Student Online Teaching Advice Notice

The materials and content presented within this session are intended solely for use in a context of teaching and learning at Trinity.

Any session recorded for subsequent review is made available solely for the purpose of enhancing student learning.

Students should not edit or modify the recording in any way, nor disseminate it for use outside of a context of teaching and learning at Trinity.

Please be mindful of your physical environment and conscious of what may be captured by the device camera and microphone during videoconferencing calls.

Recorded materials will be handled in compliance with Trinity's statutory duties under the Universities Act, 1997 and in accordance with the University's policies and procedures.

Further information on data protection and best practice when using videoconferencing software is available at https://www.tcd.ie/info_compliance/data-protection/.

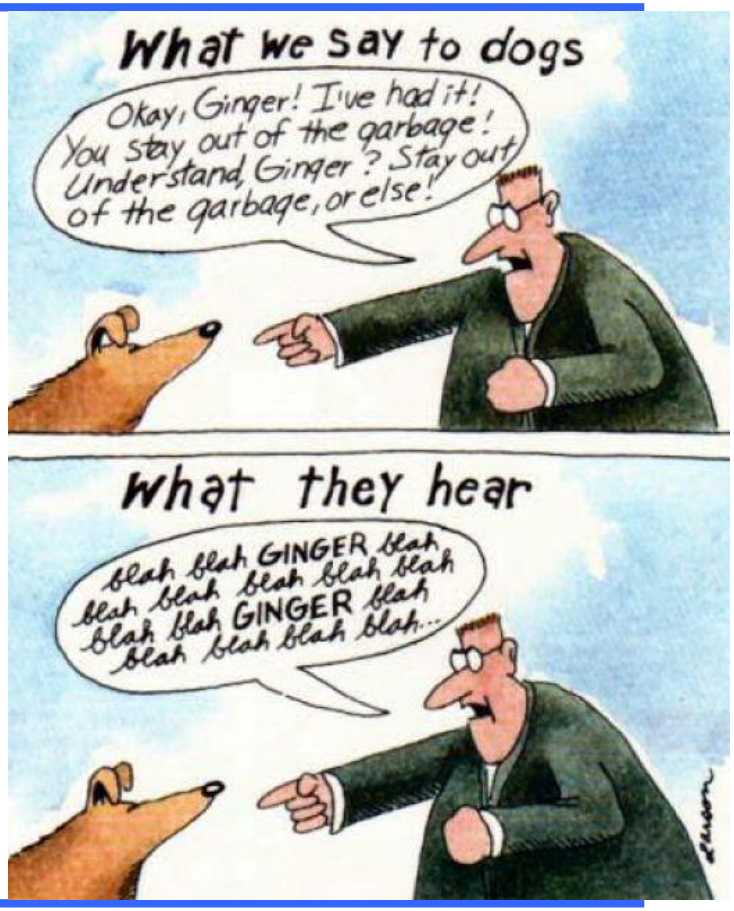
University of Dublin
Trinity College

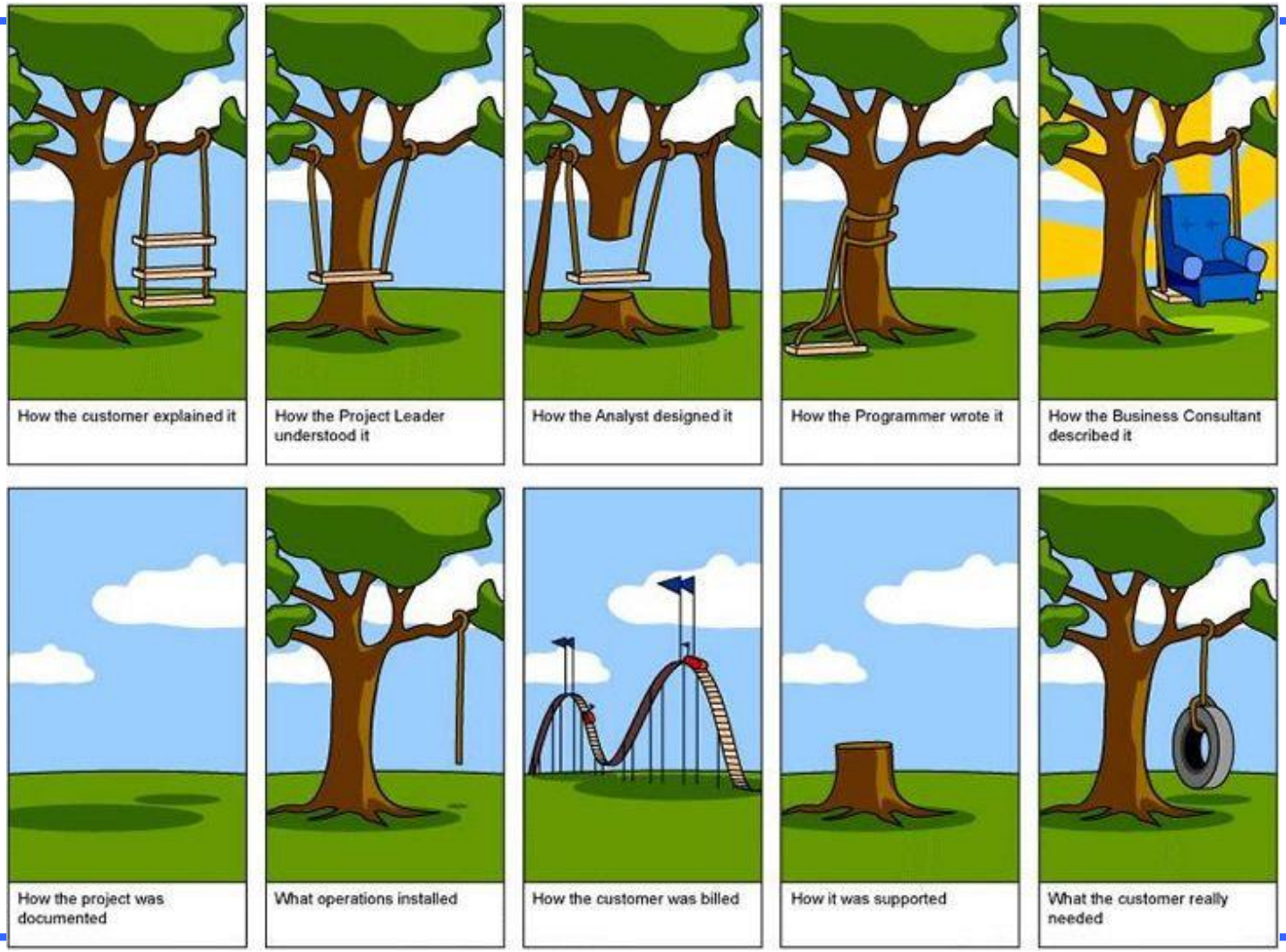
Information Modeling

Using

The Unified Modelling Language (UML)

… the art of communication of the design of information..




How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Why Use a Model?

A model is quicker and easier to build

A model can be used in a simulation

A model can evolve as we learn

We can choose which details to include in a model

A model can represent real or imaginary things from any domain using any materials

How Engineering Models are Used

1. To help us *understand* complex systems
   - Useful for both *requirements* and *designs*
   - Minimize risk by detecting errors and omissions early in the design cycle (at low cost)
     - *Through analysis and experimentation*
     - *Investigate and compare alternative solutions*
   - To communicate understanding
     - *Stakeholders: Clients, users, implementers, testers, documenters, etc.*
2. To *drive implementation*
   - The model as a blueprint for construction

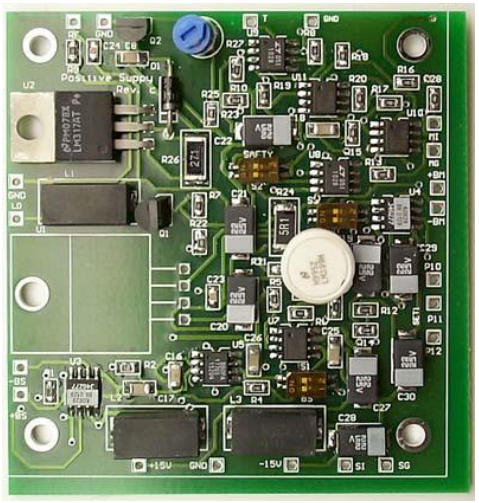
Reducing Risk

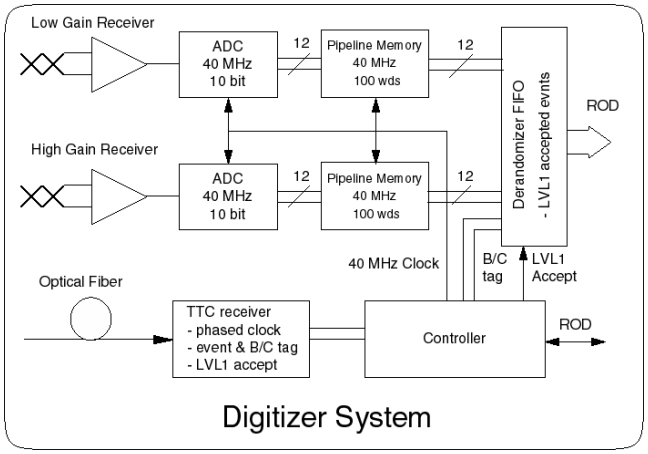
Traditional means of reducing engineering risk

Charles Parsons model steam turbine, 1880, Parsons Building TCD

Perspectives

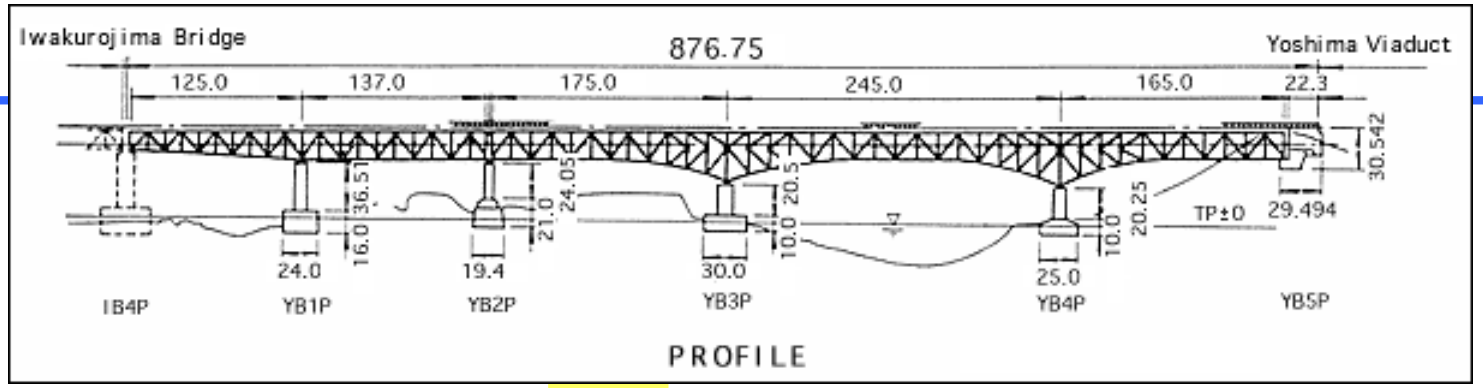
Engineering model: A reduced representation of some system that highlights the properties of interest from a given perspective

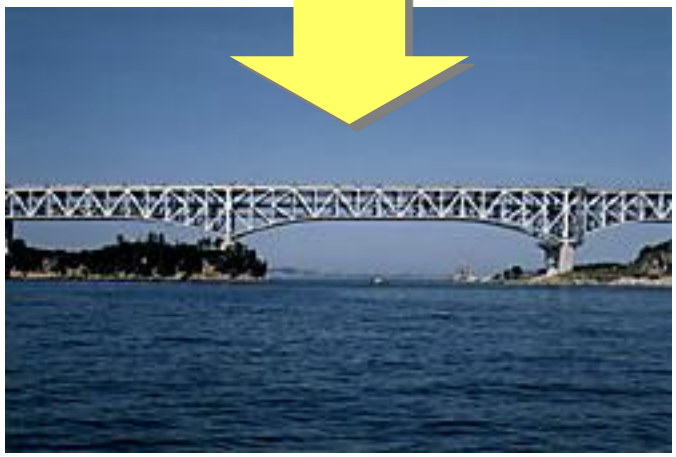




- We don’t see everything at once
- We use a representation (notation) that is easily understood

A Common Problem with Engineering Models





***Semantic Gap* due to:**

- **Idiosyncrasies of actual construction materials**
- **Construction methods**
- **Scaling effects**
- **Skill sets**
- **Misunderstandings**

Can lead to serious errors and discrepancies in the realization

Affordance- Understanding what works

Changing Contexts……

Scope Creep……


I ADDED ALL OF THE PRODUCT FEATURES THAT EACH OF YOU DEMANDED.
PRODUCT
NOW OUR PRODUCT IS A WORTHLESS HODGEPODGE OF COMPLEXITY.
I APPRECIATE YOUR INPUT. I COULDN'T HAVE FAILED WITHOUT YOU.
TEAM-WORK!
DilbertCartoonist@gmail.com
Dilbert.com

Modeling

Models are **abstractions** of the real world

Models are always less detailed than the real world

Many models can be created of any given physical object depending upon the level of detail and point of view selected- Think of maps.

There is no such thing as the most "correct" model; models are simply better or worse suited to accomplishing a particular task

If systems are to be modeled – care must be taken to consider all components. The behaviour of the systems is not the same as the sum of the behaviour of the system as a whole.

- http://courses.cs.vt.edu/professionalism/Therac_25/Therac_1.html- check out lessons learnt at the end of the article.

Models and Systems

Modeling is required to **appropriately** automate Systems

systems have behaviour which distinguishes them from other systems and from its environment

- behaviour cannot generally be predicted by an examination of individual components.

systems are embedded in an environment

- e.g Antibiotics and “superbugs”

systems have internal structure

- amount of internal detail is limitless e.g. description of cardiovascualr system could descend to the cellular level

Why use Diagrams in modelling?

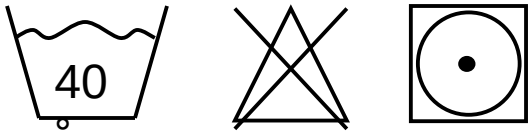
In software engineering, like in other engineering disciplines, we use diagrams to represent models …

Abstract shapes are used to represent things or actions from the real world, or from a system in our case …

Diagrams follow rules or standards

The standards make sure that different people will interpret the diagram in the same way

The intention is to and reduce ambiguity



Information Model

An information model in software engineering is a representation of concepts, relationships, constraints, rules, and operations to specify data semantics for a chosen domain of discourse.

It can provide information requirements for the domain context

Typically Technology Neutral

Most approaches to modelling have a methodology for mapping to concrete information/data implementations (e.g. relational database, XML etc.)

An **Information Model** provides complete set of views on the information for use by different stakeholders

A model may consist of a single diagram, but most consist of a set of diagrams and supporting data and documentation

Information Model Perspectives Views

Typical views are

- Structural/Static view– how data and information is associated, what tasks does the information support etc.

- Dynamic behaviour view

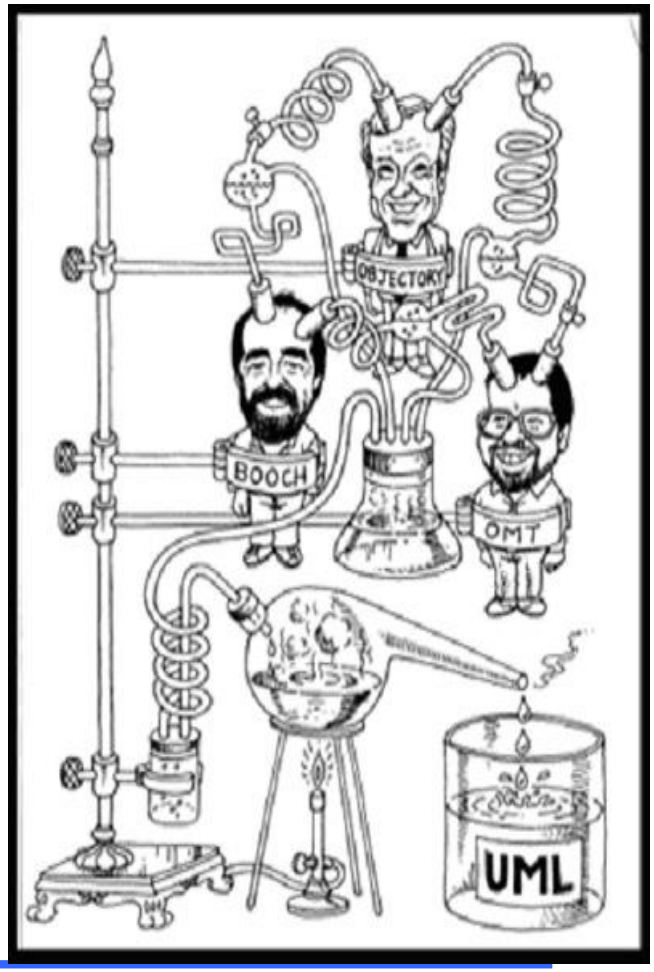
Origins of UML

Object-oriented Analysis and Design techniques

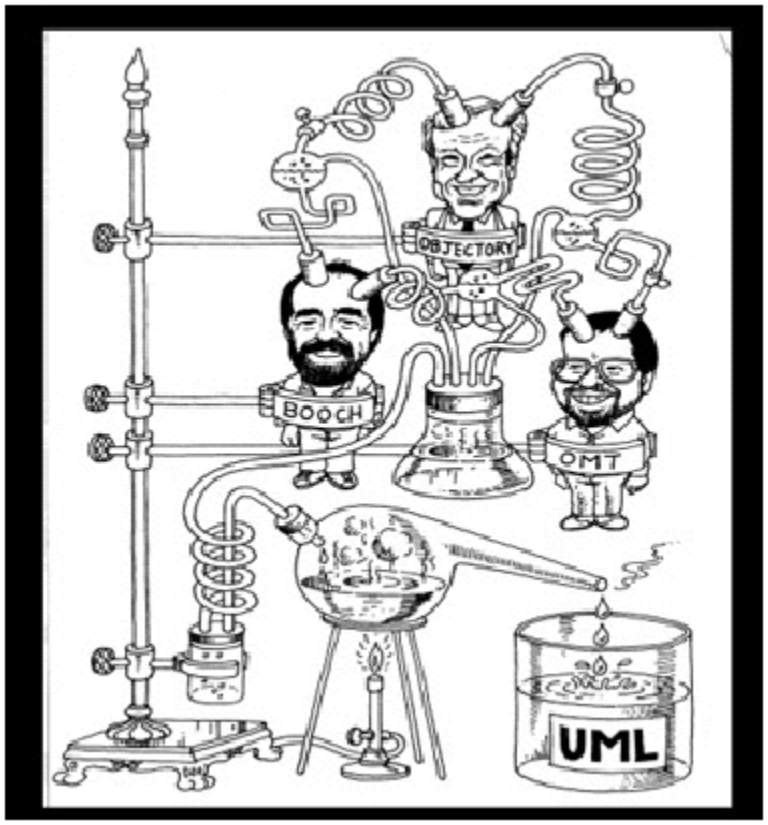
UML's goal is to bring together the best features of all notations to produce an industry standard

Managed by the Open Management Group

- http://www.omg.org/gettingstarted/what_is_uml.htm
- current version 2.5.1 (Dec 2017)
  - https://www.omg.org/spec/UML/2.5.1

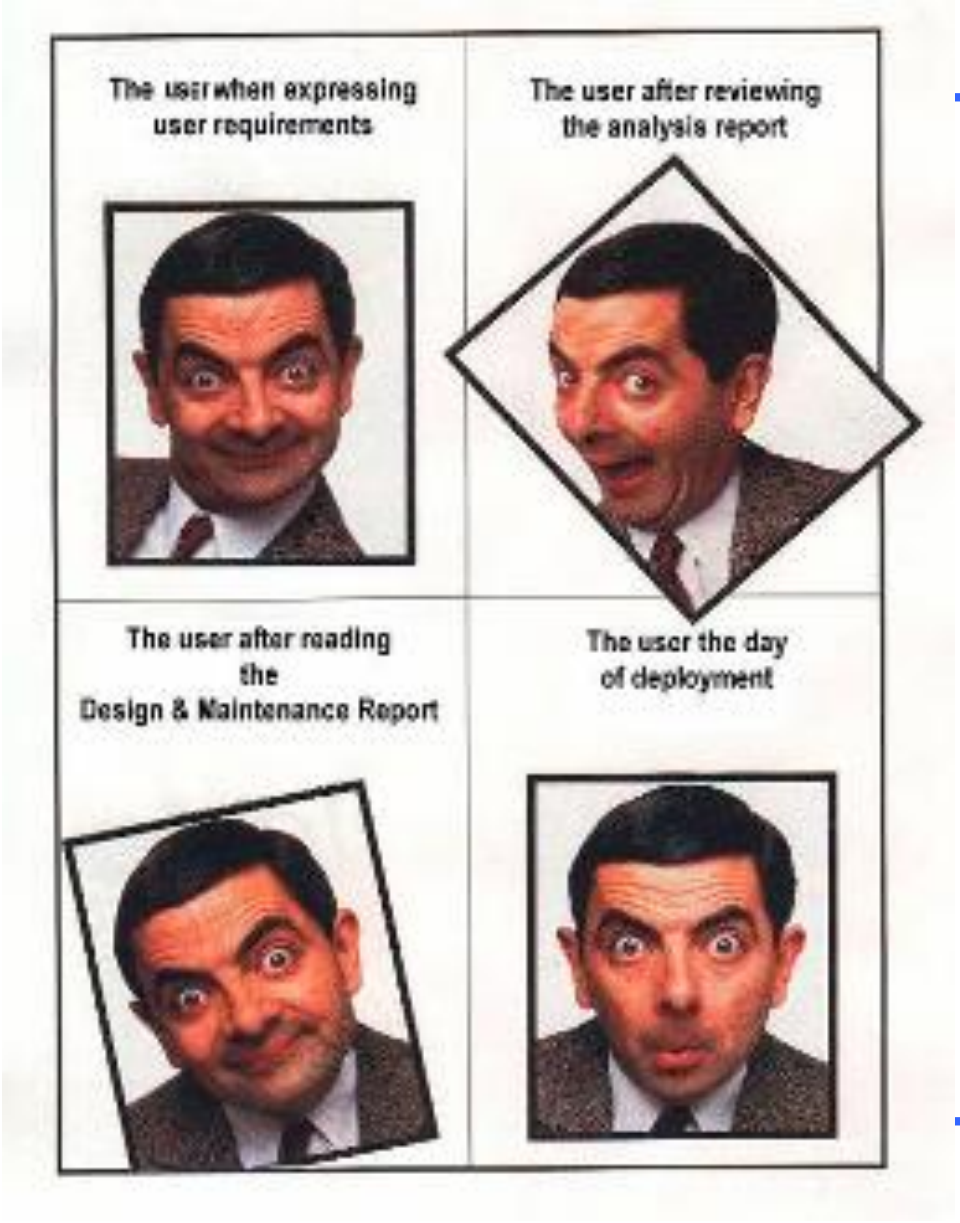


The Unified Modelling Language (UML)



Developed through the distillation of ideas from Booch, Jacobson and Rumbagh. Added to and Adopted via the Object Management Group (OMG) in 1997 as version 1.0.

Unified as a notation, across the development lifecycle, across application domains, across platforms and implementation languages, across development processes, across internal concepts.



Unified across the development lifecycle

What is UML and what is it not?

Specification and design language, not a programming language! Modeling language.

- Mostly visual but has semantics
- Diagrams consist of well-defined elements (graphical) and have rules on how to use and combine elements
- Abstract syntax, well-formedness rules, semantics can be found in the official documentation

UML is not a software tool

UML is not a methodology but only a notation

Why use UML for Information Modeling

De facto standard for object-oriented modeling, maintained by a standards organisation OMG

Bring good ideas in consistent framework, supported by many tools, profiles, methods and processes exist

Diagrams help visualize designs and cope with complexity

Implements or support the principles of:

- Abstraction
- Separation of concerns
- Modularity
- Rigor and formality

UML views and diagrams

Major View	Sub-view	Diagram	Concepts
structural	static	class diagram	association, class, dependency, generalization, interface, realization
	design	internal structure	connector, interface, part, port, provided interface, role, required interface
		collaboration diagram	connector, collaboration, collaboration use, role
		component diagram	component, dependency, port, provided interface, realization, required interface, subsystem
	use case	use case diagram	actor, association, extend, include, use case, generalization

UML views and diagrams

Major View	Sub-view	Diagram	Concepts
structural	static	class diagram	association, class, dependency, generalization, interface, realization
	design	internal structure	connector, interface, part, port, provided interface, role, required interface
		collaboration diagram	connector, collaboration, collaboration use, role
		component diagram	component, dependency, port, provided interface, realization, required interface, subsystem
	use case	use case diagram	actor, association, extend, include, use case, generalization

UML views and diagrams cont.

Major View	Sub-view	Diagram	Concepts
dynamic	state machine	state machine diagram	completion transition, do activity, effect, event, region, state, transition, trigger
	activity	activity diagram	action, activity, control flow, control node, data flow, exception, expansion region, fork, join, object node, pin
	interaction	sequence diagram	occurrence specification, execution specification, interaction, lifeline, message, signal
		communication diagram	collaboration, guard condition, message, role, sequence number

UML views and diagrams cont.

Major View	Sub-view	Diagram	Concepts
dynamic	state machine	state machine diagram	completion transition, do activity, effect, event, region, state, transition, trigger
	activity	activity diagram	action, activity, control flow, control node, data flow, exception, expansion region, fork, join, object node, pin
	interaction	sequence diagram	occurrence specification, execution specification, interaction, lifeline, message, signal
		communication diagram	collaboration, guard condition, message, role, sequence number

UML views and diagrams cont.

Major View	Sub-view	Diagram	Concepts
physical	deployment	deployment diagram	artifact, dependency, manifestation, node
model management	model management	package diagram	import, model, package
	profile	package diagram	constraint, profile, stereotype, tagged value

UML Quick Overview

You can model about 80% of problems using 20% of UML…
that is intention in this module

Use case diagrams

- Describe the functional behavior of the system as seen by the user
- Used during requirements elicitation

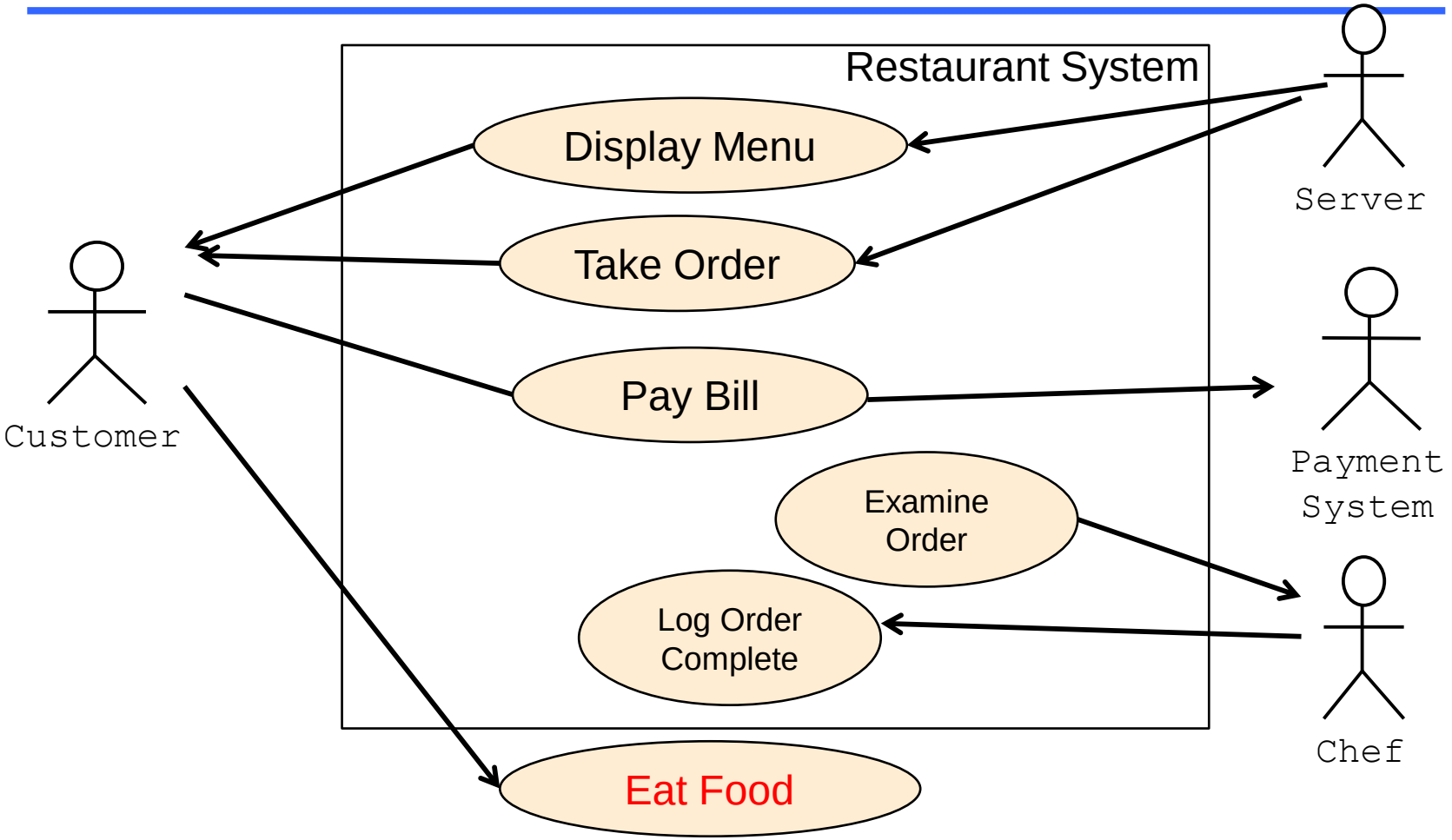
Class diagrams

- Describe the static structure of the system: Objects, attributes, associations

Sequence and Activity diagrams

- Describe the dynamic behavior between objects of the system

An Introductory **Use Case Diagram** to model functionality of an information system to support a restaurant waiter with an iPad App.

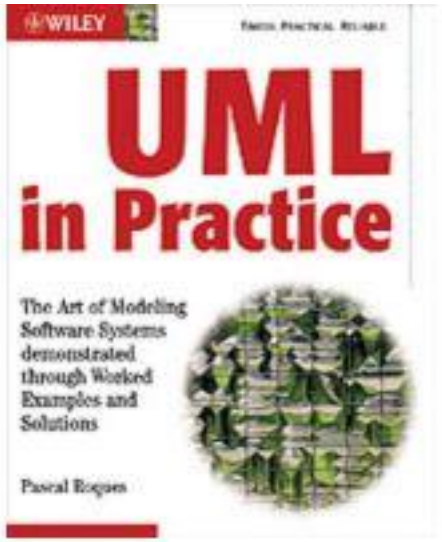


Note that we can model "Eat Food" if we like in our diagram BUT as the INFO SYSTEM does not support it, it is drawn outside the box

References

Some of the UML examples come from

UML in Practice, Author: Pascal Roques, Publisher: Wiley;



That's All Folks Thank You for Listening