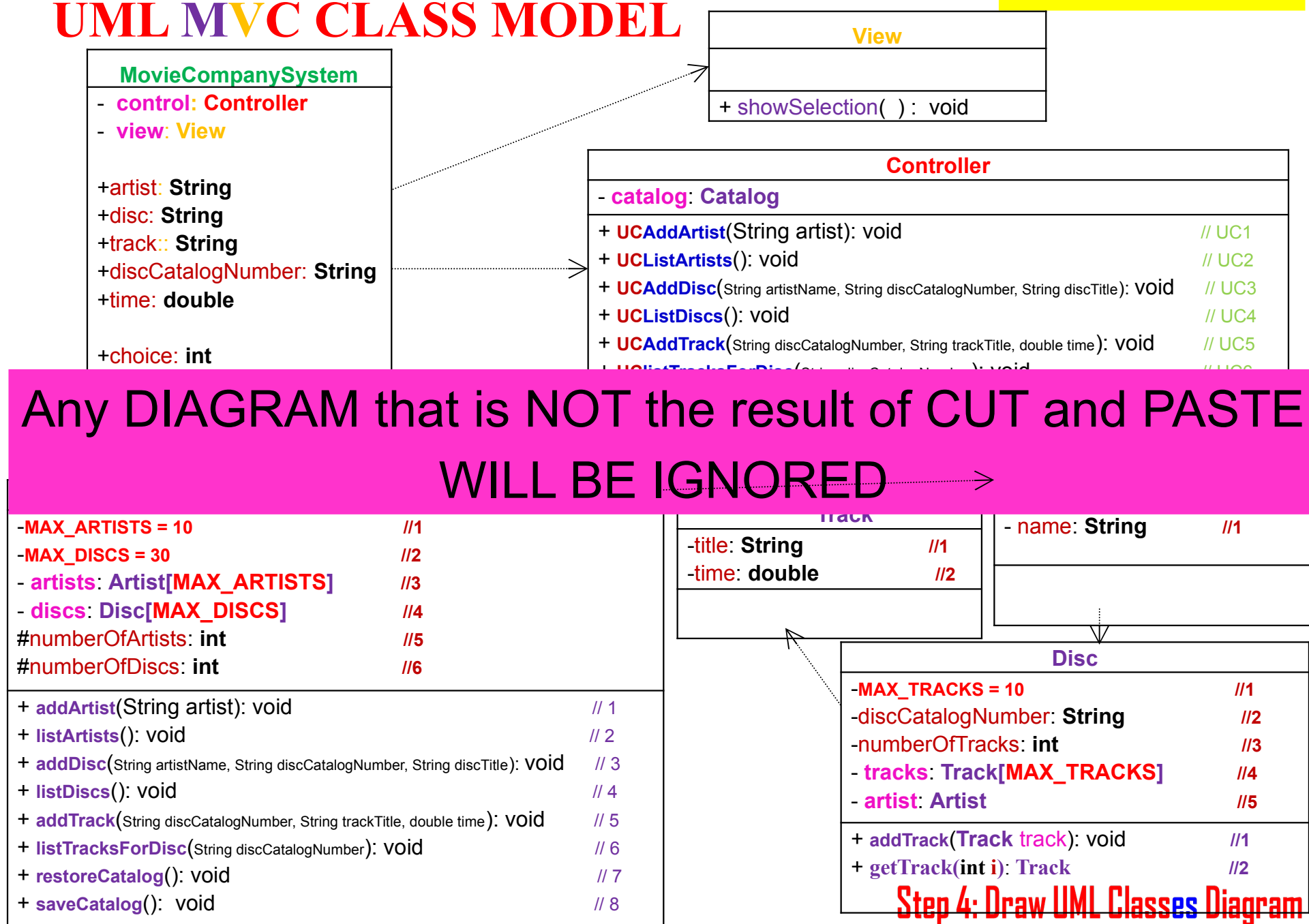
Controller 6: UCListTracksForDisc(String discCatalogNumber);
Catalog 6: listTracksForDisc(String discCatalogNumber);

Step 4: Draw UML Classes Diagram

Movie Company System

UML MVC CLASS MODEL

REARRANGE



Step 4: Draw UML Classes Diagram

Movie Company System

UML MVC CLASS MODEL

CUT & PASTE

7 Classes

4 Model Classes

1 Controller Class

1 View Class

1 Appl Class

DATABASE back end TABLES will become your MODELS in the front end.

Step 4: Draw UML Classes Diagram

Movie Company System

UML MVC CLASS MODEL

7 Classes

Step 5: Detailed Design Pseudocode for methods

Movie Company System

UML MVC CLASS MODEL

MovieCompanySystem
- control: Controller - view: View +artist: String +disc: String +track:: String +discCatalogNumber: String +time: double +choice: int
+ main ( )

public static void main ( ) {
This method will create the **Controller** and **View objects** that allows the Actor **User** to use the **MovieCompanySystem**.

Create the **instance** of the **Controller** class
Controller control = new Controller();

Call the **UC 7 control.UCrestoreCatalog()** of the **Controller** class to read the Catalog from the "**Catalog.txt**" file

Create the **instance** of the **View** class
View view= new View();

Call the **view.showSelection()** of the **View** class to display the menu of options for Actor **User**

Create **local variables** +choice: **int**

+artist: **String**
+disc: **String**
+track:: **String**
+discCatalogNumber: **String**
+time: **double**

Read the choice from the keyboard

Step 5: Detailed Design Pseudocode for methods

Movie Company System

UML MVC CLASS MODEL

MovieCompanySystem
-control: Controller -view: View +choice: int +artist: String +disc: String +track:: String +discCatalogNumber: String +time: double

Case choice of
1: Prompt for Artist Name artist
Call the Controller method **UC1 control.UCAddArtist(artist);**
2: Call the Controller method **UC2 control.UCListArtist();**
3: Prompt for Artist Name artist
Prompt for Disc Catalog Number Name discCatalogNumber
Prompt for Disc Name disc
Call the Controller method **UC3**
control.UCAddDisc(artist,discCatalogNumber,disc**);**
4: Call the Controller method **UC4 control.UCListDiscs();**

5: Prompt for Disc Catalog Number Name discCatalogNumber
Prompt for track Title and Duration trackTitle, time
Call the Controller method **UC5**
control.UCAddTrack(discCatalogNumber,trackTitle,time**);**

6: Prompt for Disc Catalog Number Name discCatalogNumber
Call the Controller method **UC6**
control.UCListTracksForDisc(discCatalogNumber**);**

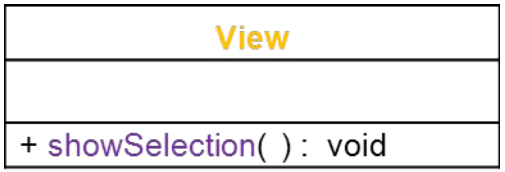
7: Call the **UC 7 control.UCrestoreCatalog()** of the **Controller** class to read the Catalog in the "**Catalog.txt**" file into memory

8: Call the **UC 8 control.UCsaveCatalog()** of the **Controller** class to write the Catalog in the "**Catalog.txt**" file
}

Step 5: Detailed Design Pseudocode for methods

Movie Company System

UML MVC CLASS MODEL




```
public void showSelection(  ) {
    Display the string "Welcome the Movie Company System"
    Display the string "Enter 1 for adding Artist:"
    Display the string "Enter 2 for listing  Artists:"
    Display the string "Enter 3 for adding Disc:"
    Display the string "Enter 4 for listing  Discs:"
    Display the string "Enter 5 for adding Track   for Disc:"
    Display the string "Enter 6 for listing   Tracks for Disc:"
}
```

Step 5: Detailed Design Pseudocode for methods

Movie Company System

UML MVC CLASS MODEL

```
Create the instance of the  Catalog  class
Catalog   catalog  = new  Catalog();

Since the catalog is private we need to call a public method in the catalog to
operate on the private instance variables of the catalog

public void  UCAddArtist(String artist){
   catalog.addArtist(artist);
}                                                    // UC1

public void  UCListArtists(){
    catalog.listArtists();
}                                                    // UC2

public void  UCAddDisc(String artistName, String discCatalogNumber, String
discTitle){
     catalog.addDisc(artistName, discCatalogNumber, discTitle);
}
                                                     // UC3
public void  UCListDiscs(){
    catalog.listDiscs();
}
                                                   // UC4
public void  UCAddTrack(String discCatalogNumber, String trackTitle, double time){
    catalog.addTrack(discCatalogNumber, trackTitle, time);
}                                                    // UC5

public void  UClistTracksForDisc(String discCatalogNumber){
    catalog.listTracksForDisc(discCatalogNumber);
}
                                                   // UC6
public void  UCrestoreCatalog(){
    catalog.restoreCatalog();
}
                                                   // UC7
public void  UCsaveCatalog(){
    catalog.saveCatalog();
}                                                  // UC8
```

Controller	
- catalog: Catalog	
+ UCAddArtist(String artist): void	// UC1
+ UCListArtists(): void	// UC2
+ UCAddDisc(String artistName, String discCatalogNumber, String discTitle): void	// UC3
+ UCListDiscs(): void	// UC4
+ UCAddTrack(String discCatalogNumber, String trackTitle, double time): void	// UC5
+ UClistTracksForDisc(String discCatalogNumber): void	// UC6
+ UCrestoreCatalog(): void	// UC7
+ UCsaveCatalog(): void	// UC8

OOD – "blue print" DONE!

AND so ON…

Can run software on PAPER!

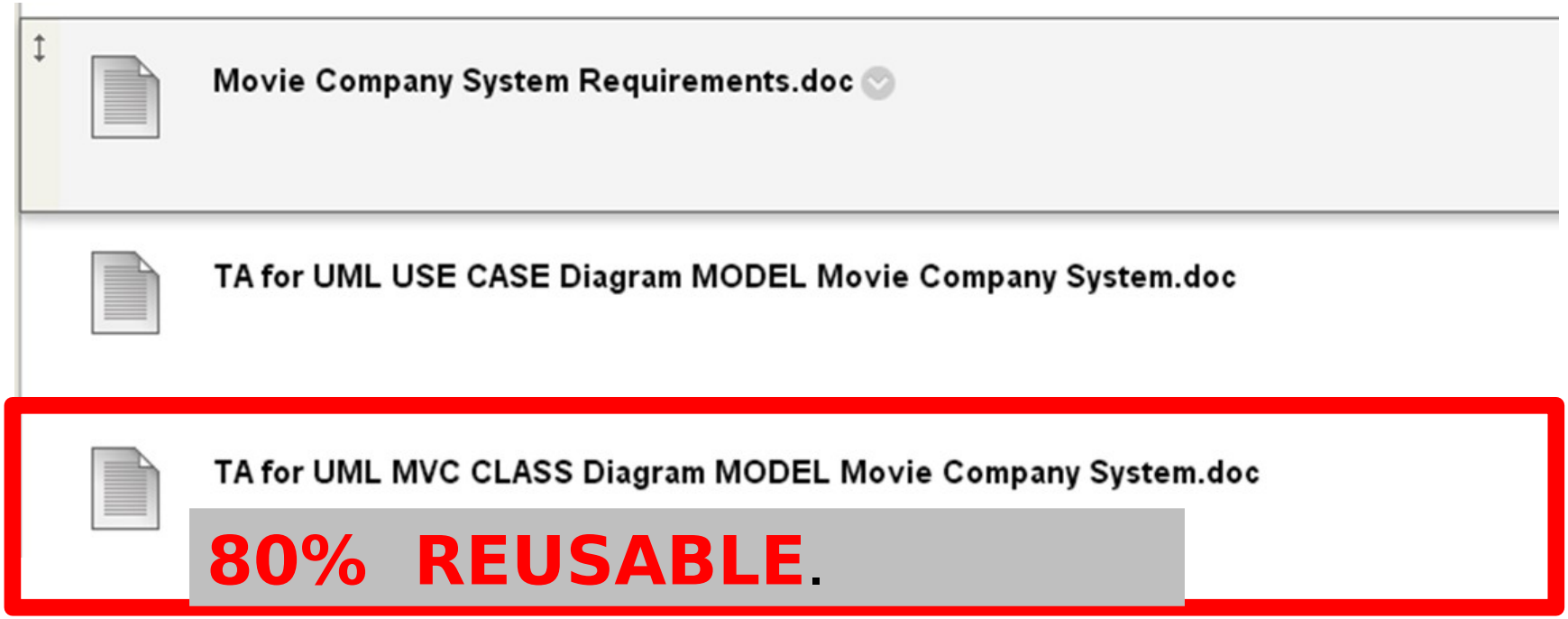
Step 5: Detailed Design Pseudocode for methods

Movie Company System

UML MVC CLASS MODEL

Please use TEMPLATE

“TA for UML MVC CLASS Diagram MODEL Movie Company System.doc “



Movie Company System

UML MVC CLASS MODEL

Complete example.

TA for UML MVC CLASS Diagram MODEL Movie Company System.doc

TA (Textual Analysis) for UML MVC CLASS Diagram MODEL

for Movie Company System

View

MovieCompanySystem

Controller

View 1: showSelection()

1 The Movie Company has the following affiliated Artists (maximum 10)

Catalog 1: MAX_ARTISTS = 10

2 **"Miles Davis", with 3 Discs:**

3 Disc 1: Title "Miles Ahead" with Track "Miles Ahead" that is 3.34 minutes

4 Disc 2: Title "Tutu with Tracks: "Tutu", 4.56, "Portia", 3.34, "Tomaas", 3.56

5 Disc 3: Title "Kind of Blue" with Tracks "So What", 4.54, "Freddie Freeloader, 5.25,

6 "Blue in Green", 2.56

7 **"John Coltrane", with 1 Disc:**

8 Disc 1: Title "Blue Train" with Blue Train" that is 5.15 minutes; "Moments Notice",

9 5.52, "Locomotion", 6.18

Catalog 6: numberOfDiscs
Catalog 5: numberOfArtists
Catalog 4: Disc[MAX_DISCS] discs
Catalog 3: Artist[MAX_ARTISTS] artists

Catalog

10 The Movie Company wants to keep a catalog of its artists and their discs (maximum

Catalog 1: addArtist(String artist);
Controller 1: UCAddArtist(String artist);

Artist

Artist 1: name

11 30) Artists have a name. The Movie Company can add an artist or get a listing of all

Catalog 2: MAX_DISCS = 30

Controller 2: UCListArtists();
Catalog 2: listArtists();

12 its artists.

Disc

13 Each artist produces music Discs. A Disc is associated with an Artist. A Disc is cataloged

Catalog 3: addDisc(String artist, String discCatalogNumber, String disc);
Controller 3: UCAddDisc(String artist, String discCatalogNumber, String disc);

14 with a discCatalogNumber. The Movie Company can add a disc or get a listing of all its

Disc 2: discCatalogNumber

Controller 4: UCListDiscs();
Catalog 4: listDiscs();

Track 1: addTrack(Track track);
Track 2: getTrack(int i) Track;

15 discs.

Track

Track 1: title Track 2: time

Disc 3: numberOfTracks

16 A Disc has a number of Tracks. The Track have a title and a duration time. A Disc has

Catalog 5: addTrack(String discCatalogNumber, String track, double time);
Controller 5: UCAddTrack(String discCatalogNumber, String track, double time);

17 a catalogNumber, a title, the number of tracks. The Movie Company can add a track or

Disc 1: MAX_TRACKS = 10

Disc 4: Track[MAX_TRACKS] tracks

18 get a listing of all tracks for a given disc.

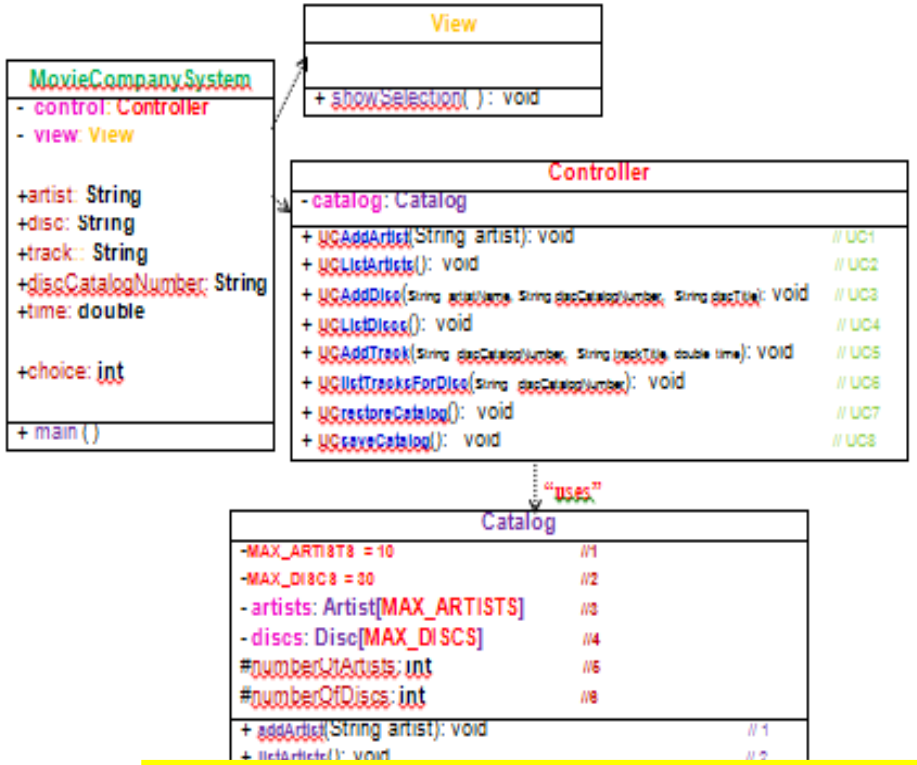
Controller 6: UCListTracksForDisc(String discCatalogNumber);
Catalog 6: listTracksForDisc(String discCatalogNumber);

Disc 5: Artist artist

Controller 7: UCrestoreCatalog();
Catalog 7: restoreCatalog();

Controller 8: UCsaveCatalog();
Catalog 8: saveCatalog();

All steps on one Requirements document!



- catalog: Catalog

Create the instance of the Catalog class

```
Catalog catalog = new Catalog();
```

Since the catalog is private we need to call a public method in the catalog to operate on the private instance variables of the catalog

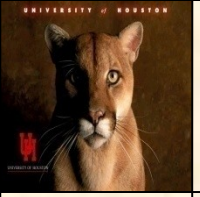
```
public void UCAddArtist(String artist){
   catalog.addArtist(artist);
}                                          // UC1

public void UCListArtists(){
    catalog.listArtists();
}                                          // UC2

public void UCAddDisc(String artistName, String discCatalogNumber, String
discTitle){
     catalog.addDisc(artistName, discCatalogNumber, discTitle);
}
                                           // UC3
public void UCListDiscs(){
    catalog.listDiscs();
}
                                           // UC4
                                                          double time){
    catalog.saveCatalog();
}
```

AND SO ON!

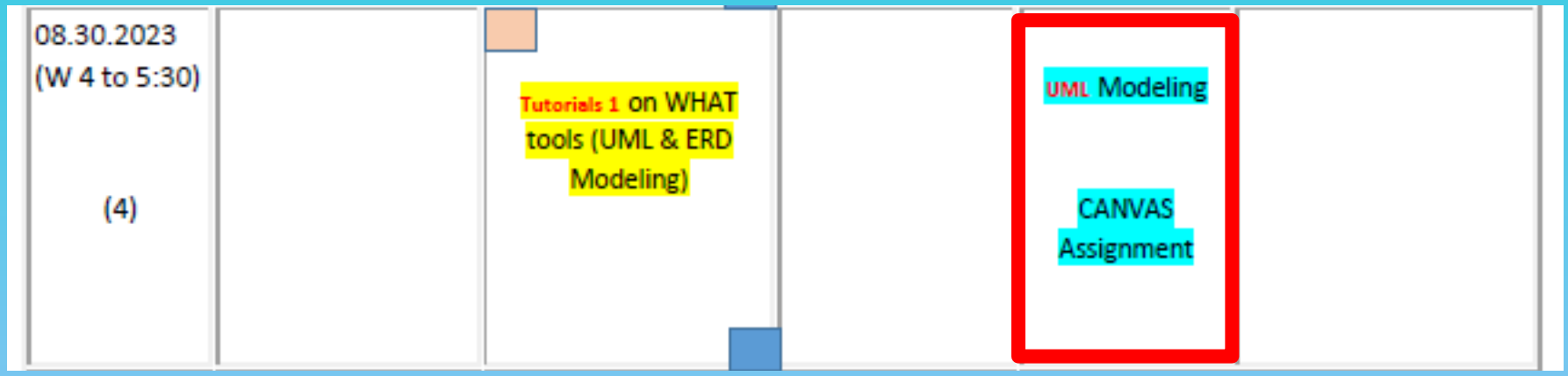
80% of the total effort (blue print)



I love UML!

The End

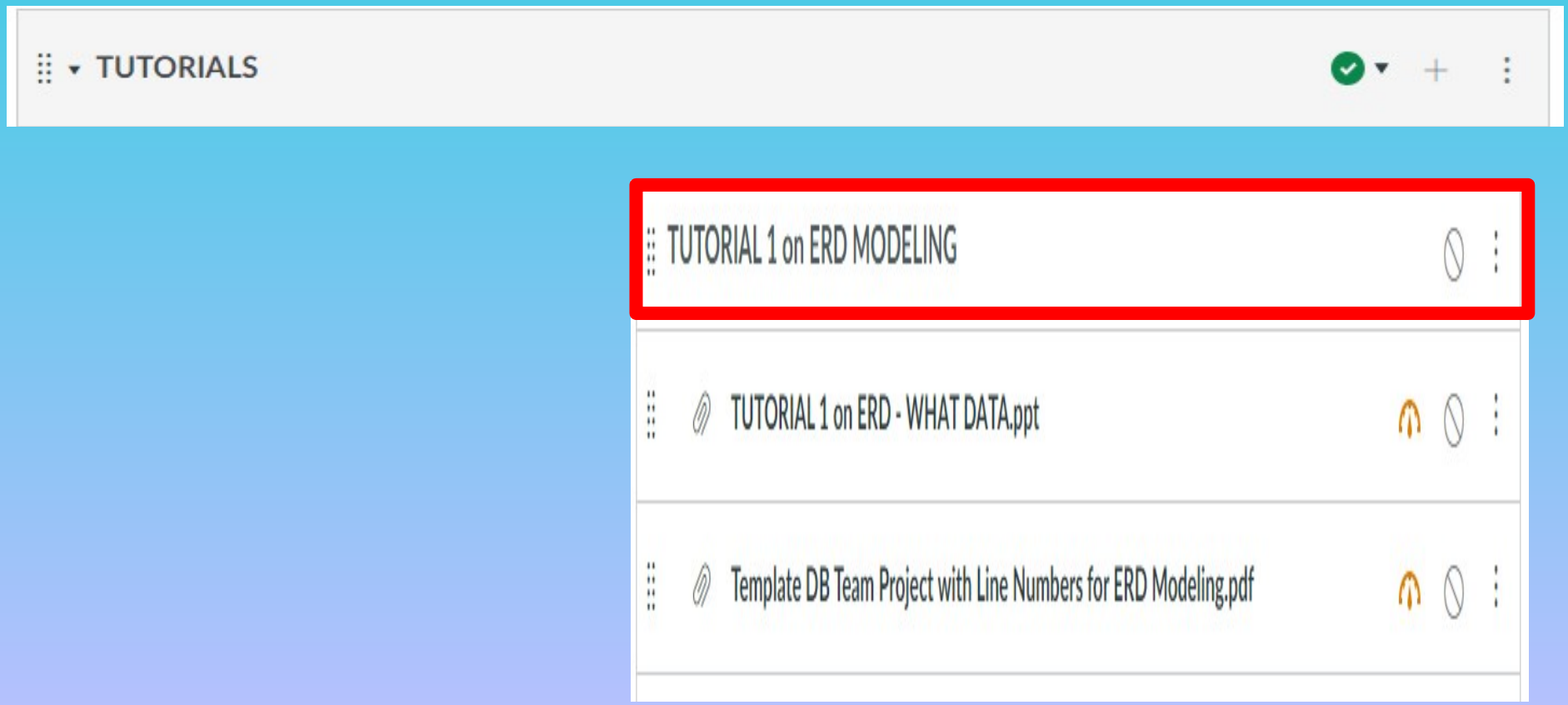
VH, feedback on UML Modeling for DVD Application



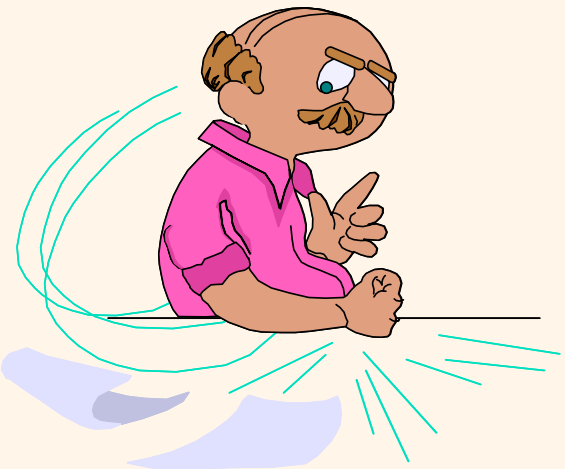
VH, show only solution on UML Modeling for DVD Application

From 4:40 to 5:00 PM – 20 minutes.

ERD Modelling



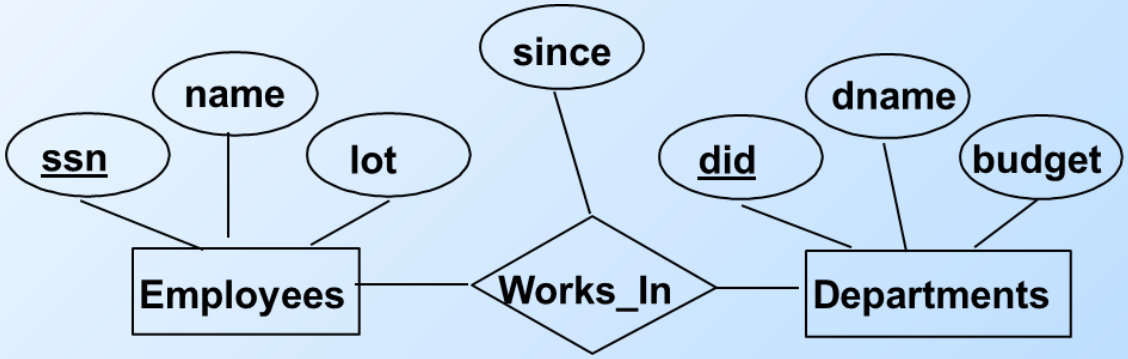
The Entity Relationship Diagr
Data Modeling
*(**WHAT**)*

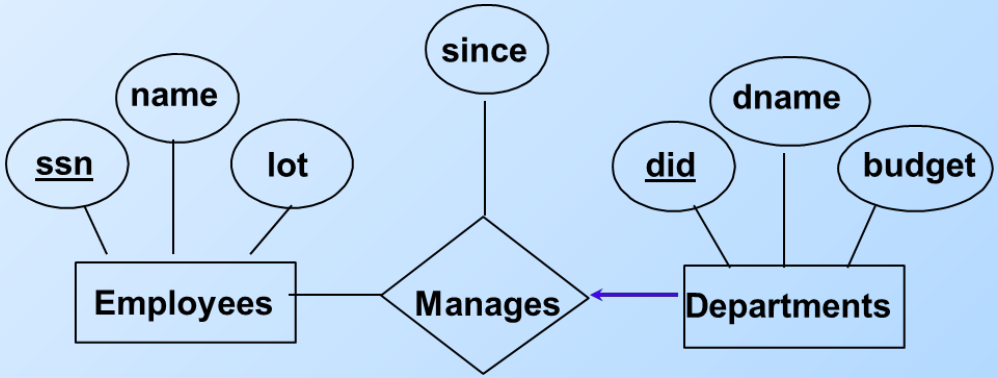


Data Modeling (WHAT)

Domain/Textual Analysis on Data Requiremen

ERD Language

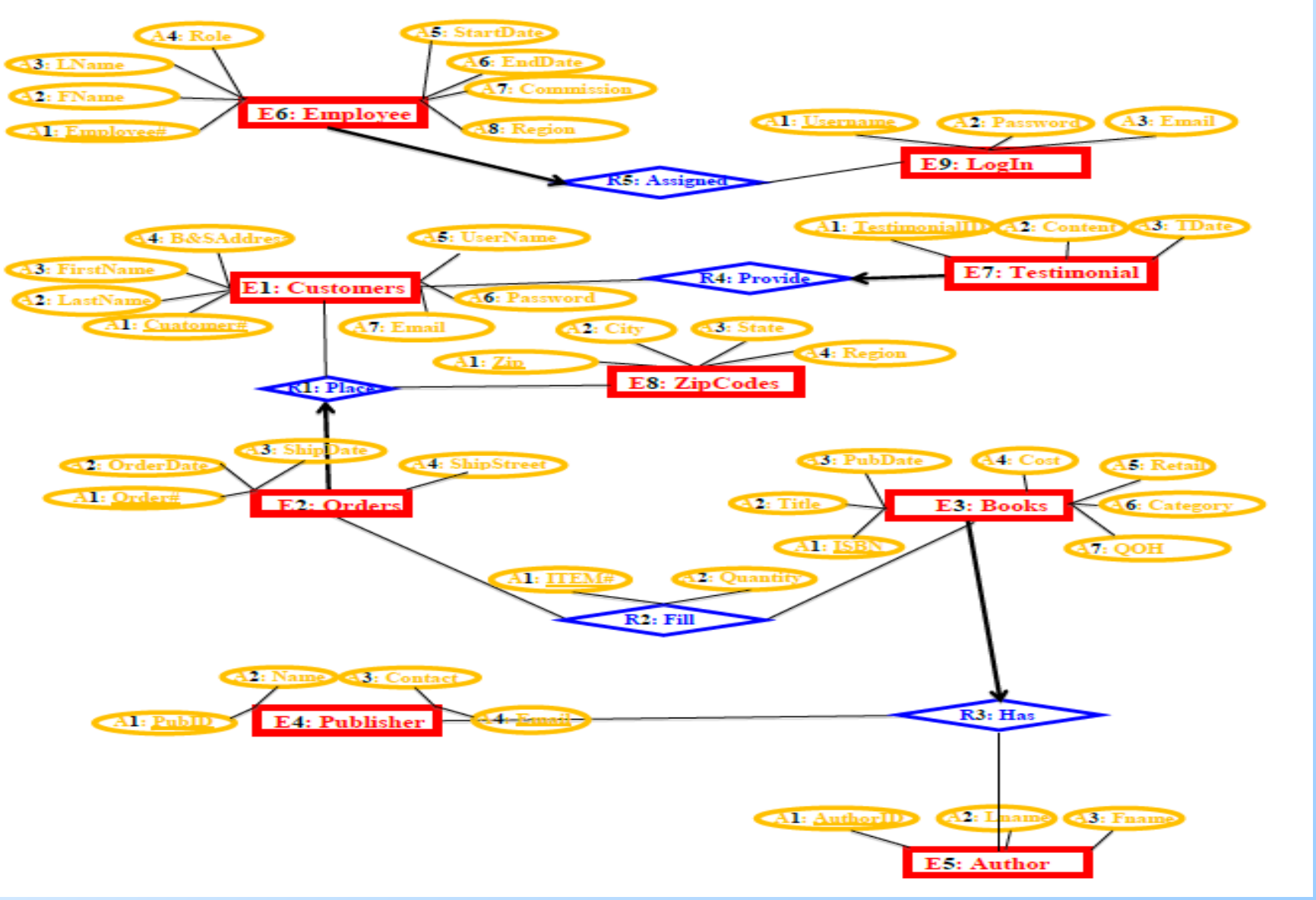




Data Modeling (WHAT)

????

ERD Model

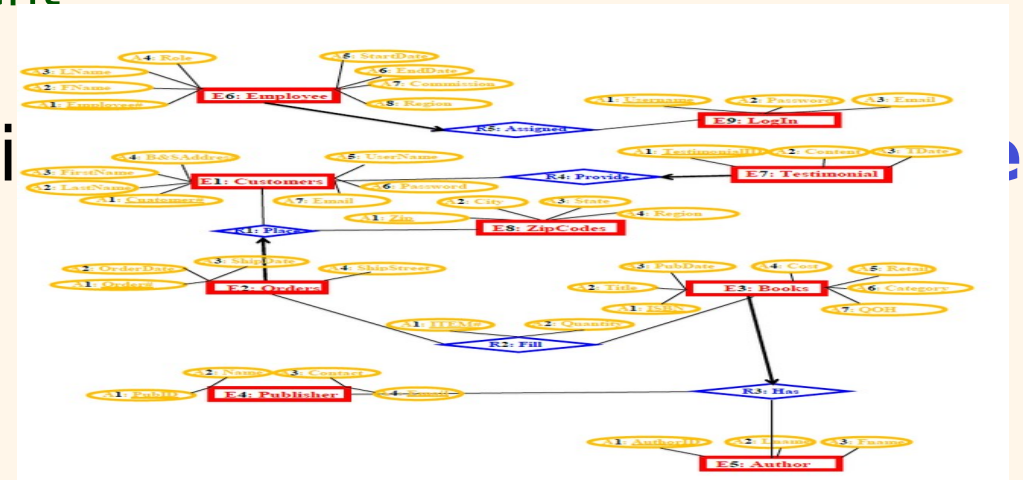


Data Modeling and Data Models

- **Data** Models
  - Relatively simple representations of complex real-world data structures
  - Often graphical

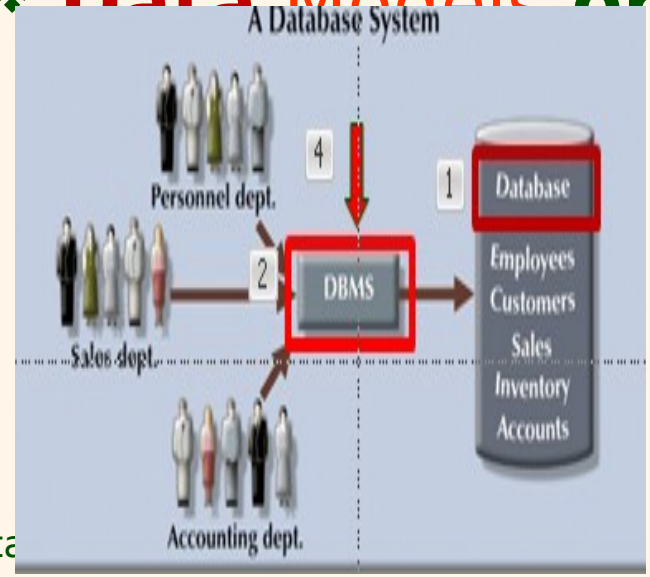
- Model: an **abstraction** of a real-world object or event
  - Useful in **understanding complexities** of the real-world environment

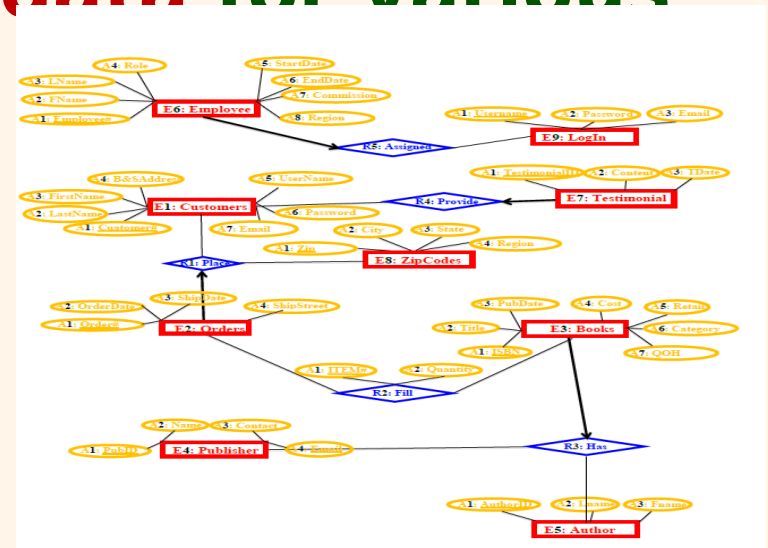
- **Data** Modeli **erative**



The Importance of ***Data Models***

- Facilitate interaction among the **Designer**, the **Applications Programmer**, and the **End user**
- **End Users** have different views and needs for **Data**
- **Data Models organizes data for various**





HealthCare.gov October 21, 2013

http://www.newyorker.com/online/blogs/elements/2013/10/why-the-healthcaregov-train-wreck-happened-in-slow-motion.html

Early in a project, there is a phase in which the client and the contractor work together to create a description of what is to be built. This is called the specification and building a complex software product without a clear, fixed set of specifications is impossible. The *Times* reported that

ERD Modeling

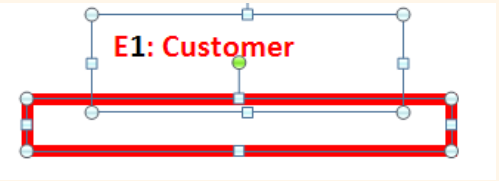
> the biggest contractor, CGI Federal, was awarded its $94 million contract in December 2011. But the government was so slow in issuing specifications that the firm did not start writing software code until this spring.... As late as the last week of September, officials were still changing features of the Web site.

This is like being told to build a skyscraper without any blueprints, while the client keeps changing the desired location of things like plumbing and wiring.

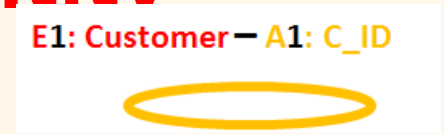
The *Times* also quoted an "insurance executive working on information technology," who said that "we foresee a train wreck" because "we don't have the I.T. specifications." He also said that "the political people in the administration do not understand how far behind they are." I've been a software contractor for the better part of fifteen years, and no one spends that long developing software without being involved in a few troubled projects. One thing they all have in common is that "train wrecks" are never a surprise to anyone working on them. They are not discrete events; they are part of drawn out processes. We only saw the wreckage of Healthcare.gov on October 1st, but the contractors have been working on a wreck for almost two years.

Data Model Basic Building Blocks

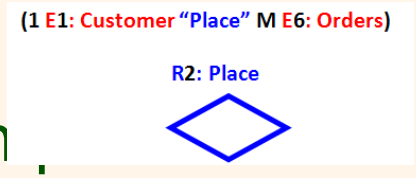
- **Entity**: anything about which **Data** are to be collected and stored



- Attribute: a characteristic of an **Entitv**



- Relationship: describes an association among **Entities**
  - One-to-many (1:M) relationship
  - Many-to-many (M:N or M:M) relationsh
  - One-to-one (1:1) relationship

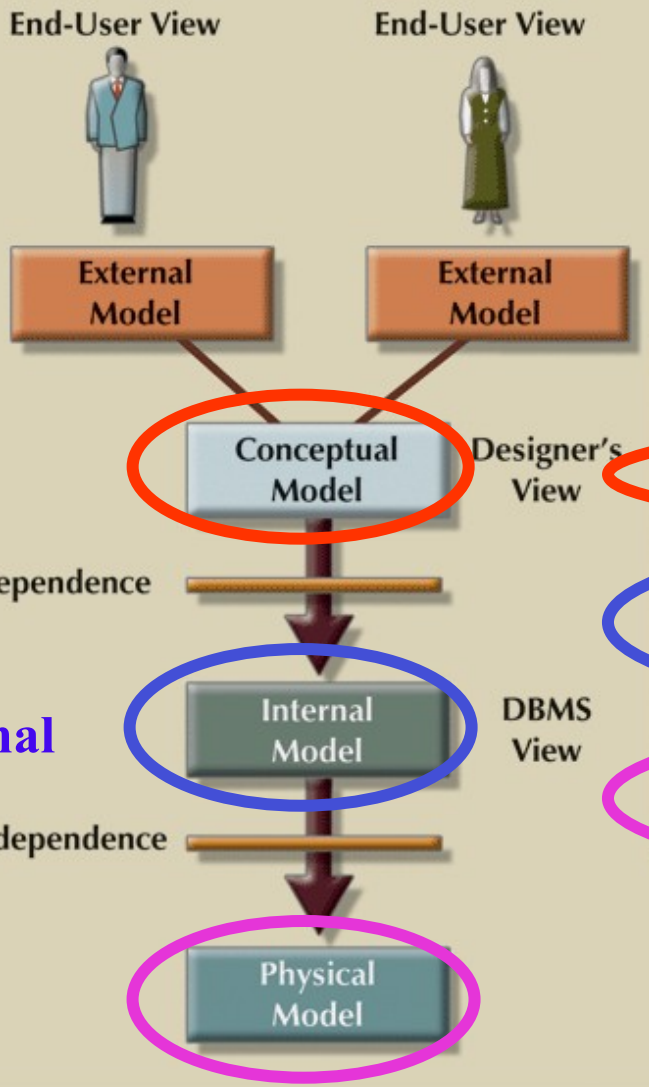


- Constraint: a restriction placed on the **Data**

Business Rules

- Descriptions of policies, procedures, or principles within a specific **organization**
  - Apply to any organization that stores and uses **Data** to generate information
- Must be in writing and kept up to date
- Must be easy to understand and widely disseminated
- Describe characteristics of **Data** as viewed by the

Data Models



What - ERD

How - Relational

SQL

Degree of Abstraction		Characteristics
High	ER	Hardware-independent
	Object-Oriented	
Medium	Relational	Hardware-independent Software-dependent
Low	Network Hierarchical	Hardware-dependent Software-dependent

Data Modeling Checklist

BUSINESS RULES

- Properly document and verify all business rules with the end users.
- Ensure that all business rules are written precisely, clearly, and simply. The business rules must help identify entities, attributes, relationships, and constraints.
- Identify the source of all business rules, and ensure that each business rule is accompanied by the reason for its existence and by the date and person(s) responsible for verifying and approving the business rule.

DATA MODELING

Naming Conventions: All names should be limited in length (database-dependent size).

- Entity Names:
  - Should be nouns that are familiar to business and should be short and meaningful
  - Should include abbreviations, synonyms, and aliases for each entity
  - Should be unique within the model
  - For composite entities, may include a combination of abbreviated names of the entities linked through the composite entity
- Attribute Names:
  - Should be unique within the entity
  - Should use the entity abbreviation or prefix
  - Should be descriptive of the characteristic
  - Should use suffixes such as _ID, _NUM, or _CODE for the
  - Should not be a reserved word
  - Should not contain spaces or special characters such as @
- Relationship Names:
  - Should be active or passive verbs that clearly indicate the

Entities:

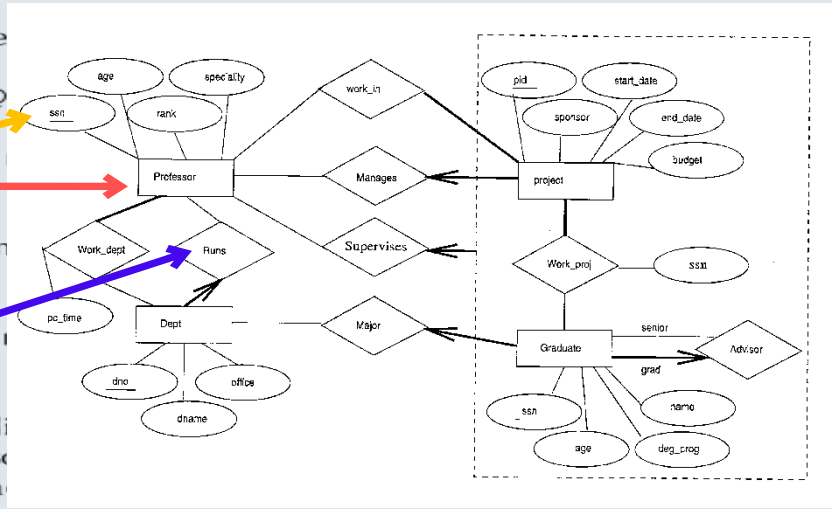
- Each entity should represent a single subject.
- Each entity should represent a set of distinguishable entity in
- All entities should be in 3NF or higher.
- The granularity of the entity instance is clearly defined.
- The PK is clearly defined and supports the selected data gra

Attributes:

- Should be simple and single-valued (atomic data)
- Should include default values, constraints, synonyms, and ali
- Derived attributes should be clearly identified and include s
- Should not be redundant unless they are required for transa used as a foreign key
- Non-key attributes must be fully dependent on the PK attribute

Relationships:

- Should clearly identify relationship participants
- Should clearly define participation and cardinality rules



ER Model:

- Should be validated against expected processes: inserts, updates, and deletes
- Should evaluate where, when, and how to maintain a history
- Should not contain redundant relationships except as required (see attributes)
- Should minimize data redundancy to ensure single-place updates
- Should conform to the minimal data rule: "All that is needed is there and all that is there is needed."

Pharmacy Enterprise ERD Model

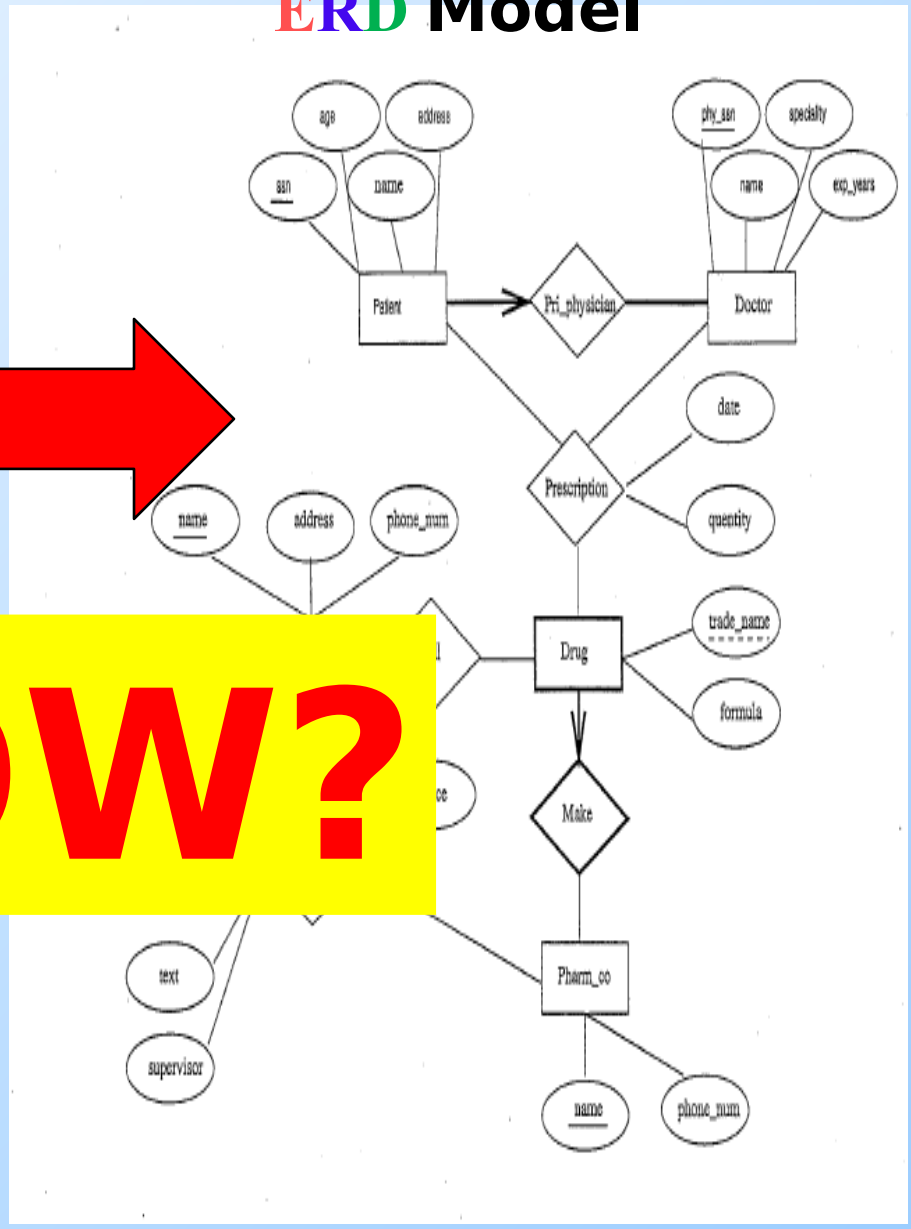
Requirements

The Prescriptions-R-X chain of pharmacies has offered to give you a free lifetime supply of medicine if you design its database. Given the rising cost of health care, you agree. Here's the information that you gather:

- Patients are identified by an SSN, and their names, addresses, and ages must be recorded.
- Doctors are identified by an SSN. For each doctor, the name, specialty, and years of experience must be recorded.
- Each pharmaceutical company is identified by name and has a phone number.
- For each drug, the trade name and formula must be recorded. Each drug is sold by a given pharmaceutical company, and the trade name identifies a drug uniquely from among the products of that company. If a pharmaceutical compa is deleted, you need not keep track of its products any longer.
- Each pharmacy has a name, address, and phone number.
- Every patient has a primary physician. Every doctor has at least one patient.
- Each pharmacy sells several drugs and has a price for each. A drug could be sold at several pharmacies, and the price could vary from one pharmacy to another.
- Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescript Each prescription has a date and a quantity associated that, if a doctor prescribes the same drug for the same only the last such prescription needs to be stored.
- Pharmaceutical companies have long-term contracts w maceutical company can contract with several pharmac contract with several pharmaceutical companies. For e store a start date, an end date, and the text of the cont
- Pharmacies appoint a supervisor for each contract. Ther visor for each contract, but the contract supervisor can of the contract.

1. Draw an ER diagram that captures the preceding information. Identify any constraints not captured by the ER diagram.
2. How would your design change if each drug must be sold at a fixed price by all pharmacies?
3. How would your design change if the design requirements change as follows: If a doctor prescribes the same drug for the same patient more than once, several such prescriptions may have to be stored.

ERD Model



HOW?

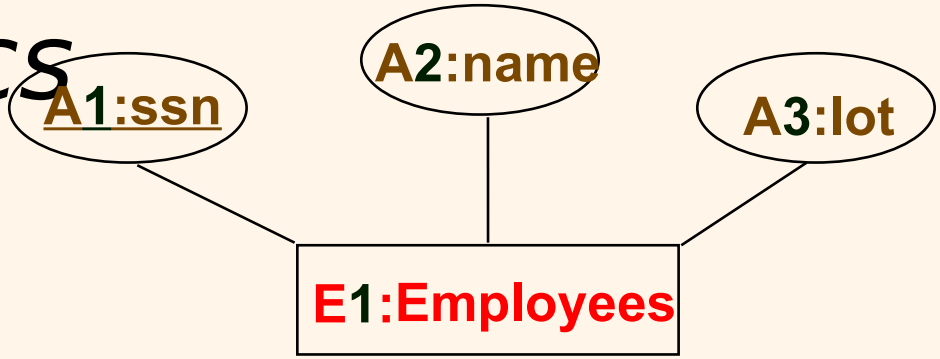
Case Study
Requirements

- A company **Data**base needs to store information about employees, departments, and children of employees(dependents).

- Employees *work* in a department; each department is *managed by* an employee.

- Employees can purchase insurance policies to *cover* their dependents; a child must be identified uniquely by *name* when the parent (who is an employee; *assume that only one parent works for the company*) is known.

ERD Model Basics

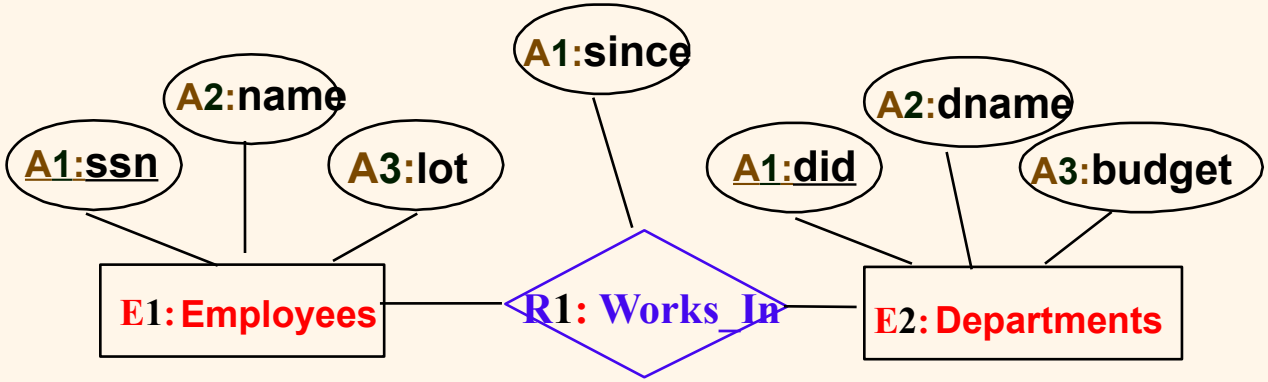


- *Entity:* Real-world object distinguishable from other objects. An *Entity* is described using a set of *attributes*.

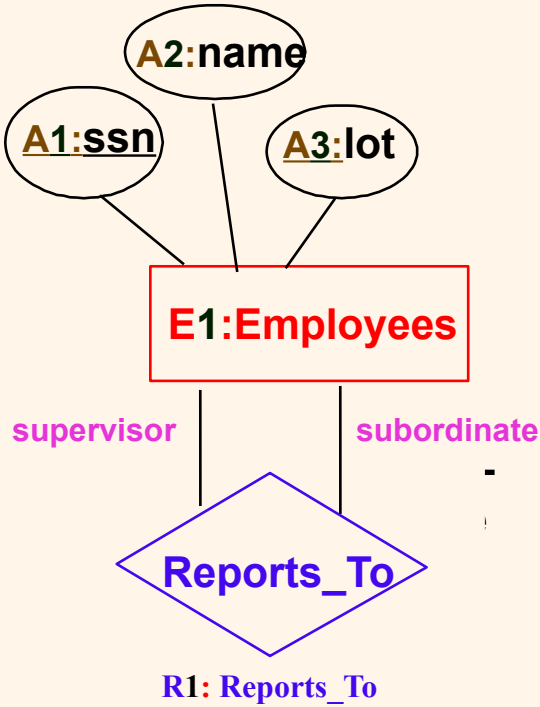
- *Entity Set*: A collection of similar *Entity*s. e.g., all employees.
  - All entities in an *Entity Set* have the same set of attributes.
  - Each *Entity Set* has a *key*.
  - Each attribute ha t in data types)

What is the *key*?

ERD Model Basics

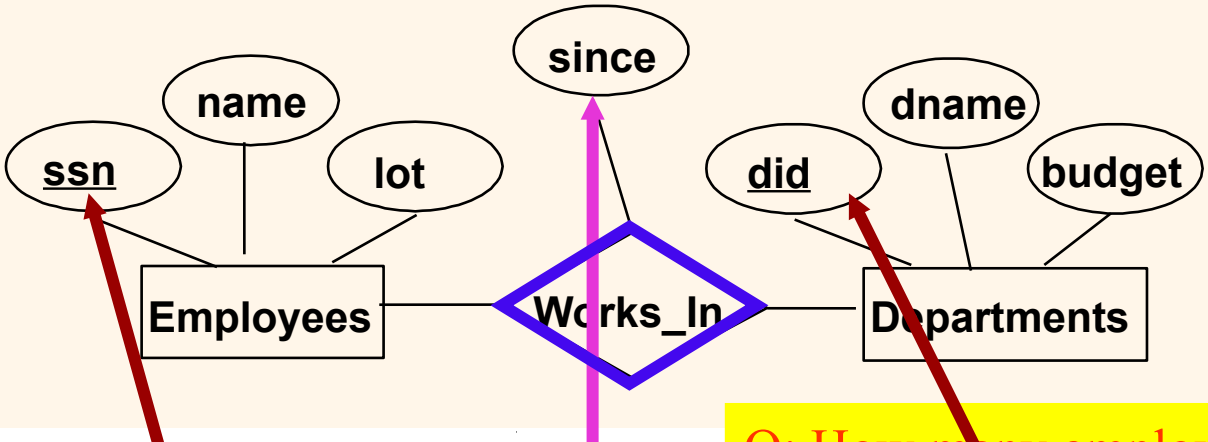


(M E1: Employees R1: "Works_In" M E2: Departments)



- *Relationship*: Association among two or more Entities. e.g., Attishoo Works_In Pharmacy Department.
- *Relationship Set*: Collection of similar Relationships.
  - An n-ary Relationship Set **R** relates n *Entity* sets E1 ... En; each Relationship in **R** involves Entities e1 E1, ..., en En

ERD Model Basics



Q: How many employees are in department 51?

A: **2**

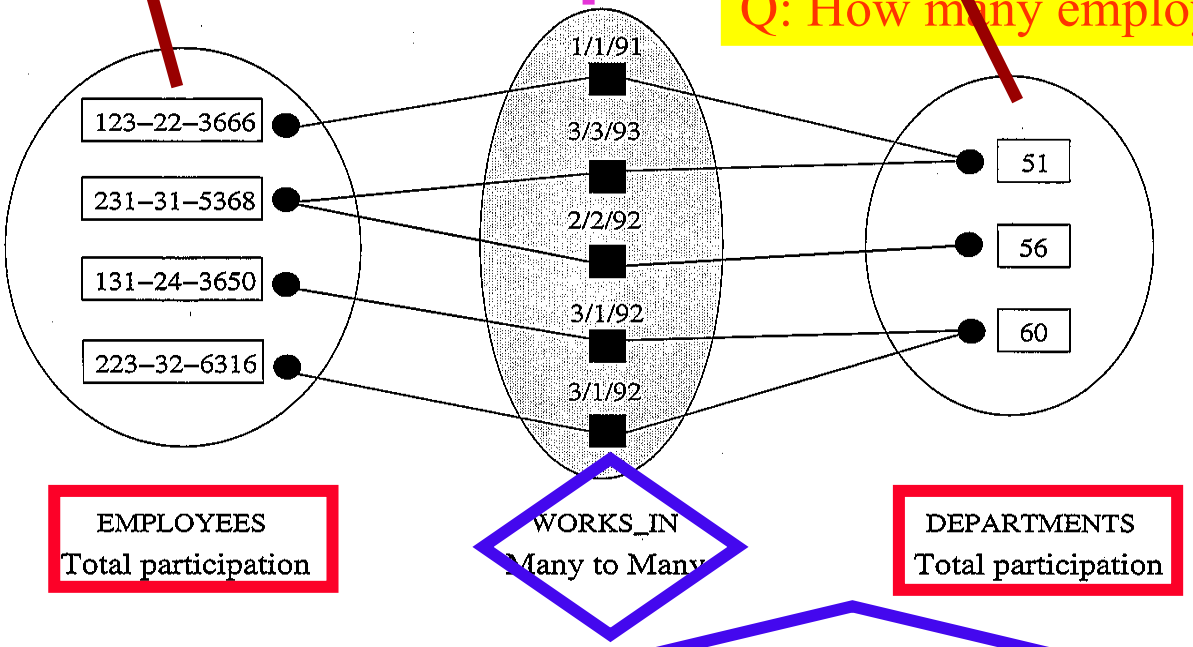
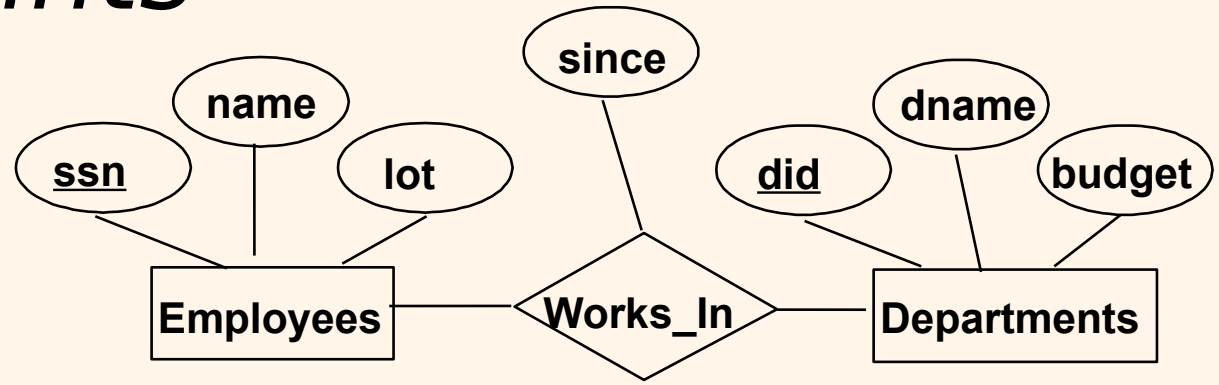


Figure 2.3 An Instance of th Works_In Relationship Set

Q: How many attributes does **Works_In** has?

A: **3**

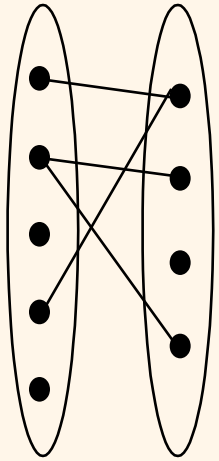
Key Constraints



Consider Works_In:

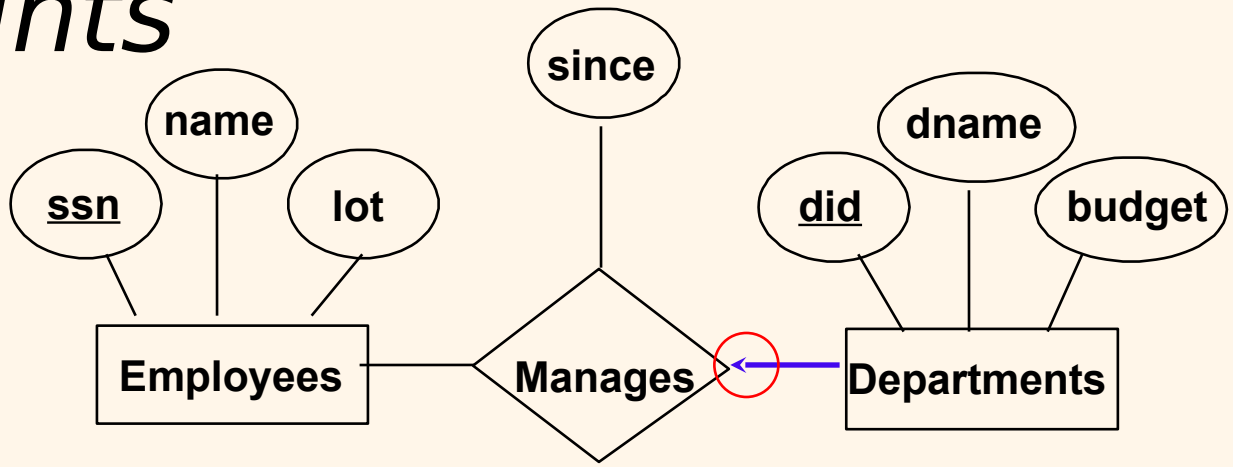
An Employee can work in many Departments;

A Department can have many Employees.



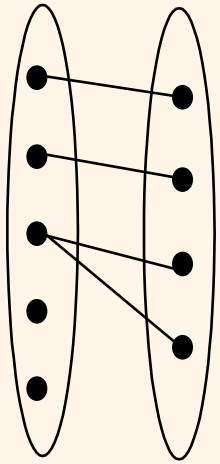
Many-to-Many

Key Constraints



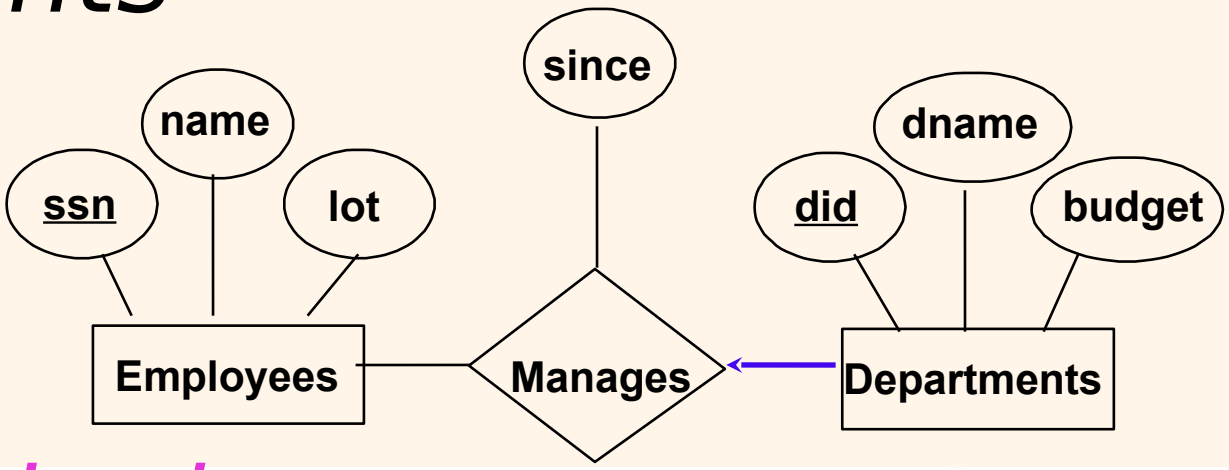
In contrast:

Each department has **at most one** manager, according the *key constraint* on Manages.



1-to-Many

Key Constraints



?????

Each department has ***at most one*** **manager.**

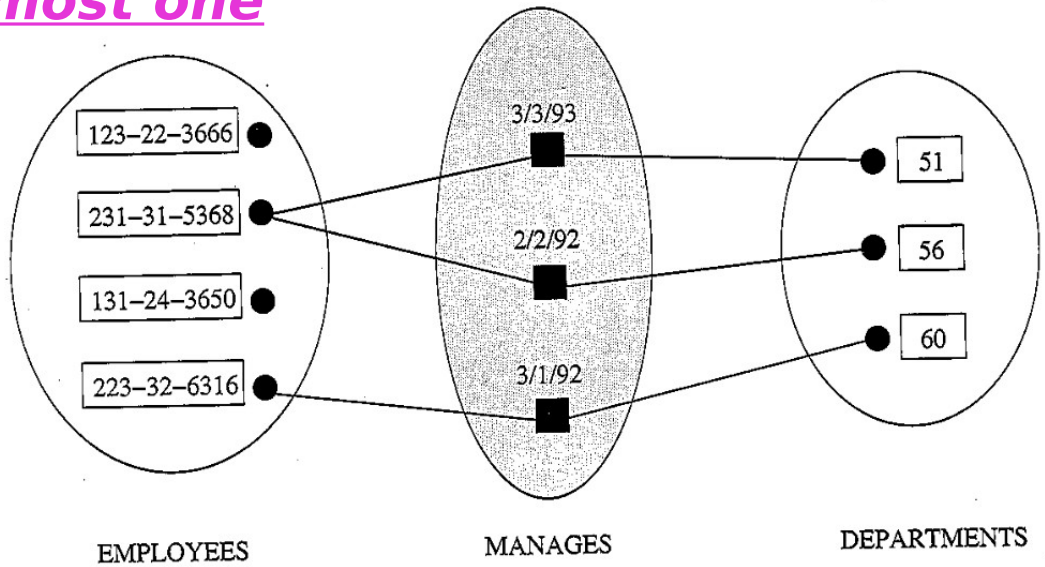


Figure 2.7 An Instance of the Manages Relationship Set

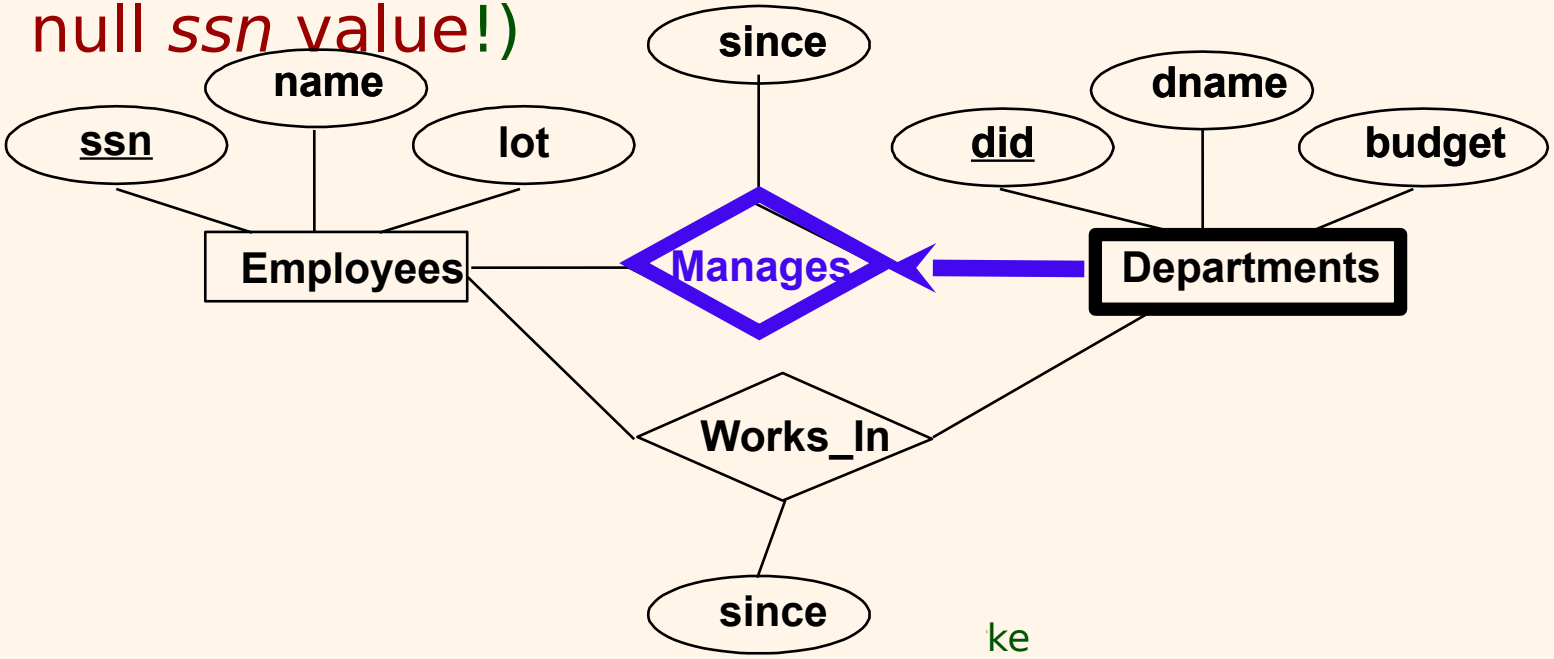
Q: How many attributes does **Manages** has?

A: **3**

Participation Constraints

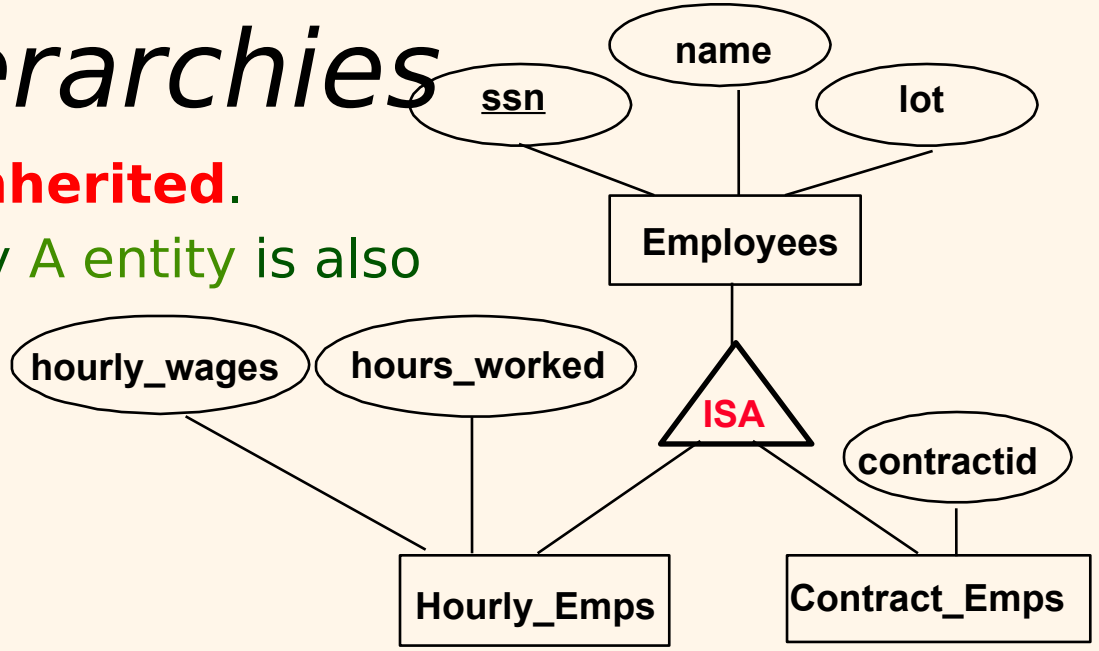
Does **every** Department have a Manager?

- If so, this is a *participation constraint*: the participation of Departments in Manages is said to be ***total*** (vs. *partial*).
  - Every *did* value in Departments table must appear in a row of the Manages table (with a non-null *ssn* value!)



ISA (`is a') Hierarchies

- As in **OOP**, attributes are **inherited**.
- If we declare A **ISA** B, every A entity is also considered to be a B entity.

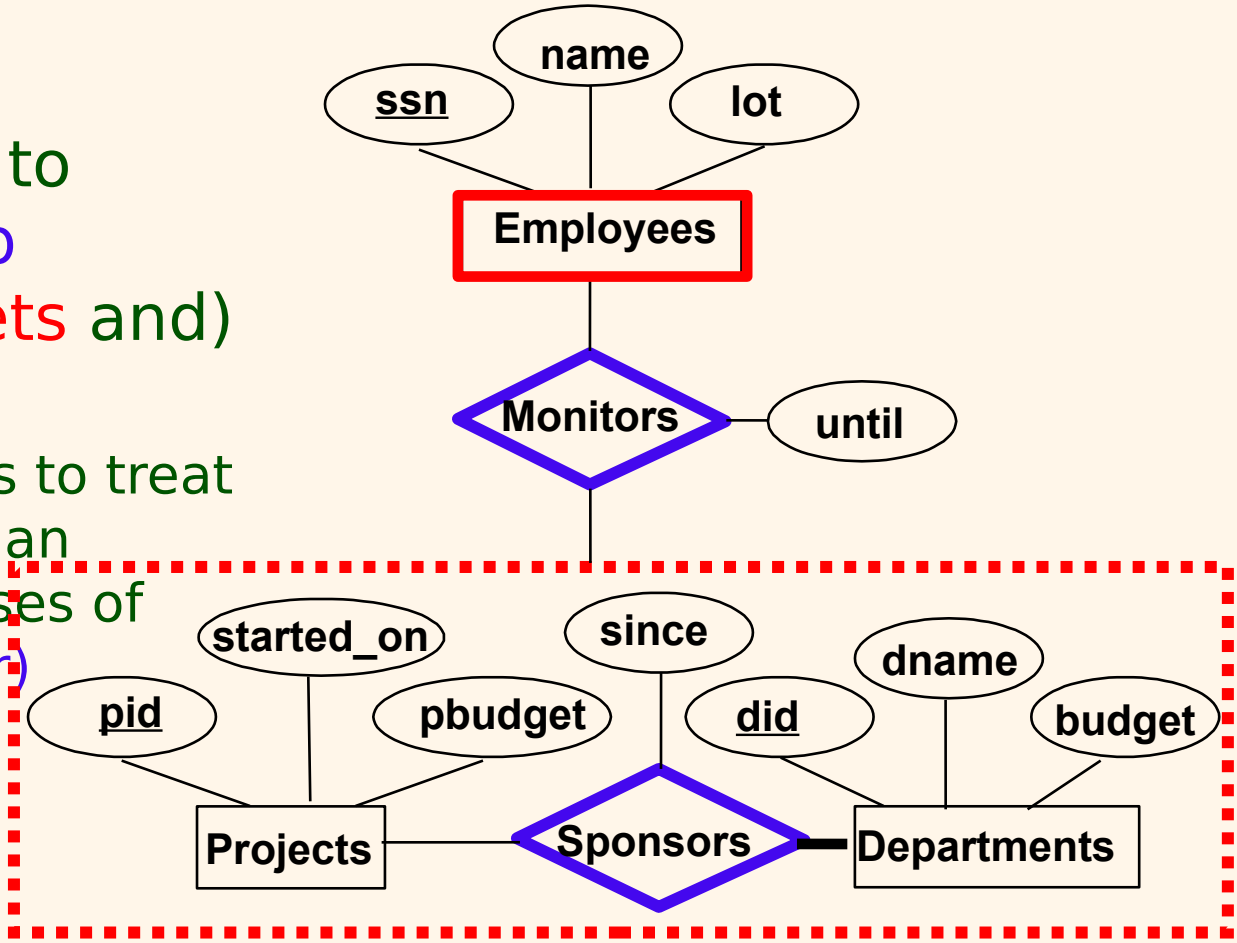


- *Overlap constraints*: Can Joe be an Hourly_Emps as well as a Contract_Emps entity? (*Allowed/disallowed*)
- *Covering constraints*: Does every Employees **Entity** also have to be an Hourly_Emps or a Contract_Emps entity? *(Yes/no)*
- Reasons for using **ISA**:
  - To add descriptive attributes specific to a subclass.
  - To identify **E**ntities that participate in a **R**elationship.

Aggregation

Used when we have to model a Relationship involving (Entitity Sets and) a *Relationship Set*.

- *Aggregation* allows us to treat a Relationship Set as an Entity Set for purposes of participation in (other) Relationships.



☛*Aggregation* *vs.* *ternary relationship*:

- ❖ Monitors is a distinct Relationship, with a descriptive attribute.
- ❖ Also, can say that each **Sponsorship** is Monitored by **at most one** Employee

2: Addendum username E3: Artist – A7: username email E3: Artist – A8: email

E3: Artist – A9: password

artistId E3: Artist – A1: artistID password E2: Customer – A2: lastName

E2: Customer

65 First, both the owner and the salespeople want to keep track o Customers last and first

E2: Customer – A8: country

E2: Customer – A7: zip E2: Customer – A5: city E2: Customer – A3: firstName

66 names, addresse (stree city state zip country), phone number (area code and phone

E2: Customer – A11: email

E2: Customer – A4: Street E2: Customer – A9: areaCode

E2: Customer – A6: state E2: Customer – A10: phoneNumber

67 number), and e-mail addresses by requesting a Customers Report and want to keep track

E3: Artists

E3: Artist – A3: firstName

68 o Artists las and first names and nationality by requesting an Artists Report. They

E3: Artist – A4: nationality

E3: Artist – A2: lastName

69 also want to know which artists have appeal to which customers by requesting a

E2: Customer – A1: customerID

R3: Interests

70 Customer Artists Preferences Report where th Customer Id is the input to such report.

(M E2: Customer "Interests" M E1: Artist)

71 The salespeople use this information to determine whom to contact when new art arrives

72 and to personalize verbal and e-mail communications with their customers.

E4: Works – A3: medium

E4: Works – A1: workID E4: Works R4: about

73 When TEAM5 purchases new art, data about the artist, the nature of the work, the

(1 E3: Artist "about" M E4: Works)

74 acquisition date, and the acquisition price are recorded in the database backend. Also, on

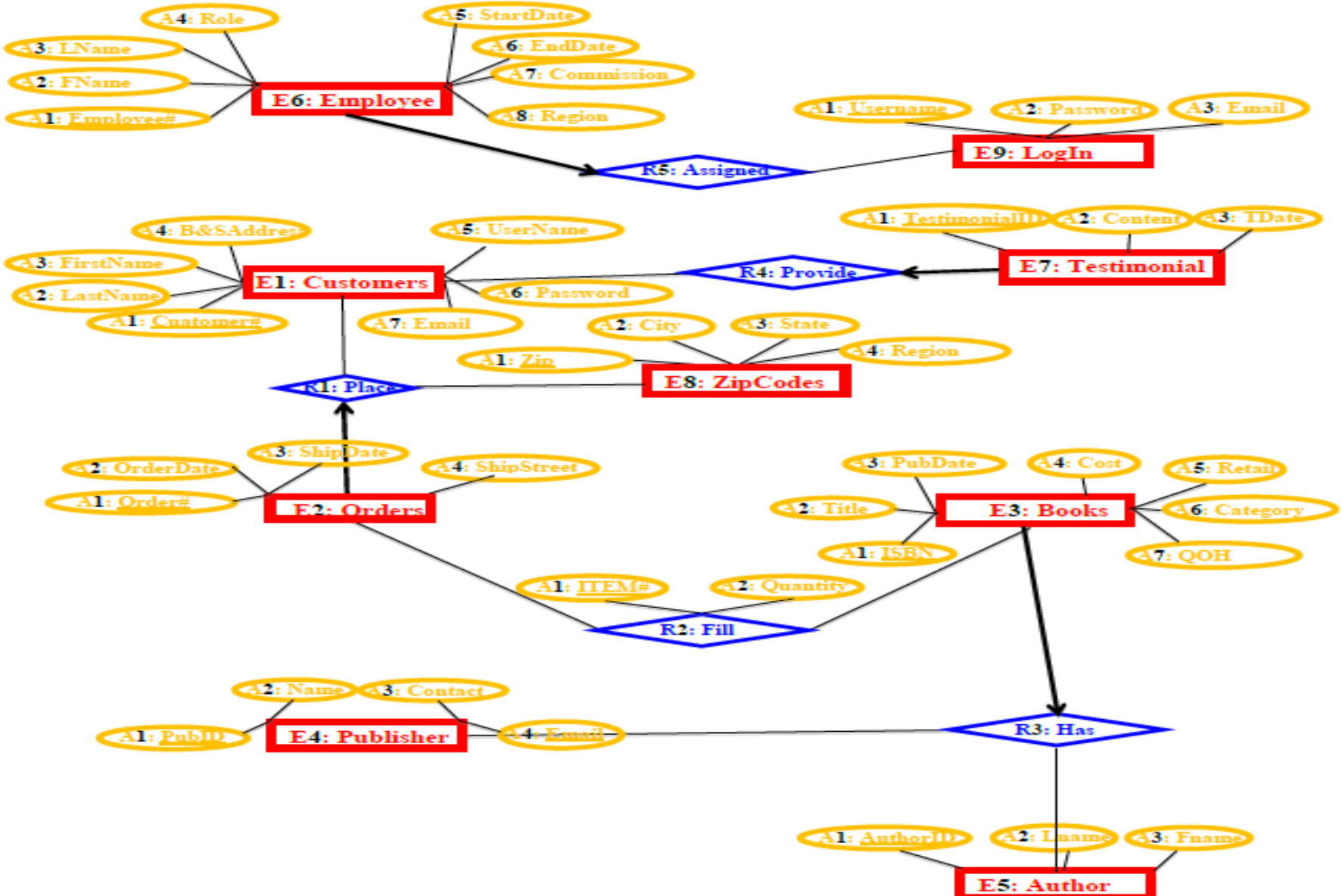
75 occasion, TEAM5 repurchases art from a customer and resells it, thus a work may appear

76 in the TEAM5 gallery multiple times.

CUT & PASTE

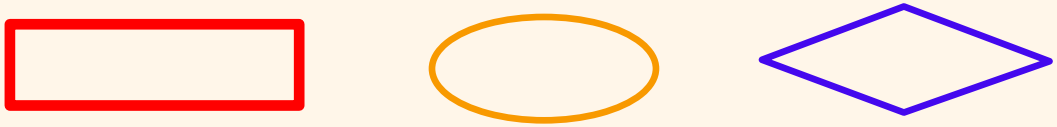
REARANGE

ERD Model (WHAT data)

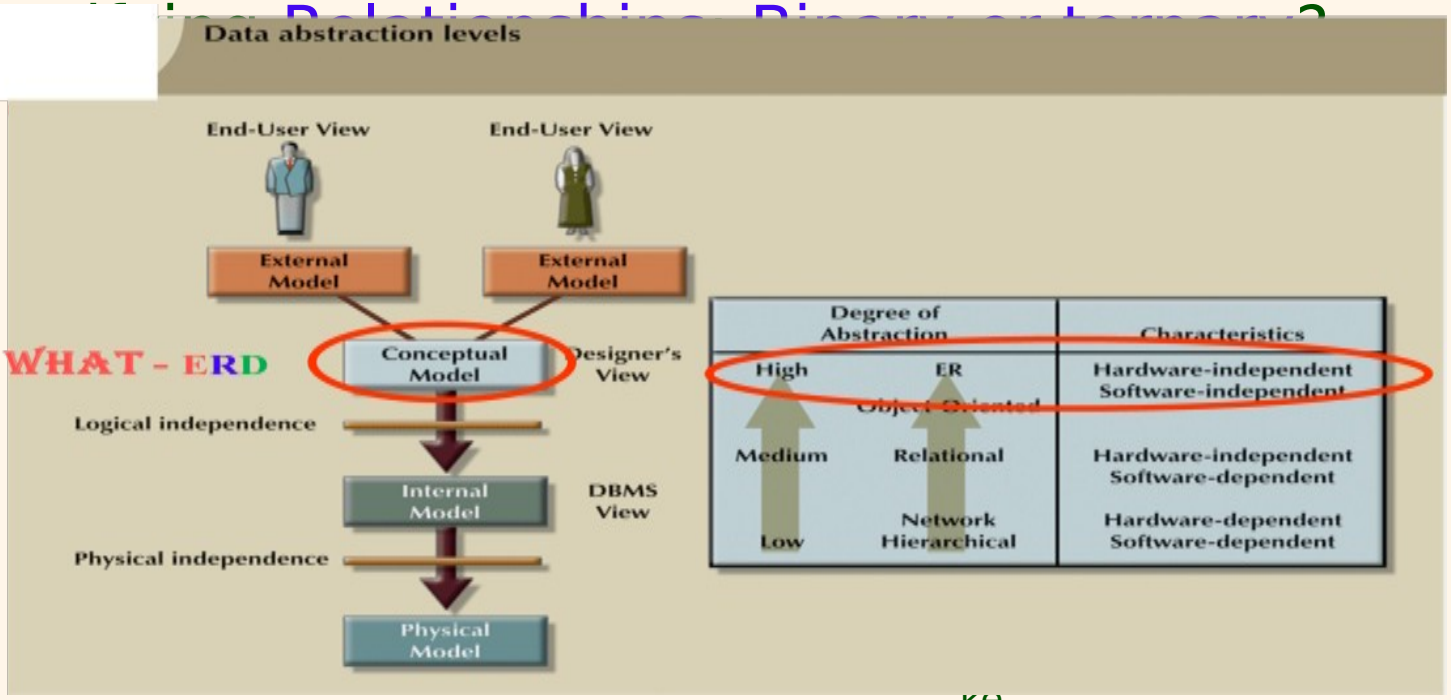


Conceptual Model Using the ERD Model

- Choices:



  - Should a concept be modeled as an Entity or an attribute?
  - Should a concept be modeled as an Entity or a Relationship?
  - Ic[redacted] Ag



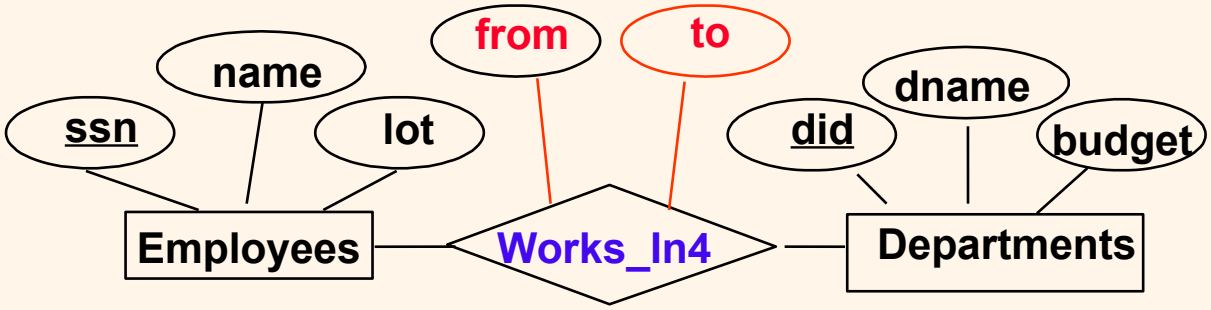
Entity vs. *Attribute*

- Should *address* be an **attribute** of Employees or an **E**ntity (connected to Employees by a relationship)?

- Depends upon the use we want to make of address information, and the semantics of the data:
  - If we have several addresses per employee, *address* must be an **E**ntity (since attributes cannot be set-valued).
  - If the structure (city, street, etc.) is important, e.g., [illegible] in a given city,

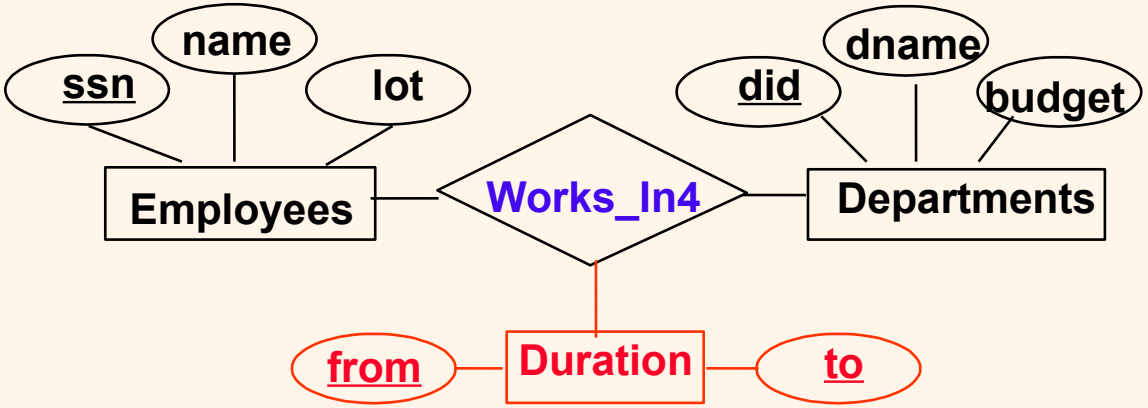
Entity vs. ***Attribute***

Works_In4 does not allow an Employee to work in a Department for two or more periods.



Similar to the problem of wanting to record several Addresses for an Employee

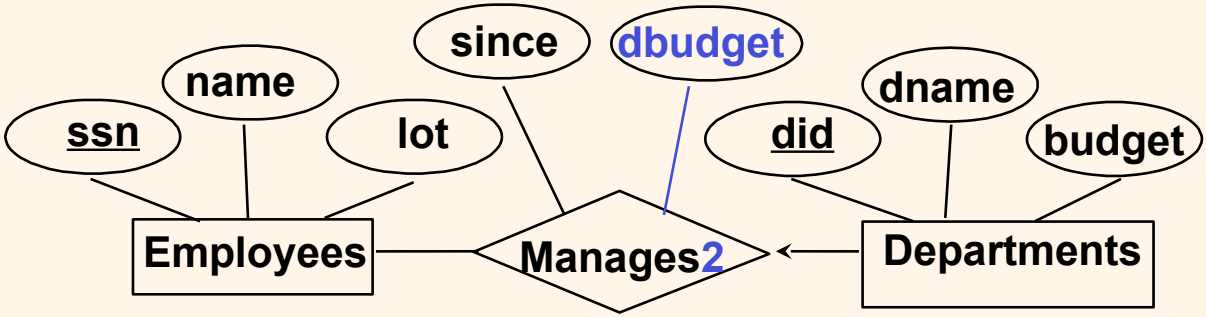
We want to record *several values of the descriptive* ***attributes*** *for each instance of this relationship.*



Accomplished by introducing new Entity Set

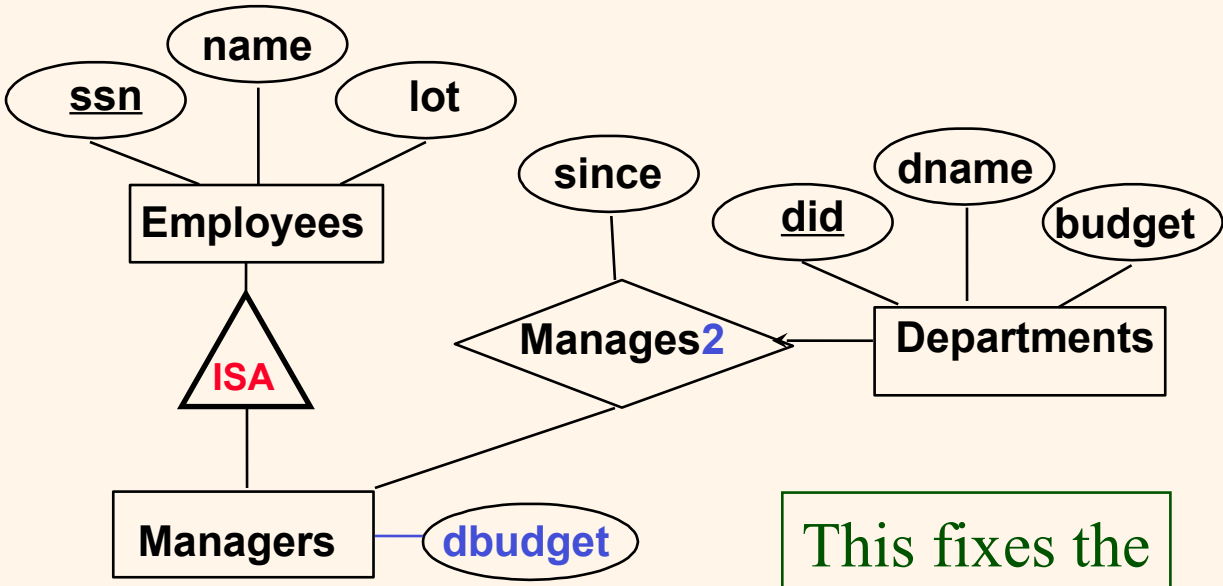
Entity vs. *Relationship*

ERD Model OK if a Manager gets a separate discretionary budget ***dbudget*** for each Department.



What if a Manager gets a discretionary budget that covers *all* managed Departments?

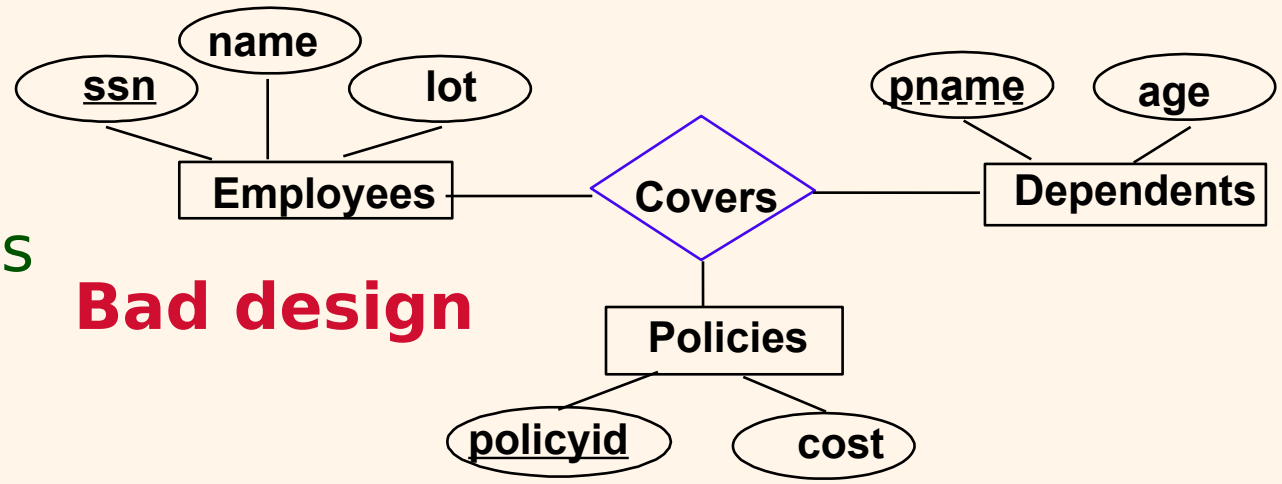
- Redundancy: ***dbudget*** stored for each Department managed by Manager.
- Misleading: Suggests



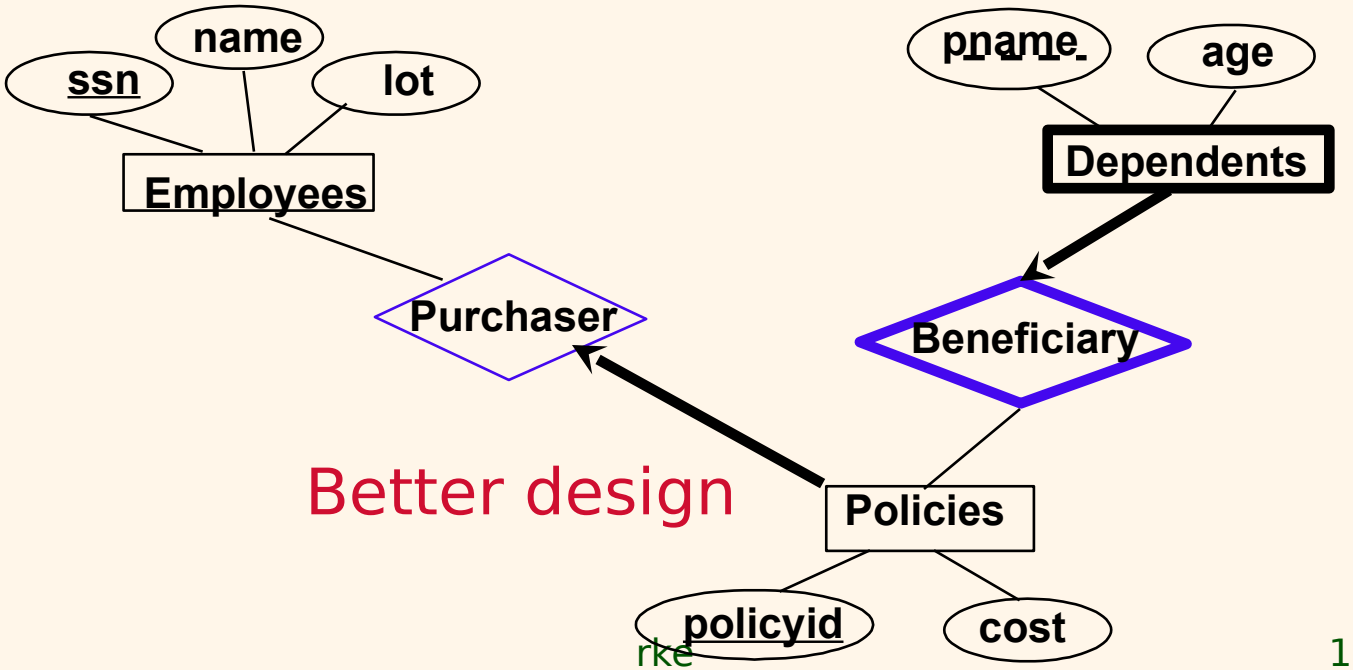
This fixes the problem!

Binary vs. Ternary Relationships

- If each Policy is owned by just 1 Employee, and each Dependent is tied to the covering Policy, **ERD** Model is inaccurate.



Bad design



Better design

*Binary **vs.** Ternary Relationships*

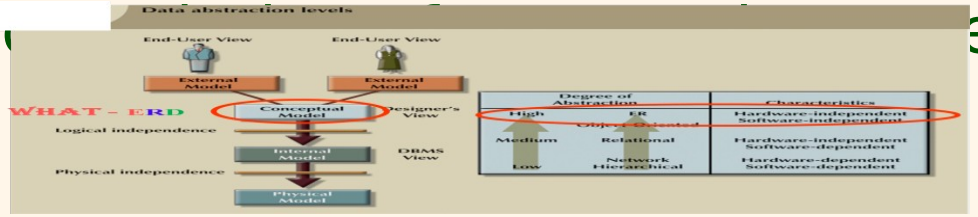
- Previous example illustrated a case when two binary relationships were better than one ternary relationship.

- An example in the other direction: a ternary relation Contracts relates entity sets Parts, Departments and Suppliers, and has descriptive attribute ***qty***.

- No combination of binary relationships is an adequate substitute:
  - S "can-supply" P, D "needs" P, and D "deals-with" S does not imply that D has agreed to buy P from S.
  - How do we record ***qty***?

Summary of Conceptual Design – ***WHAT***

- *Conceptual Design* follows ***requirements analysis*** (***DOMAIN/TEXTUAL ANALYSIS is DESIRED!!!!***)
  - Yields a high-level [illegible] ed

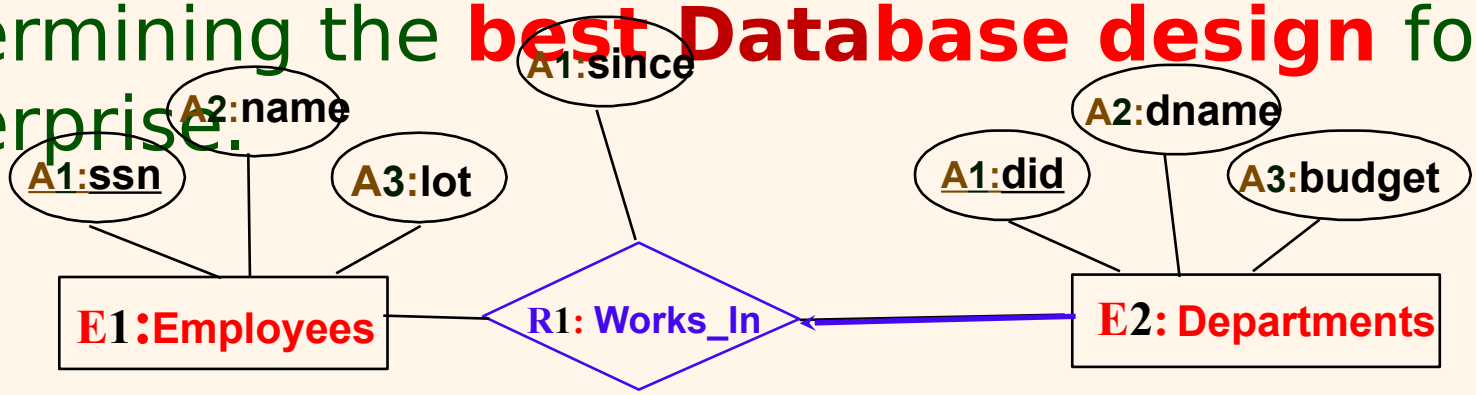


- Basic constructs: ***E****ntities*, ***R****elationships*, and *attributes* (of **E**ntities and **R**elationships).
- Some additional constructs: *weak* ***E****ntities*, ***ISA*** *hierarchies*, and *aggregation*.

Summary of ERD

- Several kinds of integrity constraints can be expressed in the ERD Model: *key constraints*, *participation constraints*, and *overlap/covering constraints* for **ISA** hierarchies. Some ***foreign key constraints*** are also **implicit** in the definition of a Relationship Set.

- Constraints play an important role in determining the **best Database design** for an enterprise.



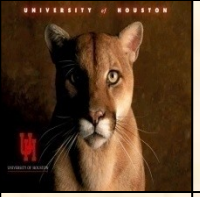
(M E1: Employees "R1: Works_In" M E2: Departments)

Summary of ERD

60% of the effort

- ERD Modeling is *subjective*. There are often many ways to Model a given scenario! **Analyzing alternatives** can be tricky, especially for a large enterprise. Common choices include:
  - Entity vs. attribute, Entity vs. Relationship, binary or n-ary Relationship, whether or not to use **ISA** hierarchies, and whether or not to use aggregation.

- Ensuring good **Data**base design: resulting [illegible] should be



I love ERD!

The End

At 5:00 PM.

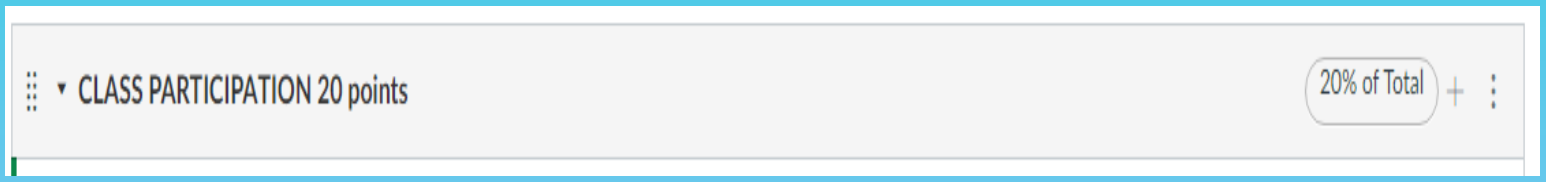
VH, next Lecture 3

09.06.2023 (W 4 to 5:30) (5)		Lecture 3: Software Development Process			

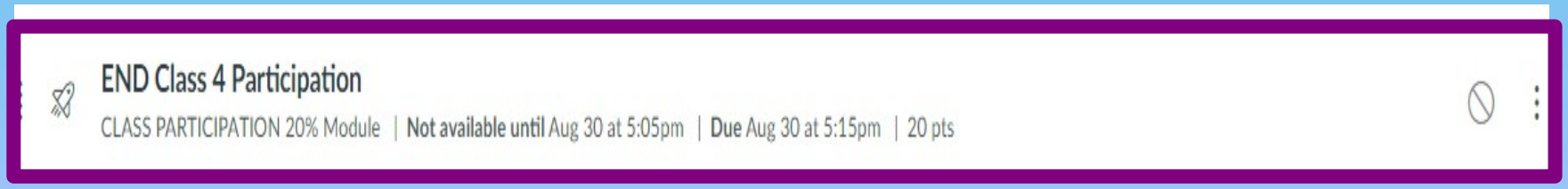
Enjoy Labor Holiday!

From 5:05 to 5:15 – 10 minutes.

08.30.2023 (W 4 to 5:30) (4)		Tutorials 1 on WHAT tools (UML & ERD Modeling)		UML Modeling CANVAS Assignment	



PASSWORD: IN TEAMS



VH, publish.

VH Download Meeting Participants

At 5:15 PM.

End Class 4

VH, upload Class 4 to CANVAS.

VH, Download Attendance Report
Rename it:
8.30.2023 Attendance Report FINAL