EECS 581
Software Engineering II

David O. Johnson

Fall 2024

Reminders

- Initial Requirements Stack & Story Point Estimate due (today): Friday, October 11, 11:59 PM
- Initial Architecture & Sprint 1 Requirements List due: Sunday, October 20, 11:59 PM

Any Questions?

In-Class Problem Solution

- 11-(10-7) In-Class Problem Solution.pptx

Any Questions?

Sources

- https://cs.ccsu.edu/~stan/classes/CS410/Notes16/05-SystemModeling.html
- https://en.wikipedia.org/wiki/Unified_Modeling_Language
- http://agilemodeling.com/essays/agileRequirements.htm

Agile and EECS 581

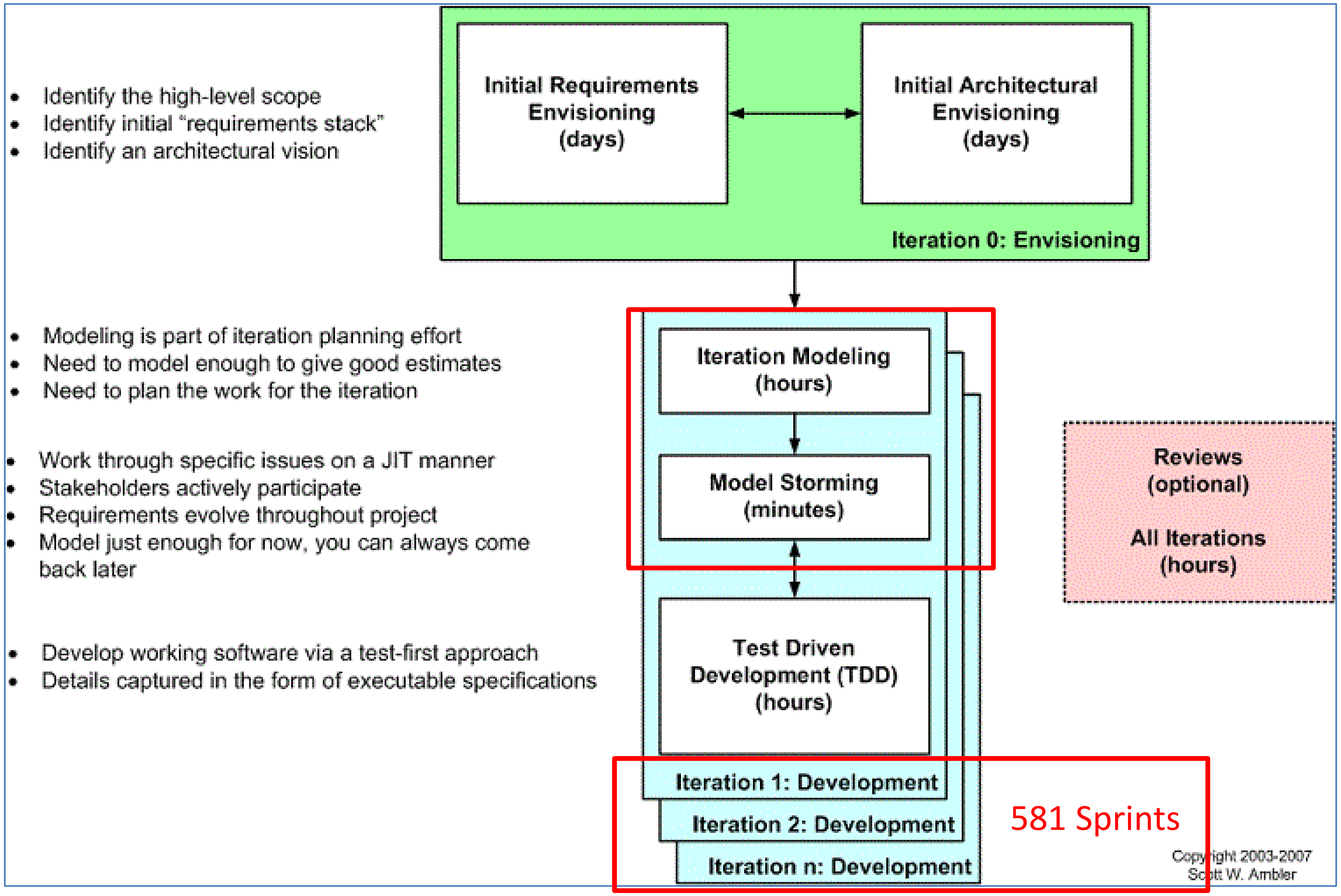
Today's Topic

Date	Day	Project Deliverable Due
8/26/2024	M	Team Formation Request
8/30/2024	F	
9/2/2024	M	
9/6/2024	F	
9/9/2024	M	Project 1
9/13/2024	F	
9/16/2024	M	
9/20/2024	F	
9/23/2024	M	Project 2
9/27/2024	F	
9/30/2024	M	
10/4/2024	F	
10/7/2024	M	Initial Requirements Stack & Story Point Estimate
10/11/2024	F	
10/14/2024	M	Initial Architecture & Sprint 1 Requirements List
10/18/2024	F	
10/21/2024	M	Sprint 1 Release & Sprint 2 Requirements List
10/25/2024	F	
10/28/2024	M	
11/1/2024	F	
11/4/2024	M	Sprint 2 Release & Sprint 3 Requirements List
11/8/2024	F	
11/11/2024	M	
11/15/2024	F	
11/18/2024	M	Sprint 3 Release & Final Sprint Requirements List
11/22/2024	F	
11/25/2024	M	
11/29/2024	F	
12/2/2024	M	Final Sprint Release & Presentation Video
12/6/2024	F	
12/9/2024	M	
12/13/2024	F	

Agile and EECS 581

581 Initial Requirements Stack & Story Point Estimate

Initial Architecture & Sprint 1 Requirements List



- Identify the high-level scope
- Identify initial "requirements stack"
- Identify an architectural vision
Initial Requirements Envisioning (days)
Initial Architectural Envisioning (days)
Iteration 0: Envisioning
- Modeling is part of iteration planning effort
- Need to model enough to give good estimates
- Need to plan the work for the iteration
- Work through specific issues on a JIT manner
- Stakeholders actively participate
- Requirements evolve throughout project
- Model just enough for now, you can always come back later
- Develop working software via a test-first approach
- Details captured in the form of executable specifications
Iteration Modeling (hours)
Model Storming (minutes)
Test Driven Development (TDD) (hours)
Reviews (optional)
All Iterations (hours)
Iteration 1: Development
Iteration 2: Development
Iteration n: Development
581 Sprints
Copyright 2003-2007 Scott W. Ambler

Initial Architecture & Sprint 1 Requirements List Instructions

Purpose: Develop an initial architecture for your 581 Agile project and a list of requirements for your first Sprint (Sprint 1).

Steps:

1. Create an Initial Architecture Document
2. Update your Agile Reference Stories spreadsheet as necessary.
3. Update your Requirements Stack spreadsheet as necessary.
4. Create and submit a Requirements Artifact for each requirement in Sprint 1.
   - Each Requirements Artifact should contain enough detail that your GTA can determine if the requirement is met in the Sprint 1 Release.
5. Submit a Team Peer Evaluation form

Requirements Artifacts Rubric

Requirements Artifacts (25% of grade – team based)				
		Grading Level		
Content	Points	Exceeds Expectations (90-100%)	Meets Expectations (80-89%)	Unsatisfactory (0-79%)
Artifacts	25	All of the artifacts are of sufficient detail that the GTA can tell if the requirement is delivered in Sprint 1 or not; some of the artifacts are the best ones from all of the GTA's teams.	All of the artifacts are of sufficient detail that the GTA can tell if the requirement is delivered in Sprint 1 or not.	Some of the artifacts do not have enough detail for the GTA to tell if the requirement is delivered in Sprint 1 or not.

Any Questions?

What Are Requirements Artifacts?

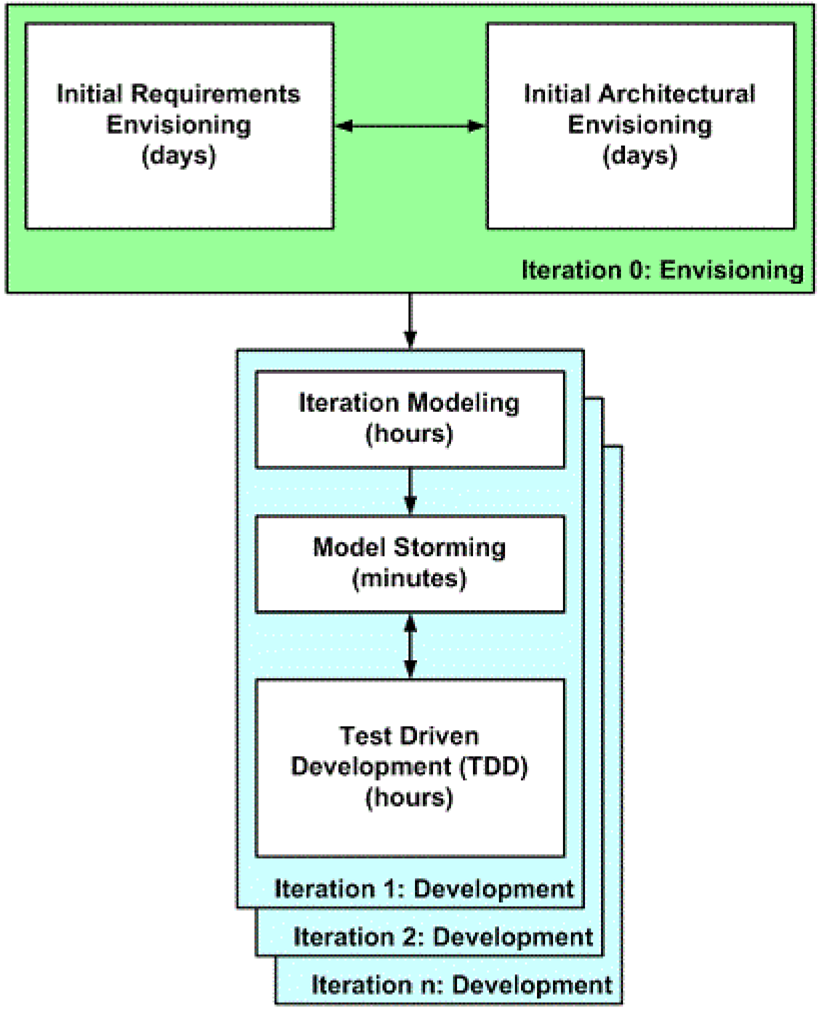
Output of Agile Iteration Modeling & Model Storming

Iteration Modeling

- Modeling is part of iteration planning effort
- Need to model enough to give good estimates
- Need to plan the work for the iteration

Model Storming

- Work through specific issues on a JIT manner
- Stakeholders actively participate
- Requirements evolve throughout project
- Model just enough for now, you can always come back later


Initial Requirements Envisioning (days)
Initial Architectural Envisioning (days)
Iteration 0: Envisioning
Iteration Modeling (hours)
Model Storming (minutes)
Test Driven Development (TDD) (hours)
Iteration 1: Development
Iteration 2: Development
Iteration n: Development

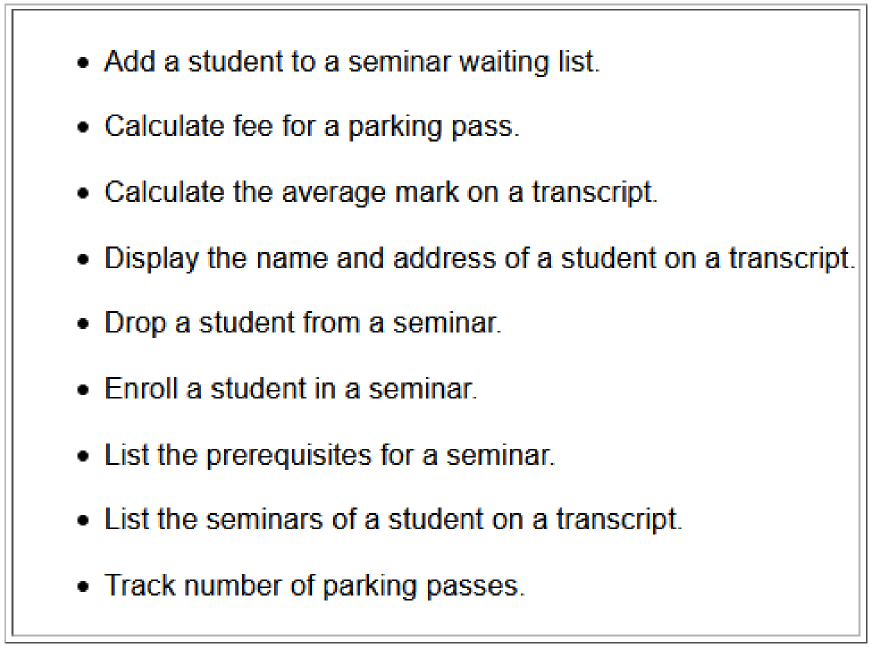
Informal Requirements Artifacts

- Generally, you start off with the informal artifacts and then progress to the more formal ones.
- You only need to formalize them enough to start the design and coding.
- Don’t get carried away with documenting the requirements.
- The three simplest informal artifacts are:
  – User stories (already used to estimate cost and duration of project or sprint)
  – Collection of features
  – User interface model

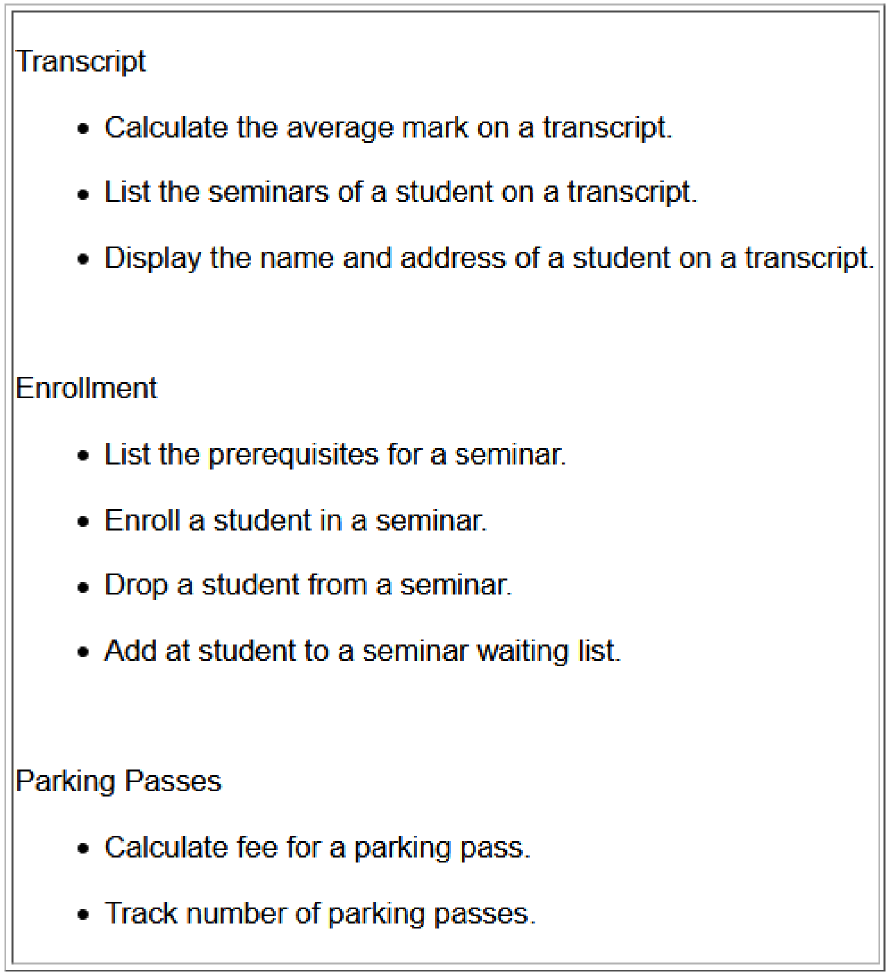
User Story

- Students can purchase monthly parking passes online.
- Parking passes can be paid via credit cards.
- Parking passes can be paid via PayPal.
- Professors can input student marks.
- Students can obtain their current seminar schedule.
- Students can order official transcripts.
- Students can only enroll in seminars for which they have prerequisites.
- Transcripts will be available online via a standard browser.

Collection of Features



- Add a student to a seminar waiting list.
- Calculate fee for a parking pass.
- Calculate the average mark on a transcript.
- Display the name and address of a student on a transcript.
- Drop a student from a seminar.
- Enroll a student in a seminar.
- List the prerequisites for a seminar.
- List the seminars of a student on a transcript.
- Track number of parking passes.



Transcript

- Calculate the average mark on a transcript.
- List the seminars of a student on a transcript.
- Display the name and address of a student on a transcript.

Enrollment

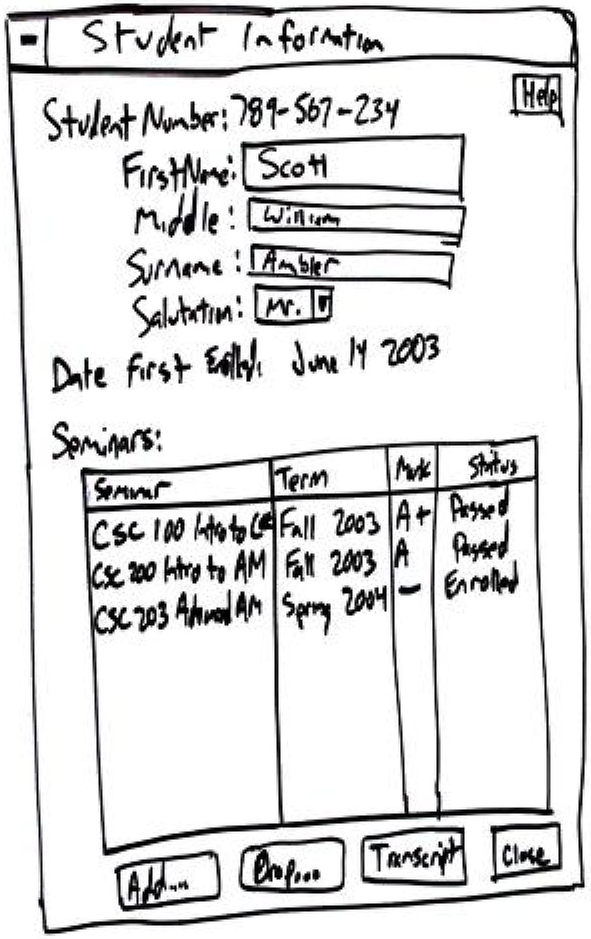
- List the prerequisites for a seminar.
- Enroll a student in a seminar.
- Drop a student from a seminar.
- Add at student to a seminar waiting list.

Parking Passes

- Calculate fee for a parking pass.
- Track number of parking passes.

User Interface Model

- For user interface intensive projects consider developing some screen sketches or even a user interface prototype.



Student Information

Help

Student Number: 789-567-234

First Name: Scott

Middle: William

Surname: Ambler

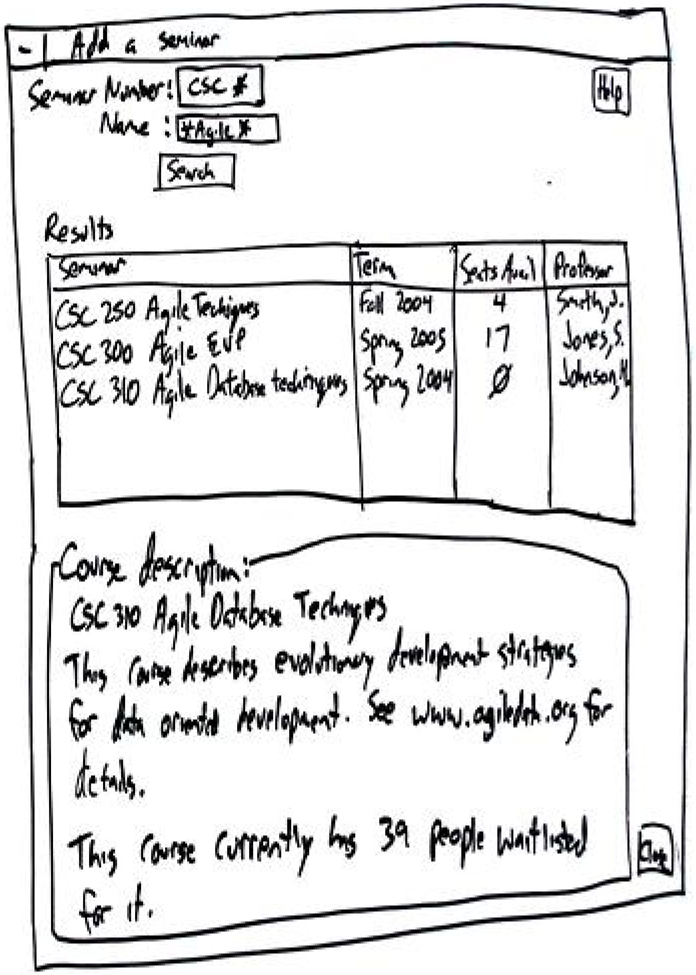
Salutation: Mr.

Date First Enrolled: June 14 2003

Seminars:

Seminar	Term	Mark	Status
CSC 100 Intro to C#	Fall 2003	A+	Passed
CSC 200 Intro to AM	Fall 2003	A	Passed
CSC 203 Advanced AM	Spring 2004	–	Enrolled

Add... Drop... Transcript Close



Add a Seminar

Seminar Number: CSC #

Name: *Agile*

Search

Help

Results

Seminar	Term	Seats Avail	Professor
CSC 250 Agile Techniques	Fall 2004	4	Smith, J.
CSC 300 Agile EUP	Spring 2005	17	Jones, S.
CSC 310 Agile Database techniques	Spring 2004	Ø	Johnson, M.

Course description:

CSC 310 Agile Database Techniques

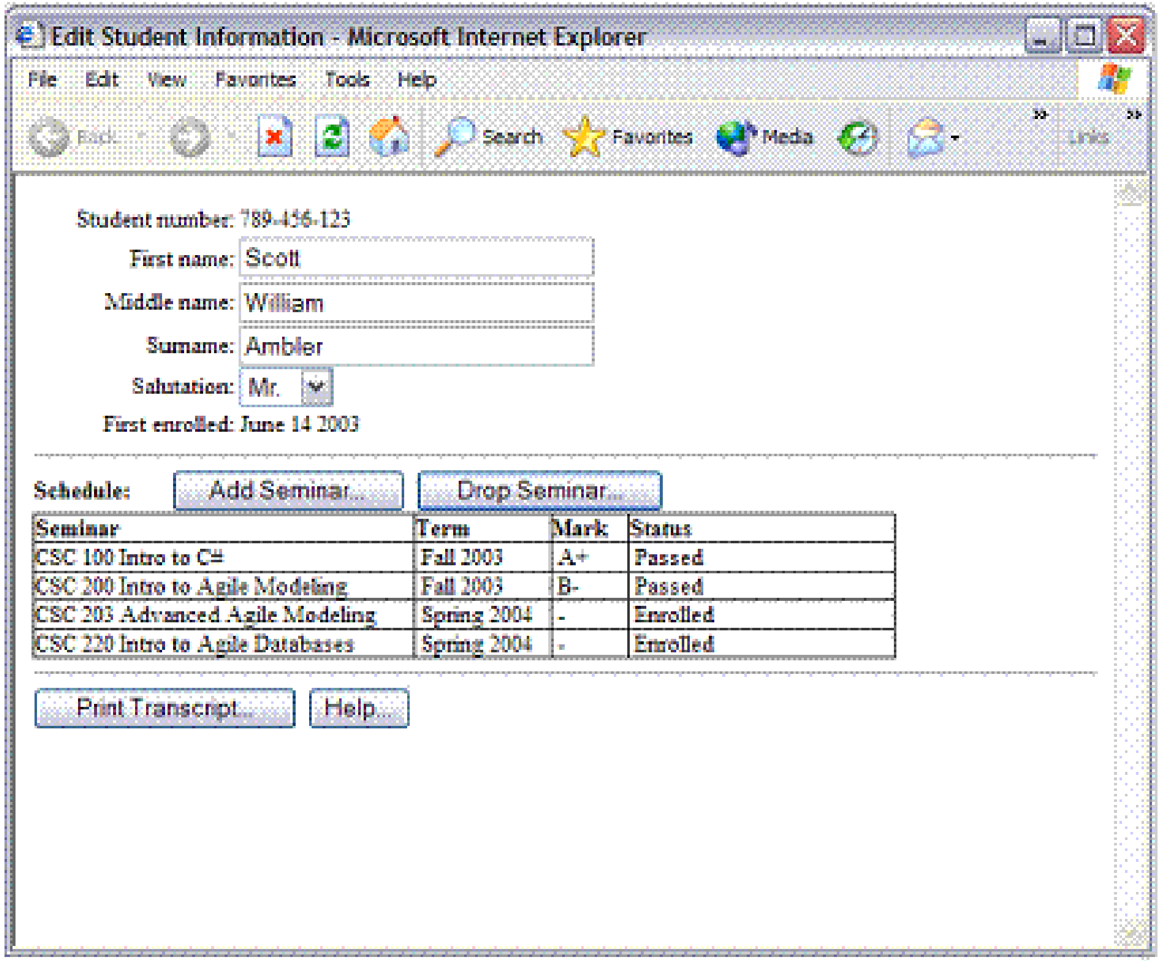
This course describes evolutionary development strategies for data oriented development. See www.agiledata.org for details.

This course currently has 39 people waitlisted for it.

Close

User Interface Model

- For user interface intensive projects consider developing some screen sketches or even a user interface prototype.


Edit Student Information - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media Links

Student number: 789-456-123

First name: Scott

Middle name: William

Surname: Ambler

Salutation: Mr.

First enrolled: June 14 2003

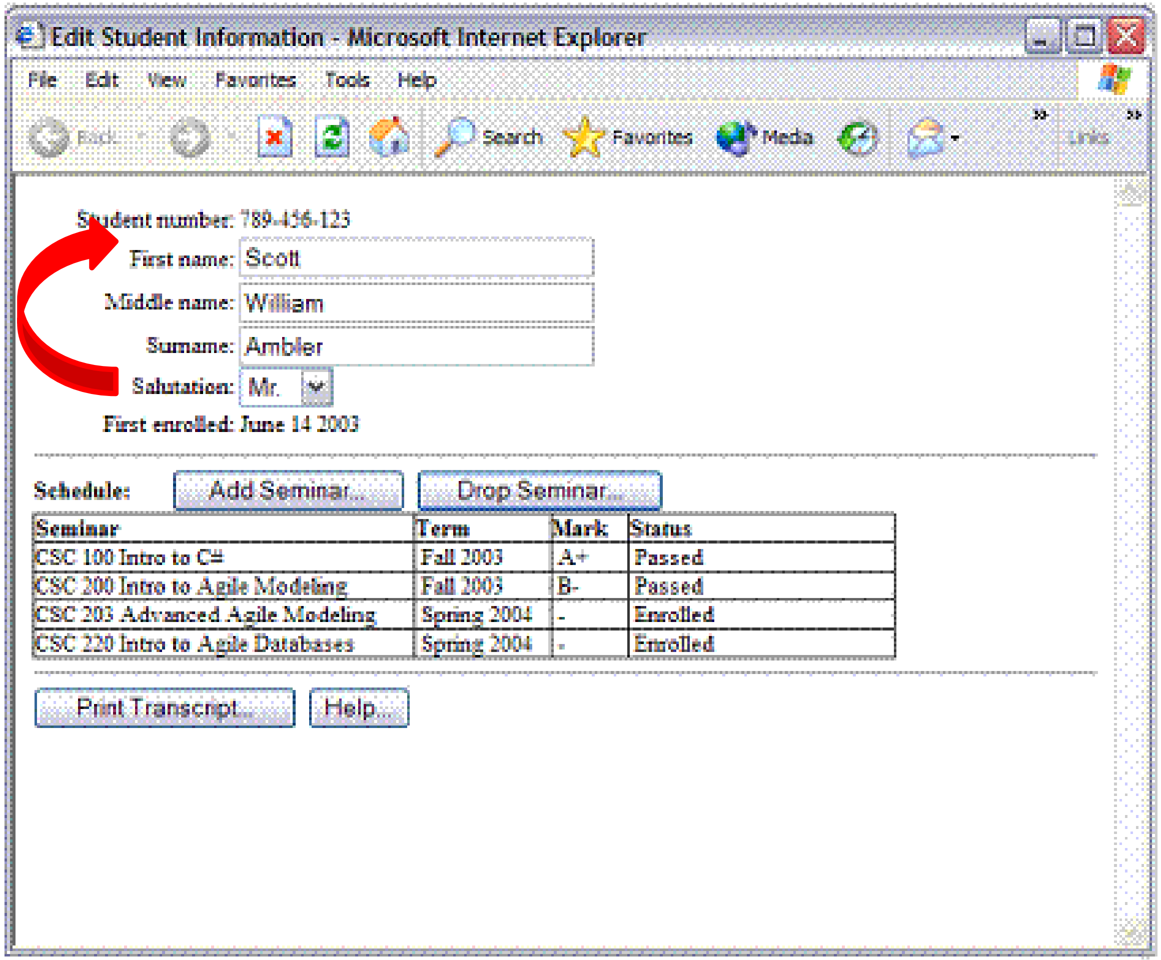
Schedule: Add Seminar... Drop Seminar...

Seminar	Term	Mark	Status
CSC 100 Intro to C#	Fall 2003	A+	Passed
CSC 200 Intro to Agile Modeling	Fall 2003	B-	Passed
CSC 203 Advanced Agile Modeling	Spring 2004	-	Enrolled
CSC 220 Intro to Agile Databases	Spring 2004	-	Enrolled

User Interface Model

- For feature additions to an existing application use a screen shot of the existing system with new features drawn in.

Move salutation above First name



Edit Student Information - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media Links
Add Seminar...
Drop Seminar...
Print Transcript...
Help...

Student number: 789-456-123

First name: Scott

Middle name: William

Surname: Ambler

Salutation: Mr.

First enrolled: June 14 2003

Schedule:

Seminar	Term	Mark	Status
CSC 100 Intro to C#	Fall 2003	A+	Passed
CSC 200 Intro to Agile Modeling	Fall 2003	B-	Passed
CSC 203 Advanced Agile Modeling	Spring 2004	-	Enrolled
CSC 220 Intro to Agile Databases	Spring 2004	-	Enrolled

Any Questions?

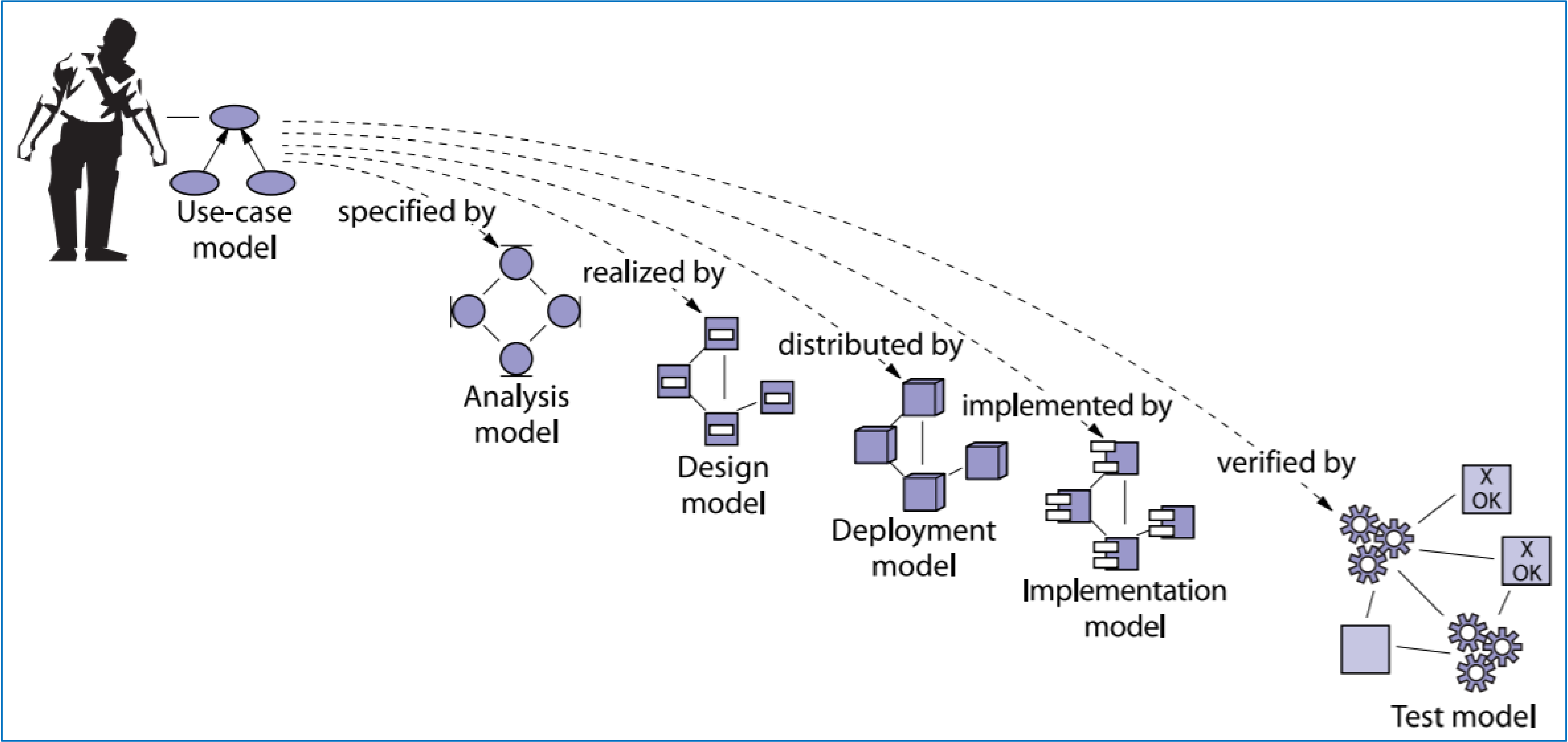
Formal Requirements Artifacts

- You can start off with these informal artifacts (or others like them) and progress to the more formal ones.
- Or you can start with the more formal ones from the Unified Modeling Language (UML).

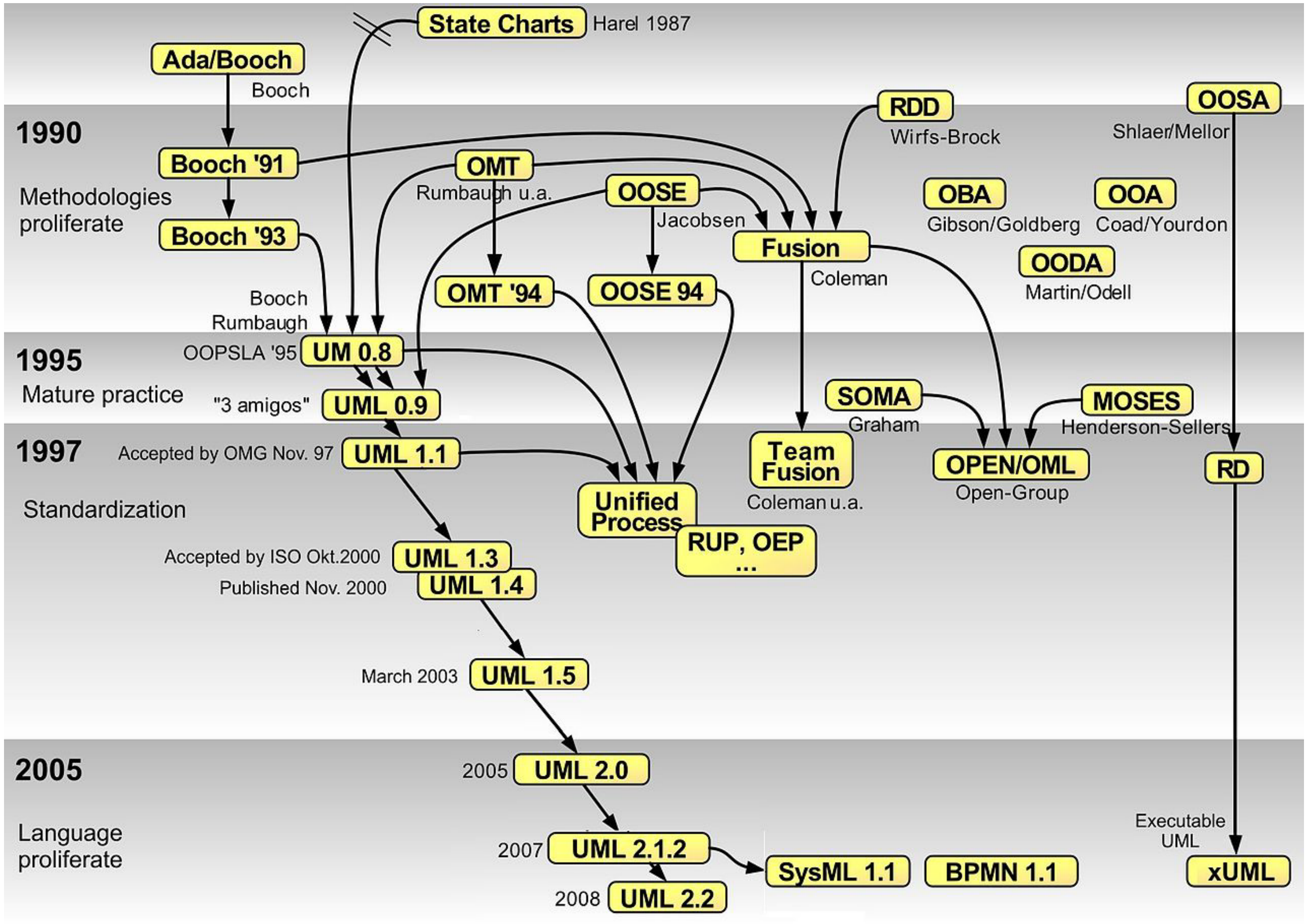
Unified Modeling Language (UML)

Recall from EECS 348:

- A visual language for specification and documentation
- UML models can be mapped into implementation languages (e.g., C++, Java, …)
- SDLC-independent

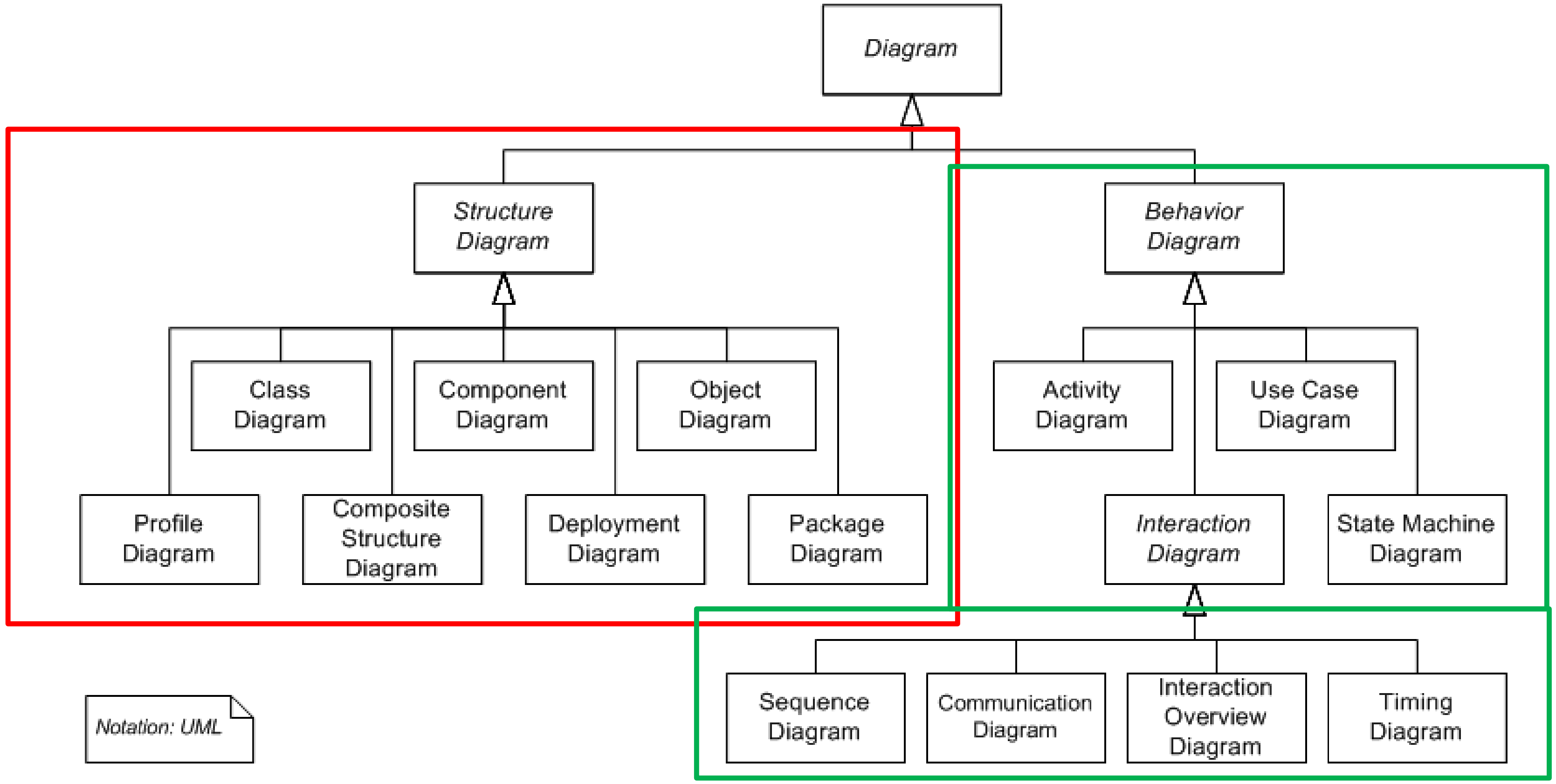


UML History



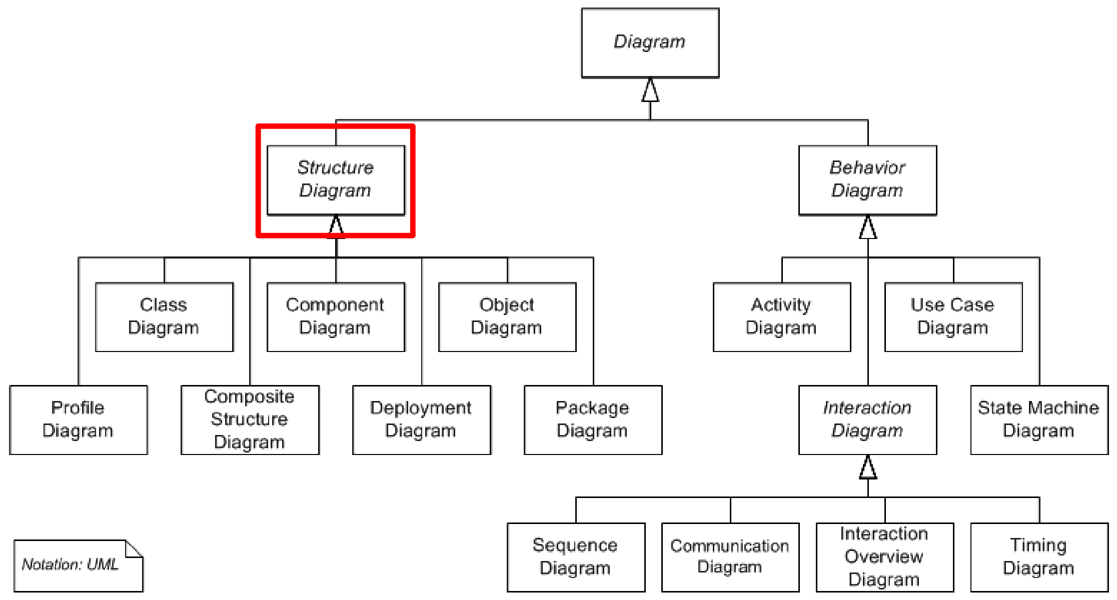
State Charts
Harel 1987
Ada/Booch
Booch
1990
Booch '91
Methodologies proliferate
Booch '93
OMT
Rumbaugh u.a.
OOSE
Jacobsen
RDD
Wirfs-Brock
OOSA
Shlaer/Mellor
OBA
Gibson/Goldberg
OOA
Coad/Yourdon
Fusion
Coleman
OODA
Martin/Odell
OMT '94
OOSE 94
Booch
Rumbaugh
1995
OOPSLA '95
UM 0.8
Mature practice
"3 amigos"
UML 0.9
SOMA
Graham
MOSES
Henderson-Sellers
1997
Accepted by OMG Nov. 97
UML 1.1
Team Fusion
Coleman u.a.
OPEN/OML
Open-Group
RD
Standardization
Unified Process
RUP, OEP ...
Accepted by ISO Okt.2000
UML 1.3
Published Nov. 2000
UML 1.4
March 2003
UML 1.5
2005
2005
UML 2.0
Language proliferate
Executable UML
2007
UML 2.1.2
SysML 1.1
BPMN 1.1
xUML
2008
UML 2.2

Unified Modeling Language (UML)



- UML 2.2 has 14 types of diagrams divided into two categories:
  - Structure (or static)
  - Behavior (or dynamic)

UML Structure Diagrams



Structure diagrams:

Emphasize the things that must be present in the system being modeled.

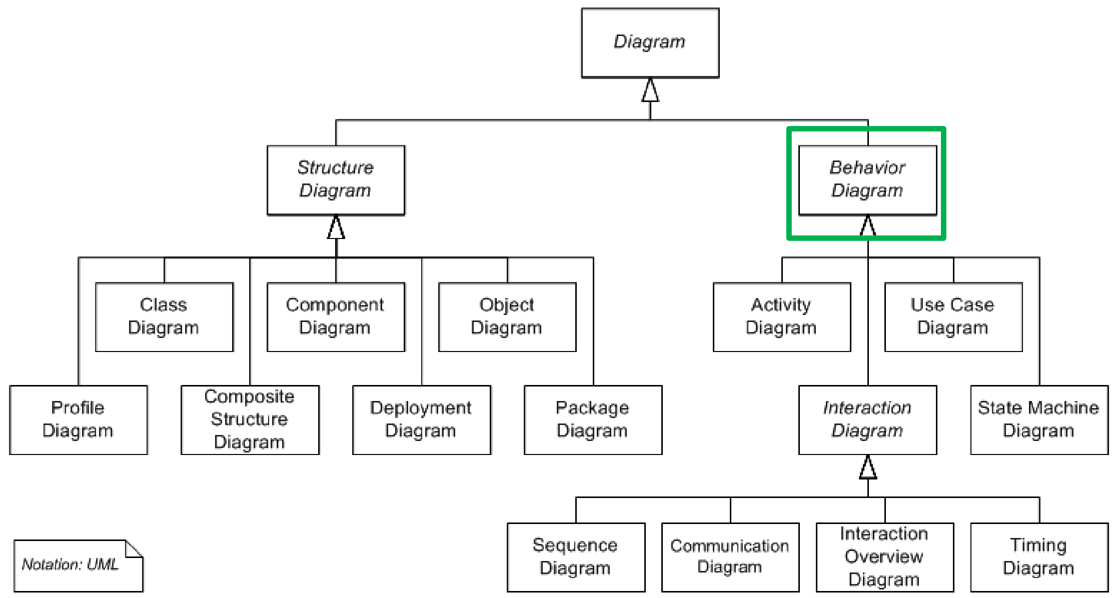
Since structure diagrams represent the structure, they are used extensively in documenting the software architecture and design of software systems.

You did a lab in EECS 348 on the Class Diagram (Design Artifact)

The Deployment Diagram is one recommended for the Initial Architecture Document

They are generally not used as Requirement Artifacts.

UML Behavior Diagrams



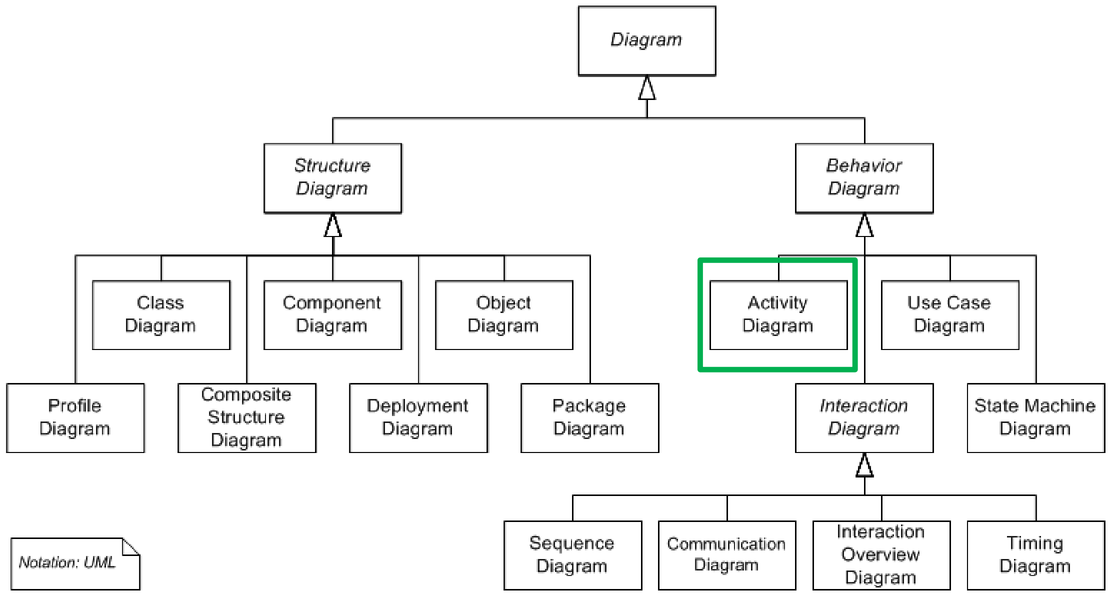
Behavior diagrams:

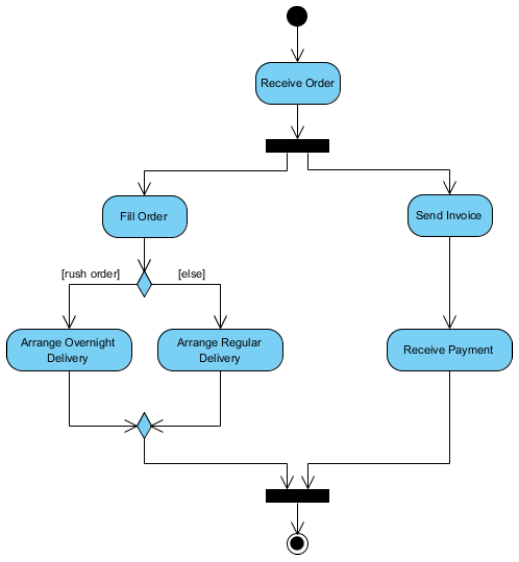
Emphasize what must happen in the system being modeled.

These are used as Requirements Artifacts.

Any Questions?

UML Activity Diagram



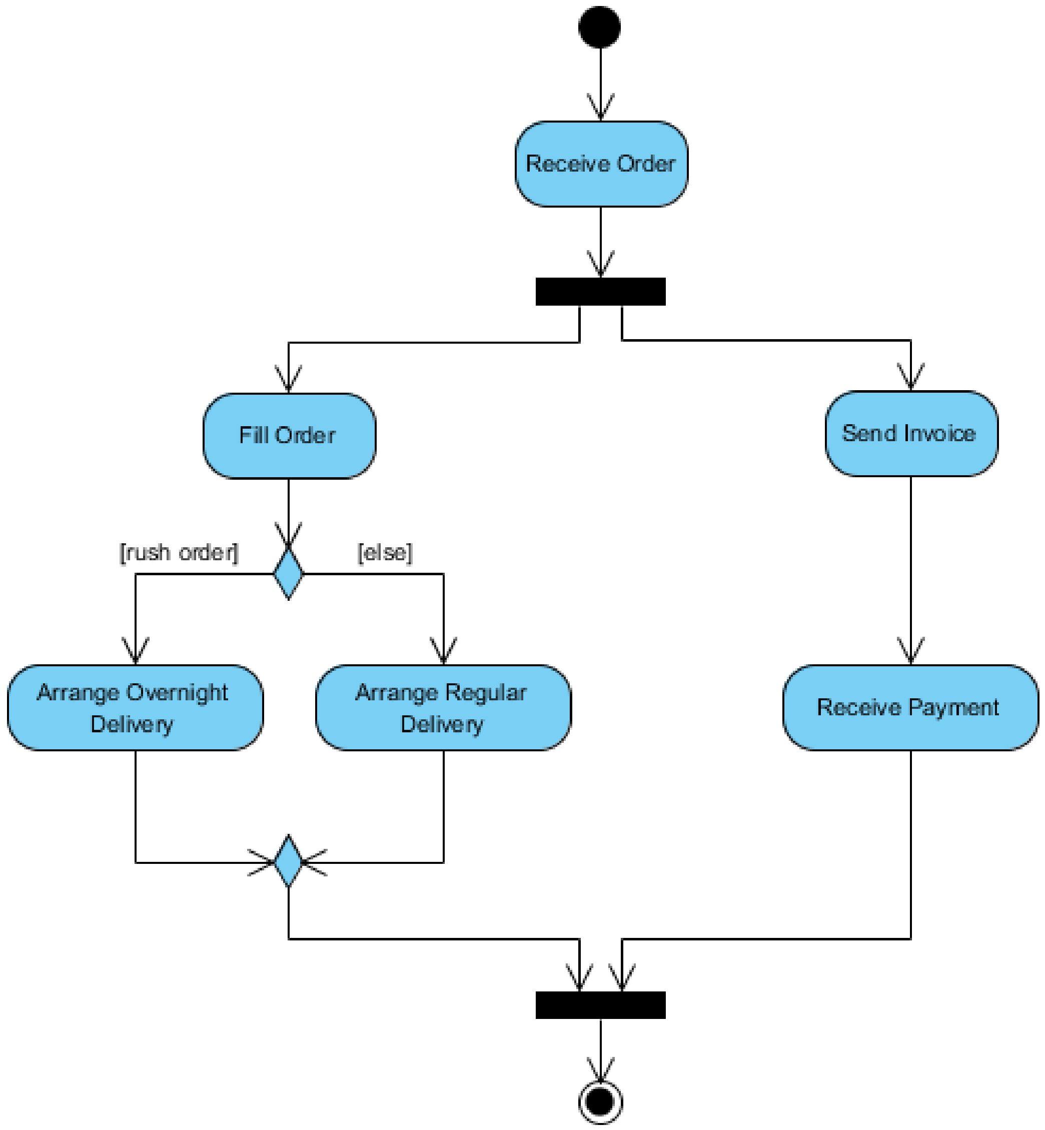


Activity diagram:

Describes the business and operational step-by-step workflows of components in a system.

An activity diagram shows the overall flow of control.

UML Activity Diagram

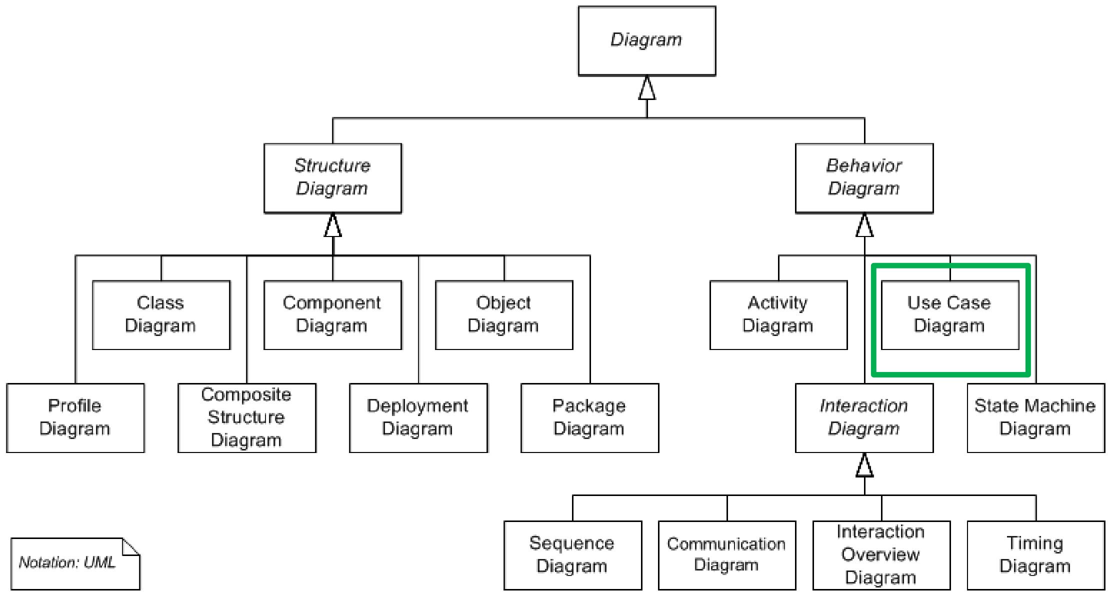


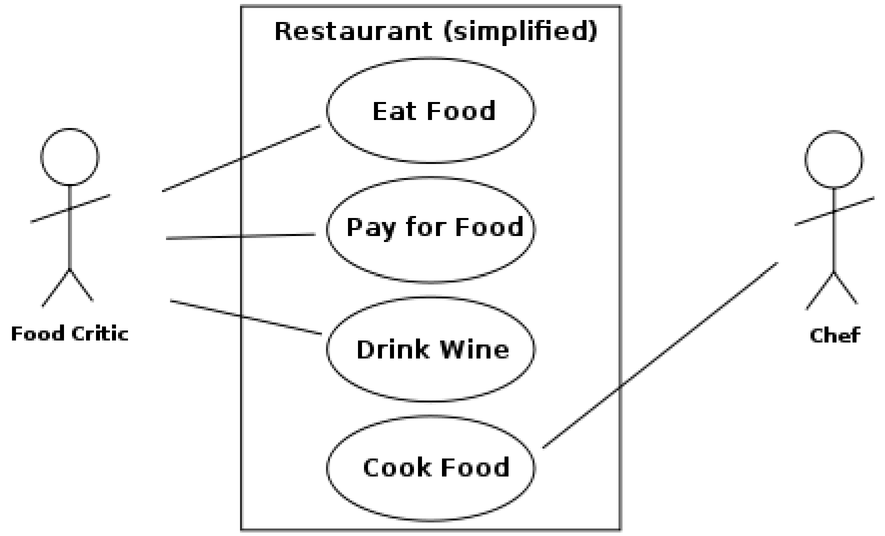
Activity diagram:

Describes the business and operational step-by-step workflows of components in a system.

An activity diagram shows the overall flow of control.

UML Use Case Diagram



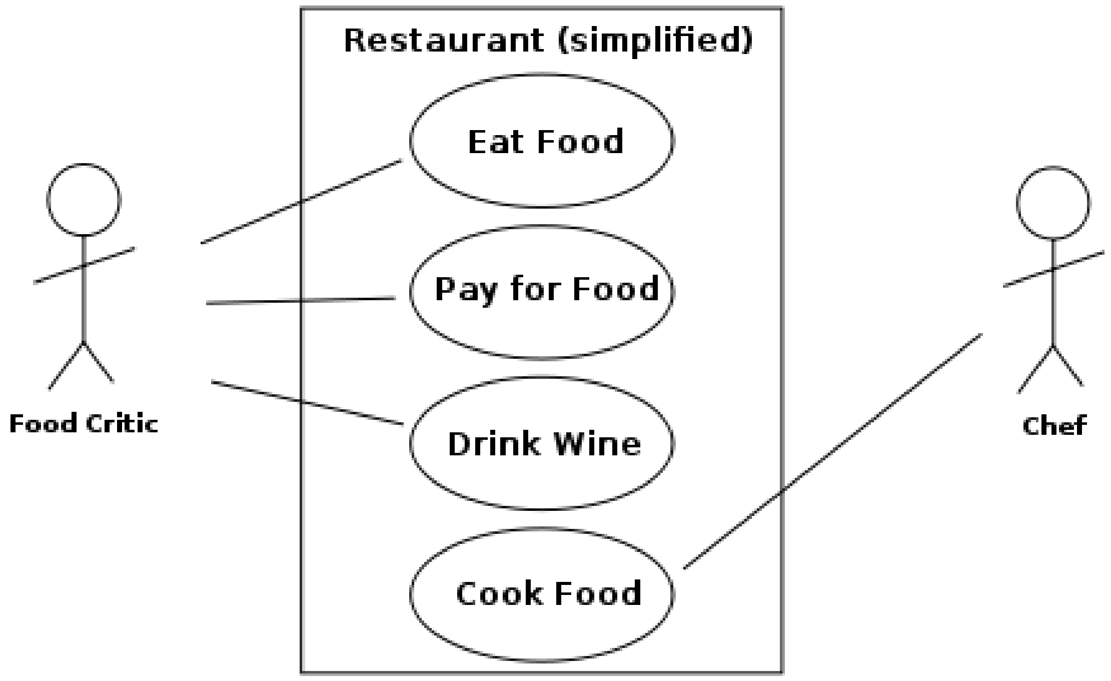


Use case diagram:

Describes the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases.

You did a lab on this one in EECS 348.

UML Use Case Diagram

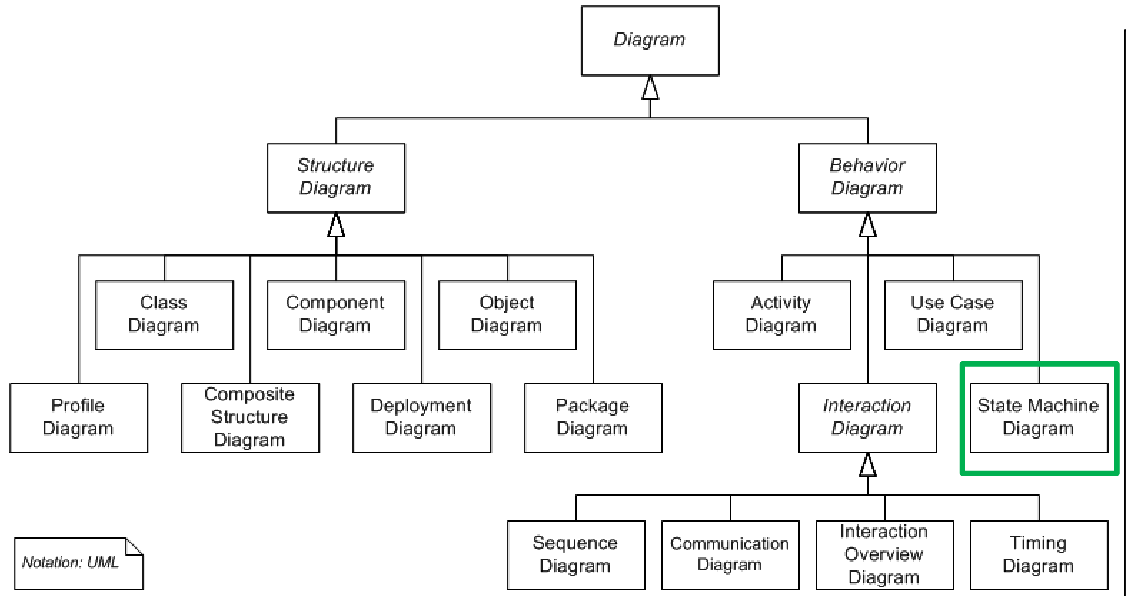


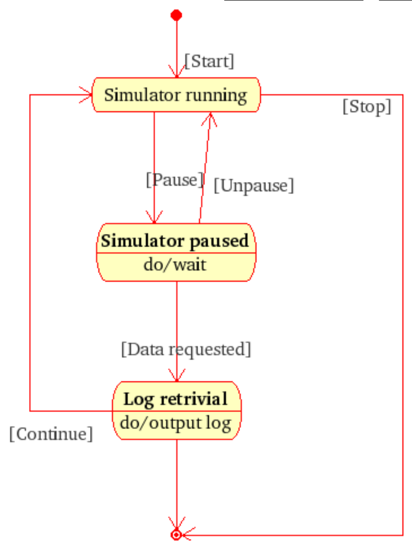
Use case diagram:

Describes the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases.

You did a lab on this one in EECS 348.

UML State Machine Diagram

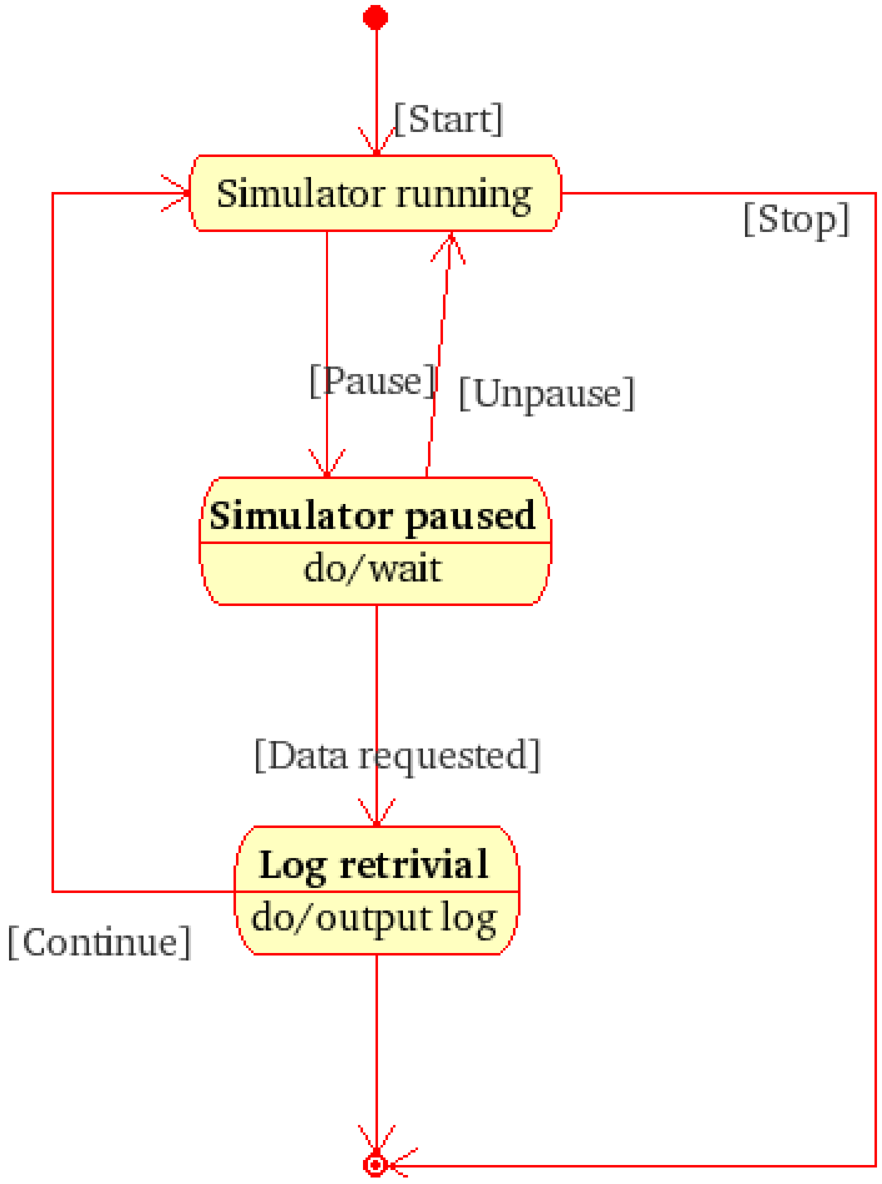




UML state machine diagram:

Describes the states and state transitions of the system.

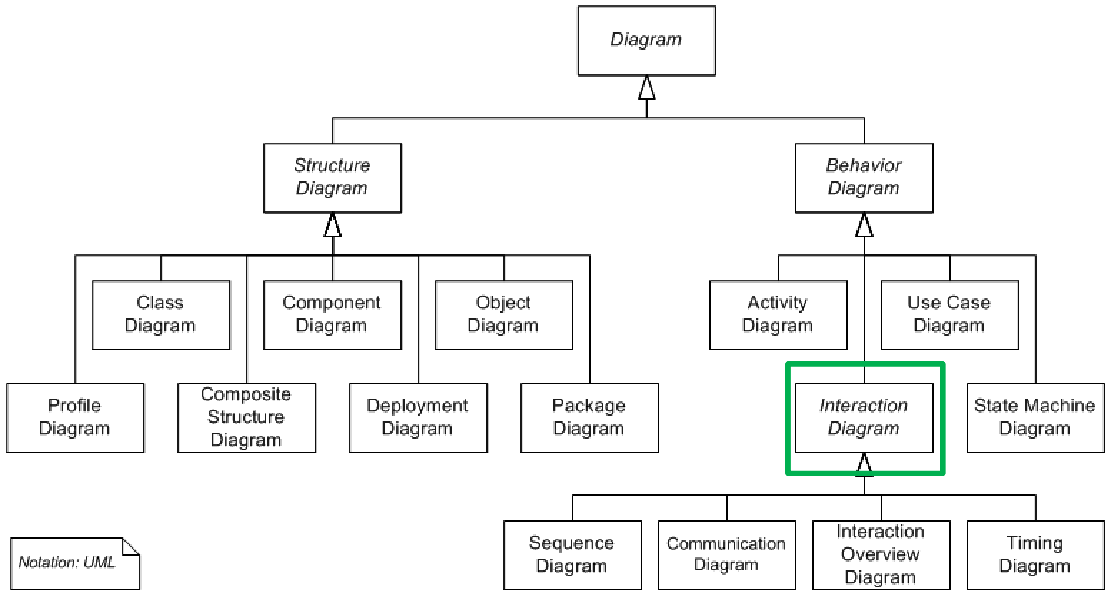
UML State Machine Diagram



UML state machine diagram:

Describes the states and state transitions of the system.

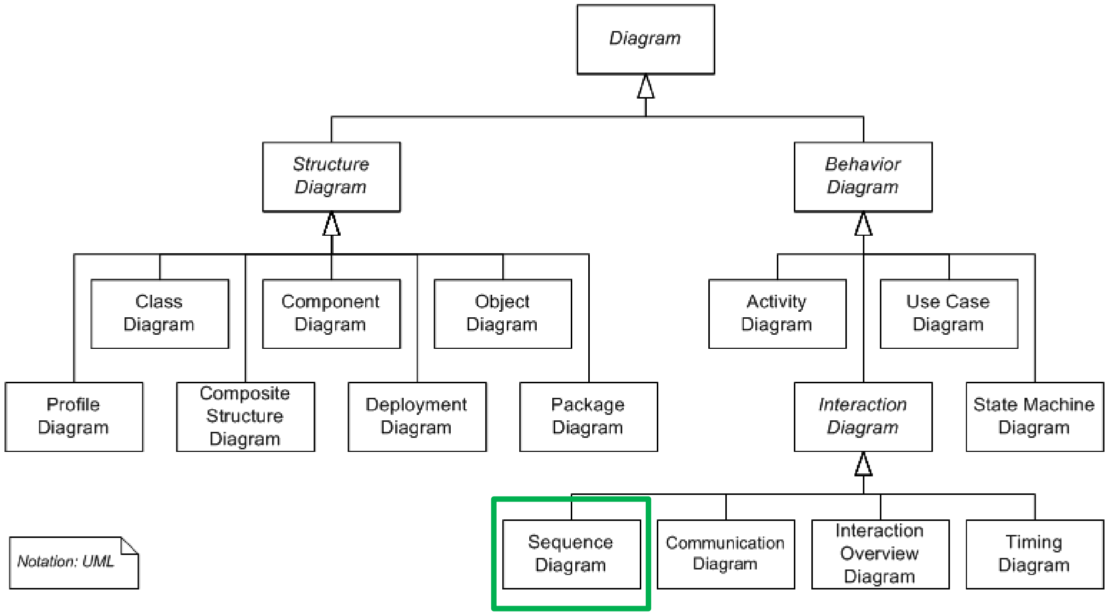
UML Interaction Diagrams

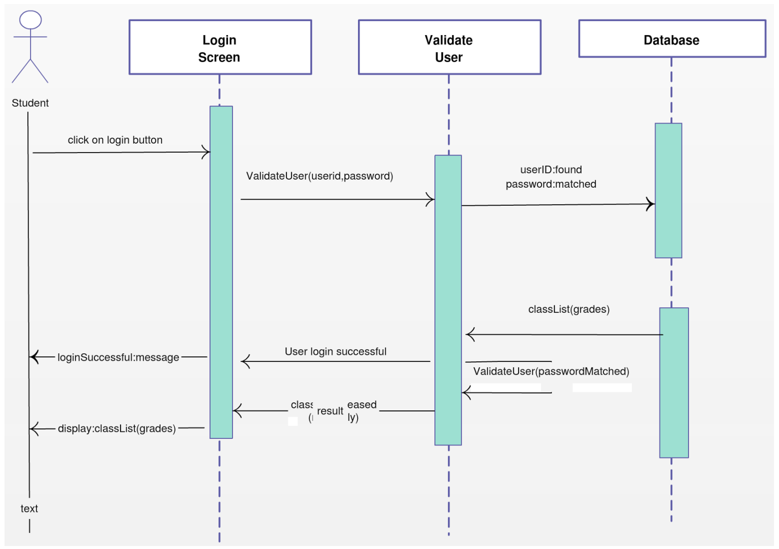


Interaction diagrams:

A subset of behavior diagrams that emphasize the flow of control and data among the things in the system being modeled.

UML Sequence Diagram



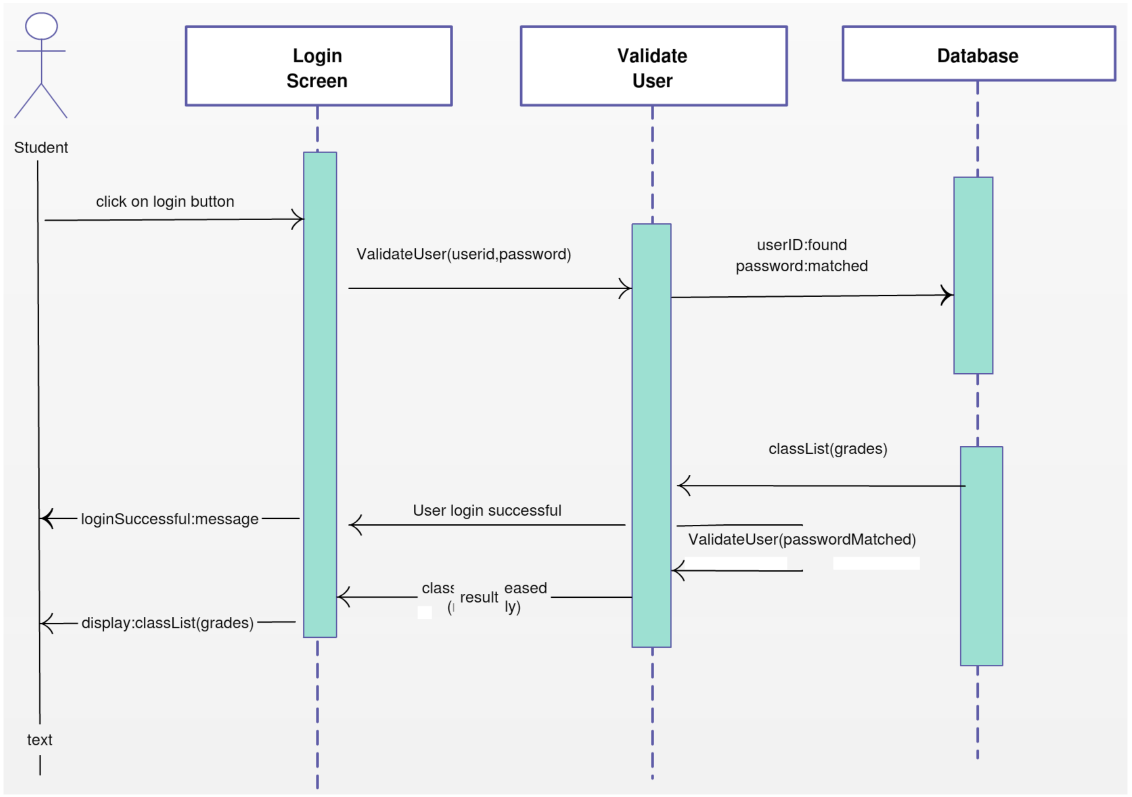


Sequence diagram:

Shows how objects communicate with each other in terms of a sequence of messages.

Also indicates the life spans of objects relative to those messages.

UML Sequence Diagram

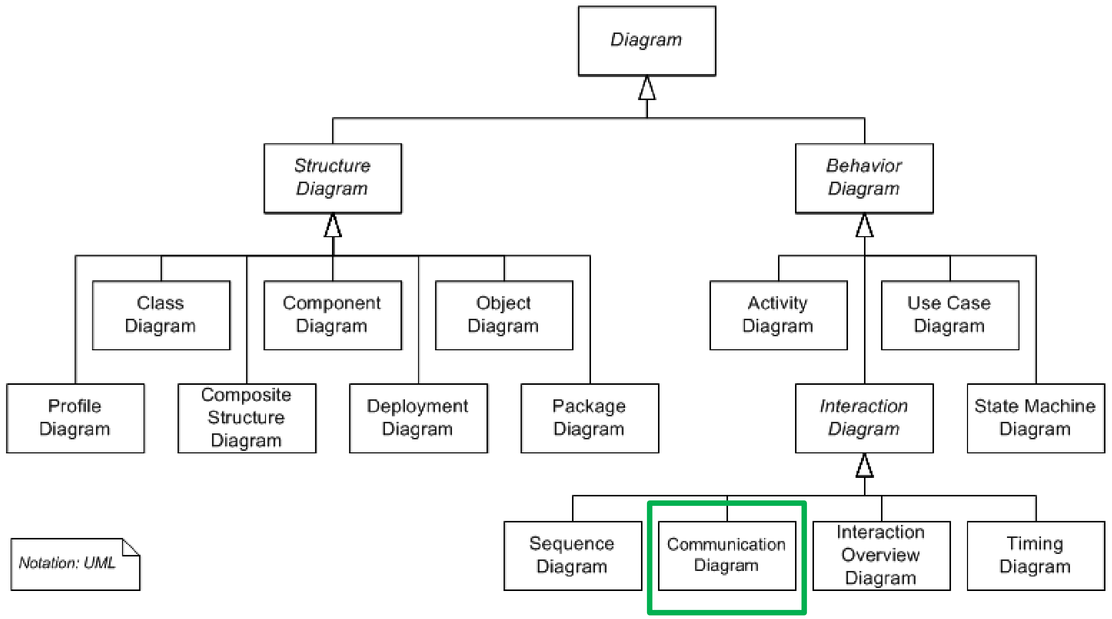

Student
Login Screen
Validate User
Database
click on login button
ValidateUser(userid,password)
userID:found password:matched
classList(grades)
loginSuccessful:message
User login successful
ValidateUser(passwordMatched)
class result eased (l ly)
display:classList(grades)
text

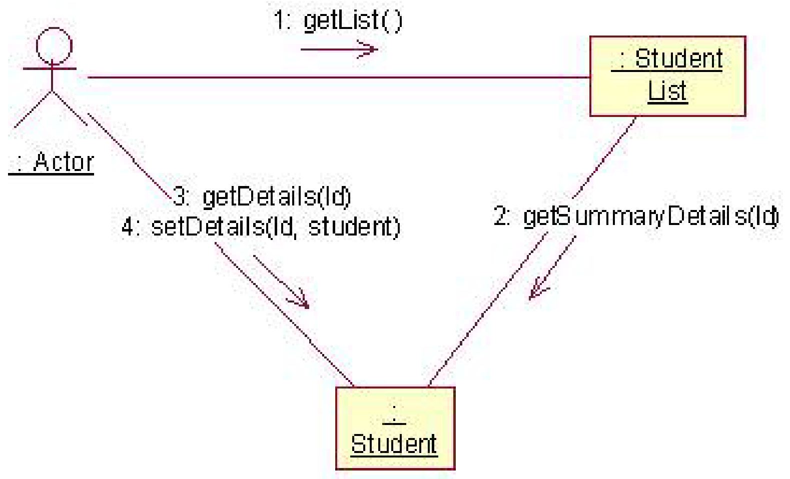
Sequence diagram:

Shows how objects communicate with each other in terms of a sequence of messages.

Also indicates the life spans of objects relative to those messages.

UML Communication Diagram

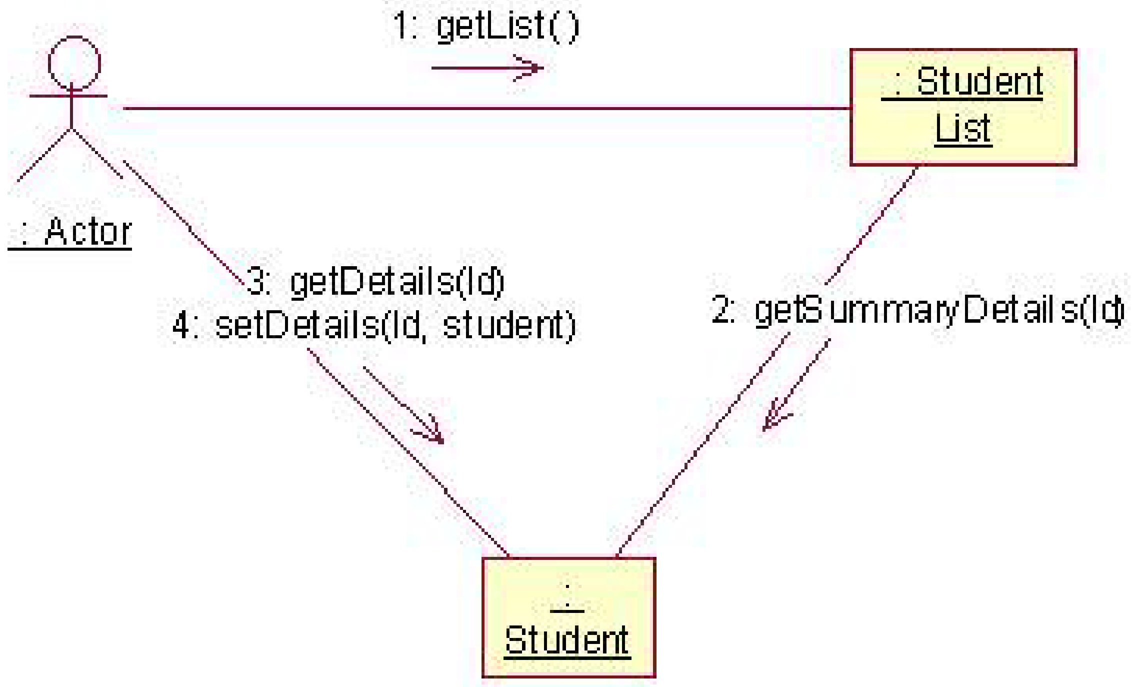




Communication diagram:

Shows the interactions between objects or parts in terms of sequenced messages.

UML Communication Diagram

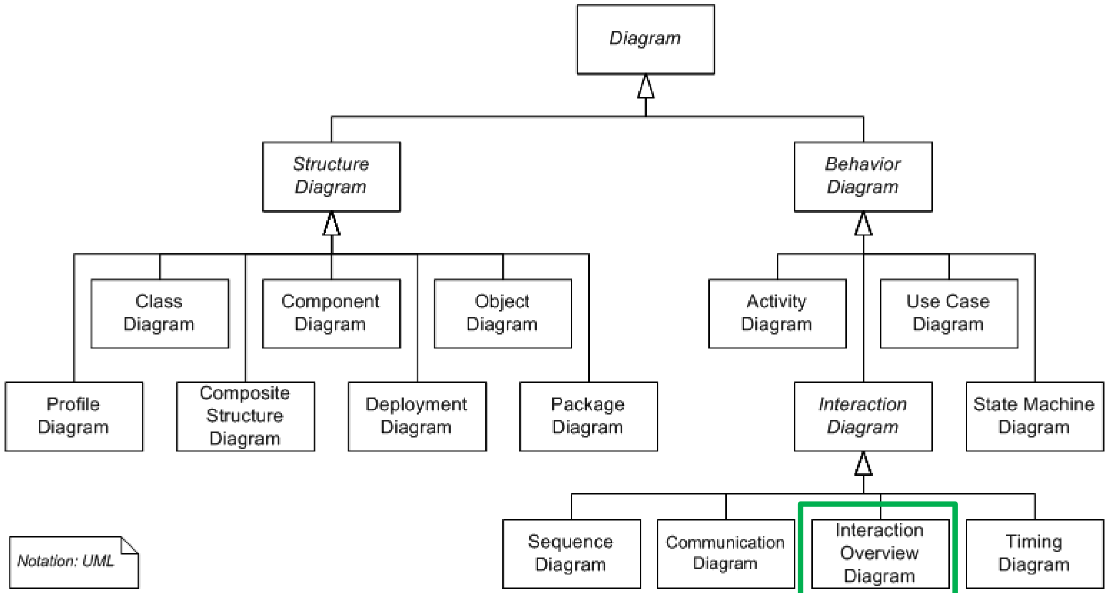


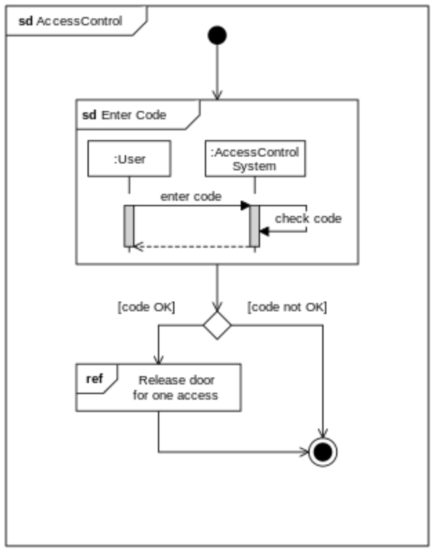
Communication diagram:

Shows the interactions between objects or parts in terms of sequenced messages.

Any Questions?

UML Interaction Overview Diagram

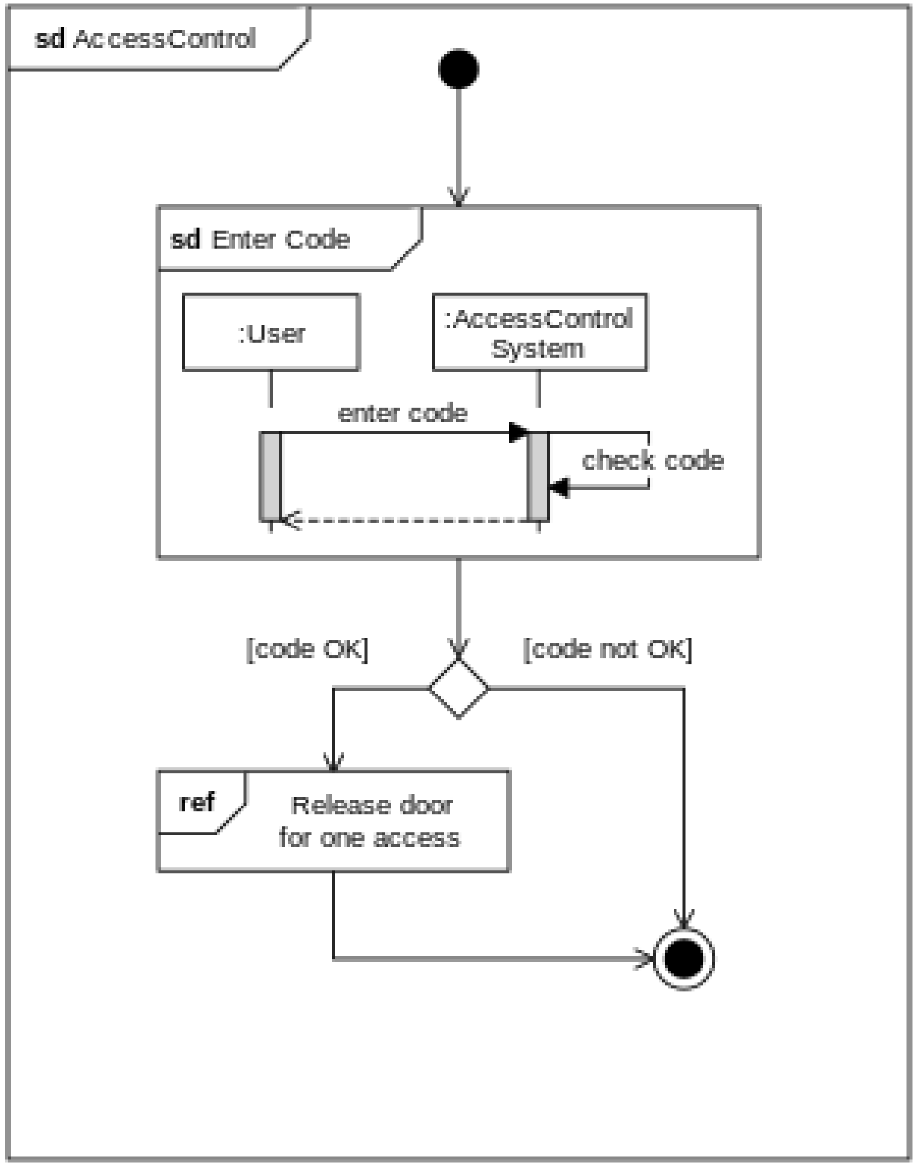




Interaction overview diagram:

Provides an overview in which the nodes represent Communication Diagrams.

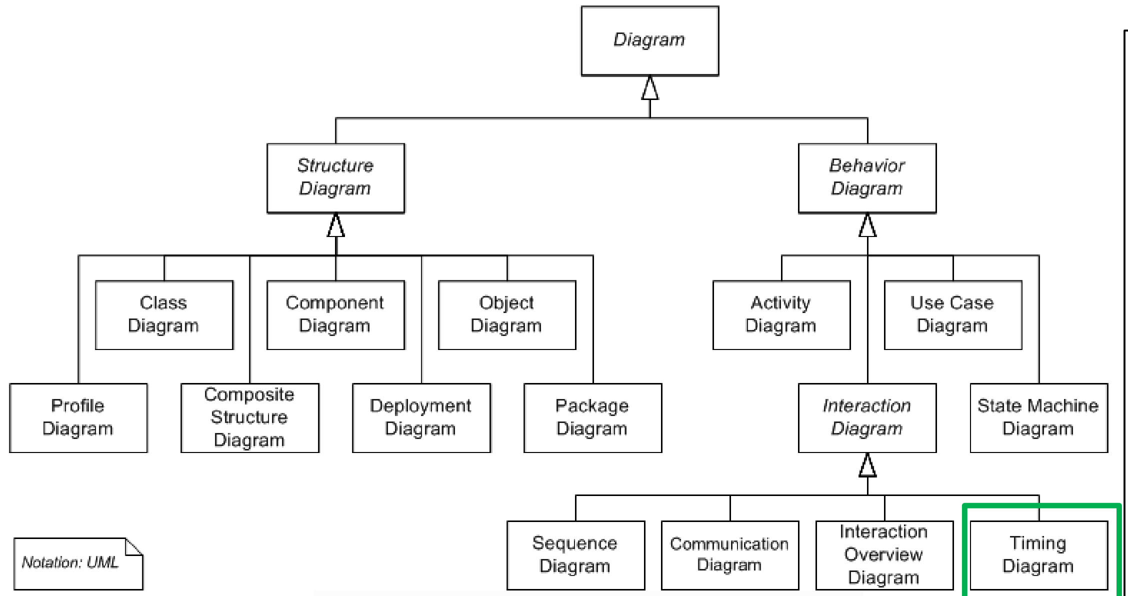
UML Interaction Overview Diagram

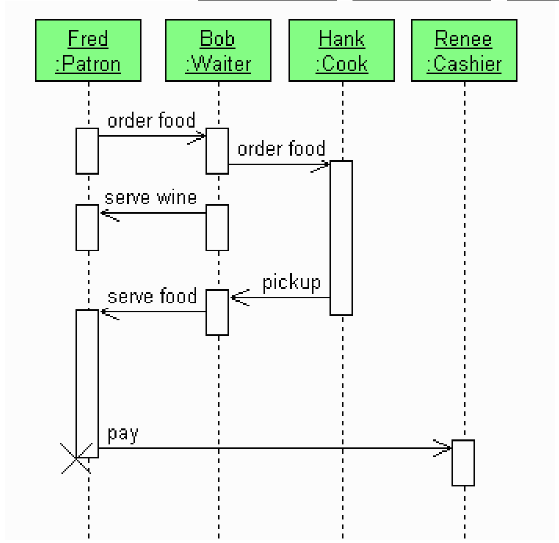


Interaction overview diagram:

Provides an overview in which the nodes represent Communication Diagrams.

UML Timing Diagram

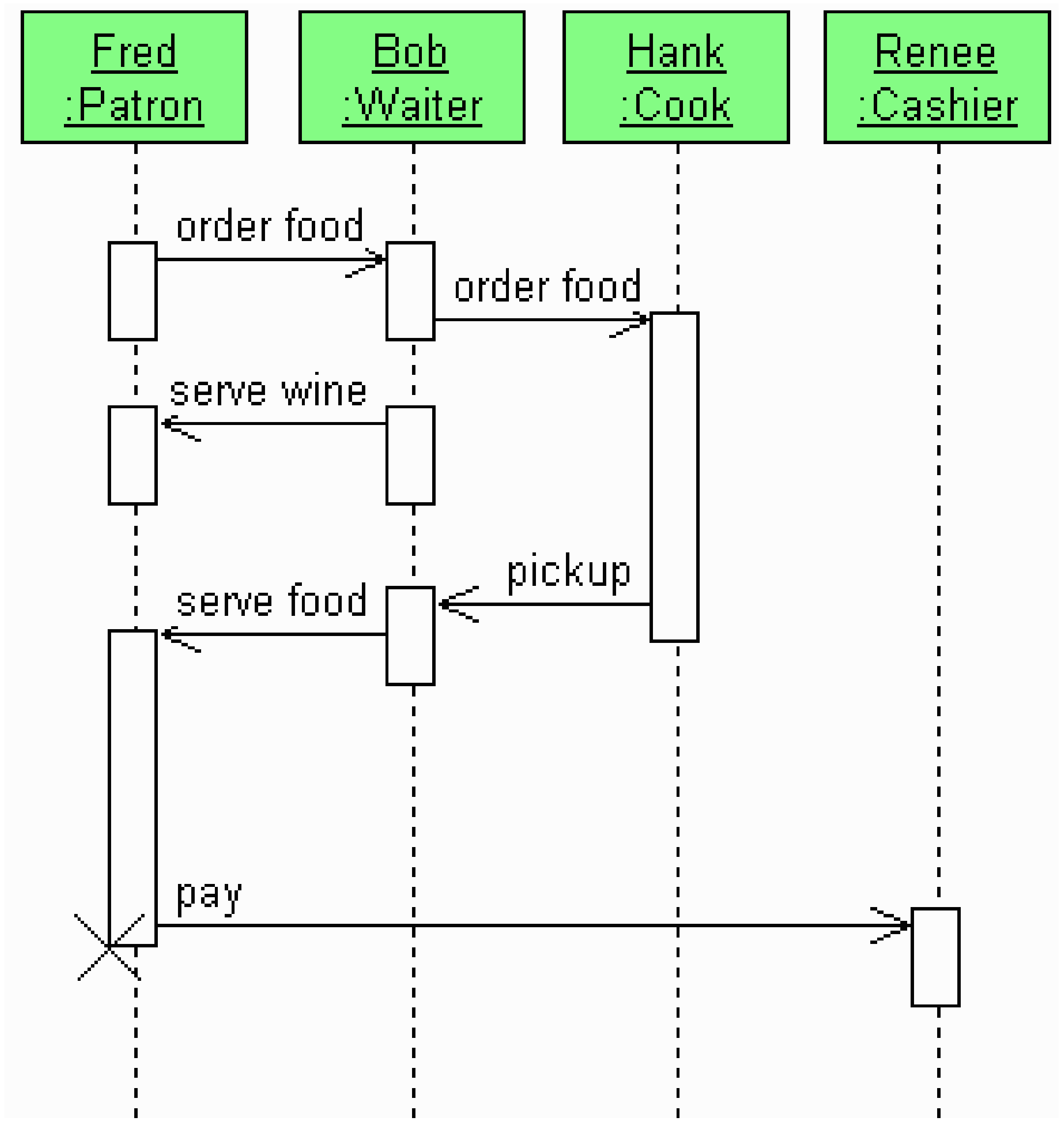




Timing diagrams:

A specific type of interaction diagram where the focus is on timing constraints.

UML Timing Diagram



Timing diagrams:

A specific type of interaction diagram where the focus is on timing constraints.

Any Questions?

UML Diagram Similarities

- As you can see there are a lot of similarities and overlap between the various diagrams.
- For example:
  - Use Case diagram can become Sequence diagram by adding timing information
  - Activity diagram can become State diagram by adding timing information

- Some of this is probably due to the fact that UML was developed by a committee!

UML Open Source Drawing Tools

- Visual Paradigm (visual-paradigm.com)
- Rational Software Modeler (ibm.com)
- Altova (altova.com)
- ArgoUML (argouml.tigris.org)
- Creately for UML (creately.com)
- MagicDraw (nomagic.com)
- Microsoft Visio (Microsoft Visual Studio)

Any Questions?

In-Class Problem Rubric

Question	Points	Grading Level		
		Exceeds Expectations (90-100%)	Meets Expectations (80-89%)	Unsatisfactory (0-79%)
1	80	Requirements Artifact is consistent with requirement specified and is concise, descriptive, and unique	Requirements Artifact is consistent with requirement specified, but is not concise, generic, or not unique	Otherwise
2	10	100% if one of the Requirements Artifacts is identified 0% otherwise		
3	10	100% if exact wording of requirement being modeled is given 0% otherwise		

In-Class Problem

1. Use one of the informal or formal Requirements Artifacts to model one of the requirements from your Requirements Stack you developed for the “Agile Release Planning” lecture In-Class Problem.
2. Identify the Requirements Artifact you are using.
3. Give the exact wording of the requirement you are modeling.

Note:

- This should be very individual. No two students should have the same answers.