SENG 350
- Software Architecture & Design

Shuja Mughal

UML Diagrams

Fall 2024

University of Victoria

UML Diagrams

- ERD
- Component/Package Diagram
- Deployment Diagram

Entity Relationship Diagram

University of Victoria

Entity Relationship Modeling

Objectives:

- To illustrate how relationships between entities are defined and refined.
- To know how relationships are incorporated into the database design process.
- To describe how ERD components affect database design and implementation.

University of Victoria

What is Conceptual Database Design?

- Process of describing the data, relationships between the data, and the constraints on the data.
- After analysis - Gather all the essential data required and understand how the data are related
- The focus is on the *data*, rather than on the processes.
- The output of the conceptual database design is a **Conceptual Data Model** (*+ Data Dictionary*)

University of Victoria

Gathering Information for Conceptual Data Modeling

Two perspectives

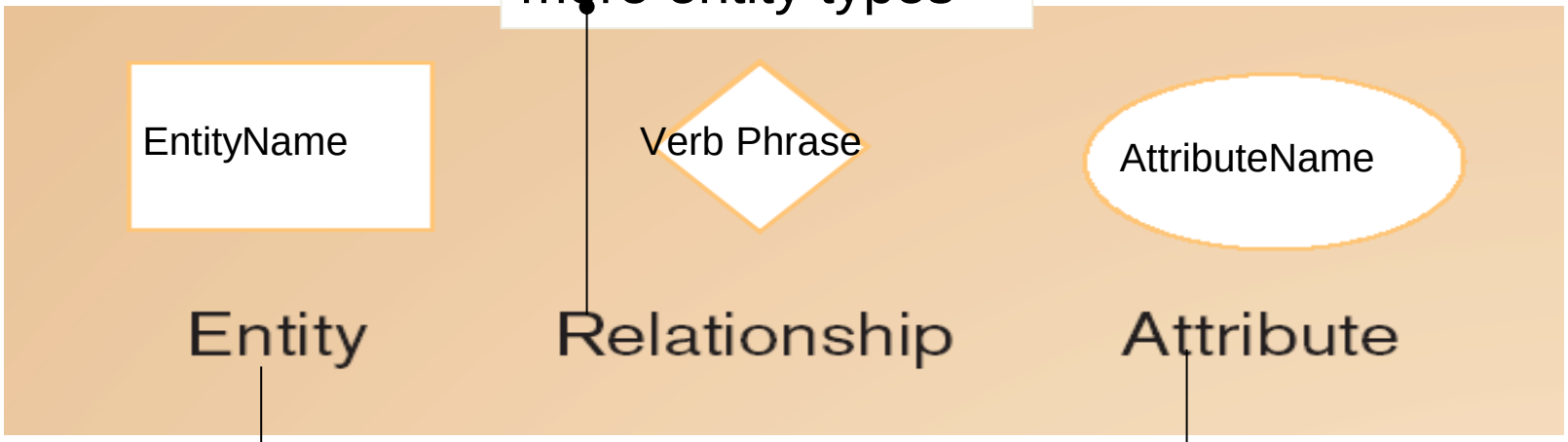
- Top-down
  - Data model is derived from an intimate understanding of the business.
- Bottom-up
  - Data model is derived by reviewing specifications and business documents.

Entity-Relationship (ER) Modeling.

- **ER Modeling** is a *top-down* approach to database design.
- Entity Relationship (ER) Diagram
  - A detailed, logical representation of the entities, associations and data elements for an organization or business
- Notation uses three main constructs
  - Data entities
  - Relationships
  - Attributes

Chen Notation

Association between the instances of one or more entity types



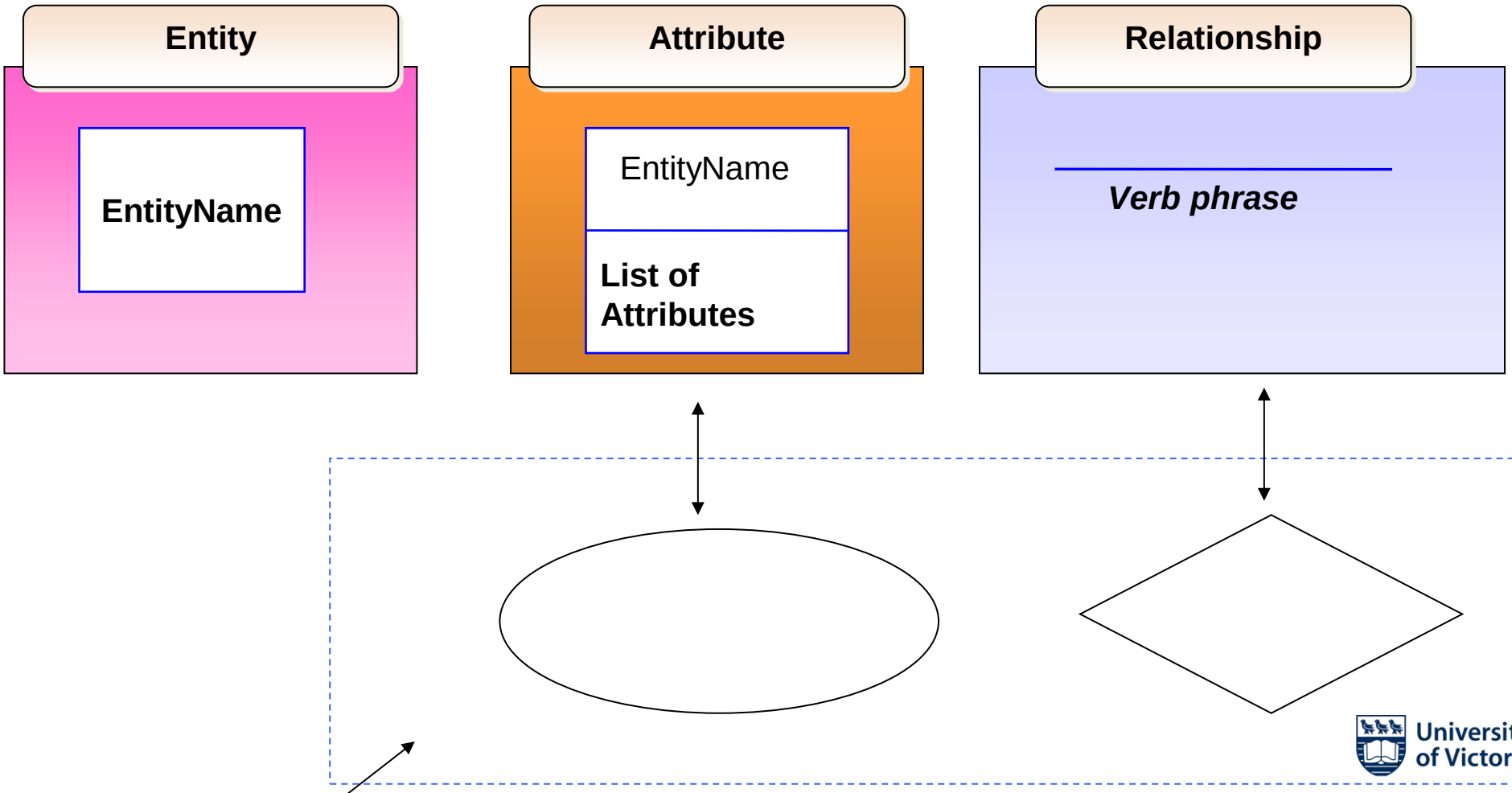
Person, place, object, event or concept about which data is to be maintained

named property or characteristic of an entity

Represents a set or collection of objects in the real world that share the same properties

University of Victoria

Crow's Foot Notation



Entities

Examples of entities:

- Person: EMPLOYEE, STUDENT, PATIENT
- Place: STORE, WAREHOUSE
- Object: MACHINE, PRODUCT, CAR
- Event: SALE,REGISTRATION, RENEWAL
- Concept: ACCOUNT, COURSE




Guidelines for naming and defining entity types:

- An entity type name is a singular noun
- An entity type should be descriptive and specific
- An entity name should be concise
- Event entity types should be named for the result of the event, not the activity or process of the event.

University of Victoria

Attributes

Example of entity types and associated attributes:

STUDENT: Student_ID, Student_Name, Home_Address, Phone_Number, Major

Guidelines for naming attributes:

An attribute name is a noun.

An attribute name should be unique

To make an attribute name unique and clear, each attribute name should follow a standard format

Similar attributes of different entity types should use similar but distinguishing names.

Identifier Attributes

Candidate key

- Attribute (or combination of attributes) that uniquely identifies each instance of an entity type
- Some entities may have more than one candidate key
  - Ex: A candidate key for EMPLOYEE is Employee_ID, a second is the combination of Employee_Name and Address.
  - If there is more than one candidate key, need to make a choice.

Identifier

- A candidate key that has been selected as the unique identifying characteristic for an entity type

Referential Attributes

Make Reference to another instance in another table

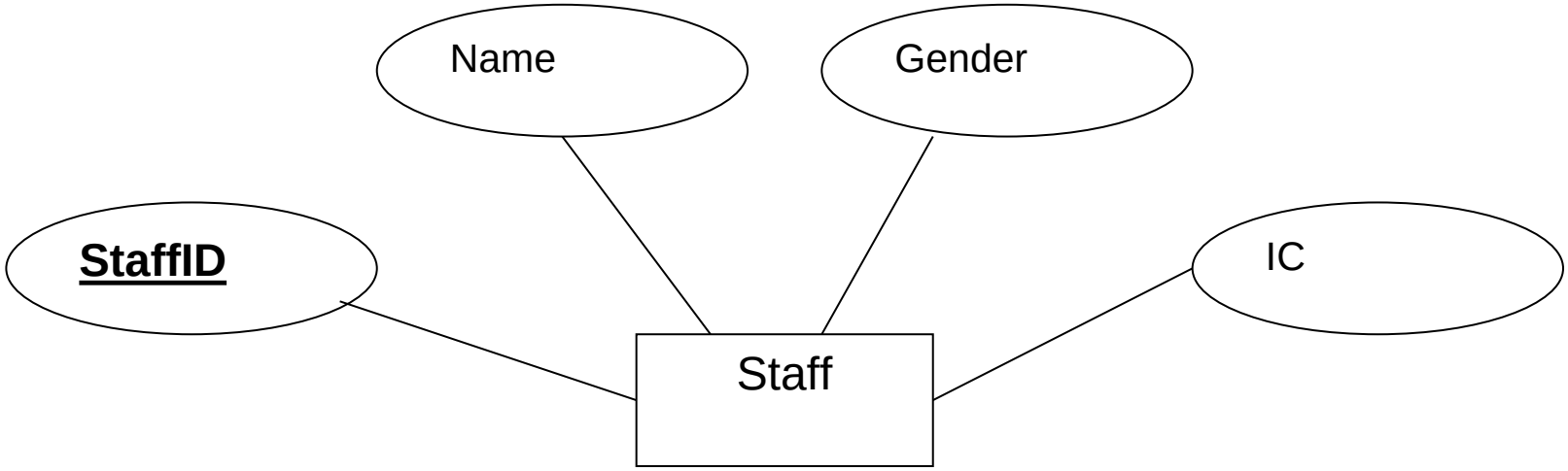
Referential attribute: Ties the lecturer entity to another entity that is department.

Instance of Lecturer.

Name	IdNum	DeptID	Email
Sam	105	LG	sam@a.com
Mary	106	IT	mary@a.com
John	107	ENG	john@a.com
Lim	108	IT	lim@a.com

University of Victoria

Example



Staff
StaffID **PK**
Name Gender IC

University of Victoria

Relationships

→ Associations between instances of one or more entity types that are of interest

→ Given a name that describes its function.

• relationship name is an active or a passive verb.

Relationship name:
writes

Author ⟷ **Book**

An author writes one or more books
A book can be written by one or more authors.

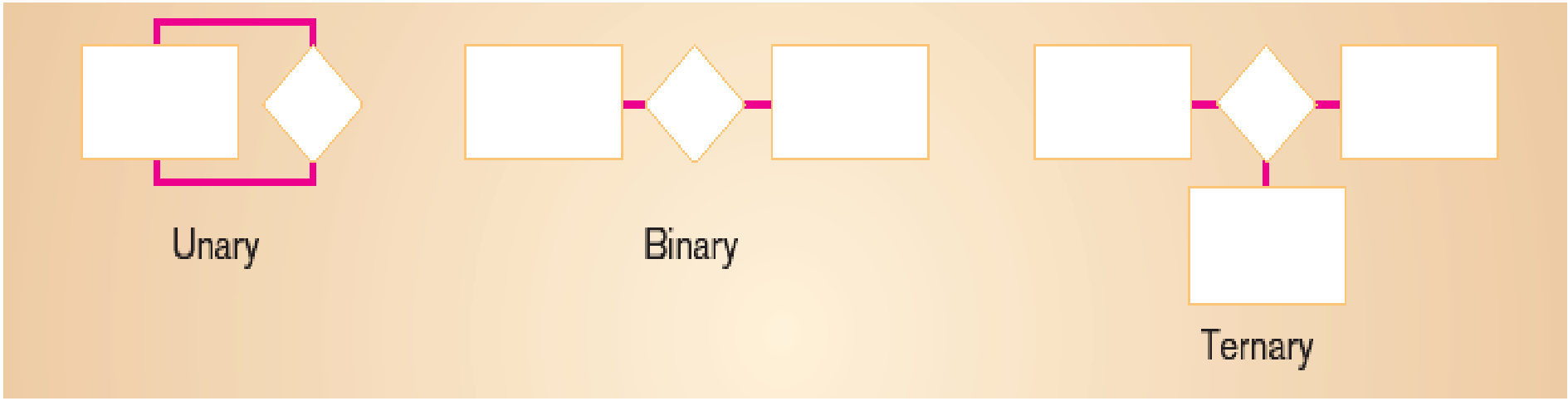
University of Victoria

Degree of Relationships

Degree: number of entity types that participate in a relationship

Three cases

- **Unary:** between two instances of one entity type
- **Binary:** between the instances of two entity types
- **Ternary:** among the instances of three entity types



University of Victoria

Cardinality and Connectivity

Relationships can be classified as either

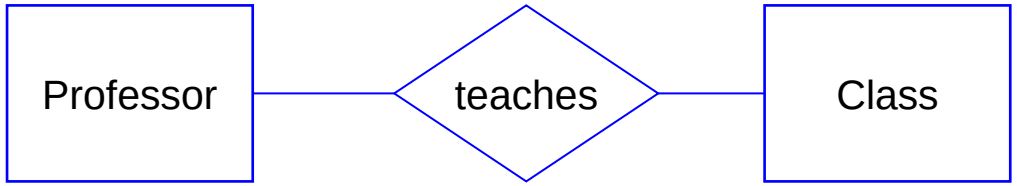
one – to – one
one – to – many
many – to –many

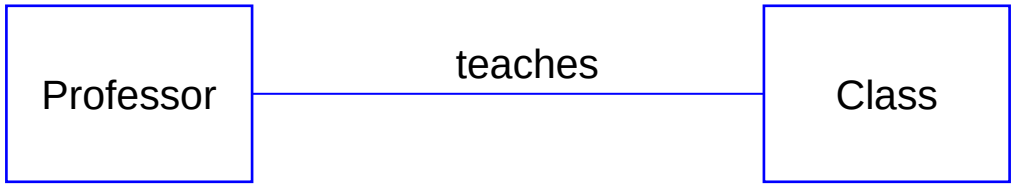
Connectivity

Cardinality : minimum and maximum number of instances of Entity B that can (or must be) associated with each instance of entity A.

University of Victoria

Cardinality and Connectivity



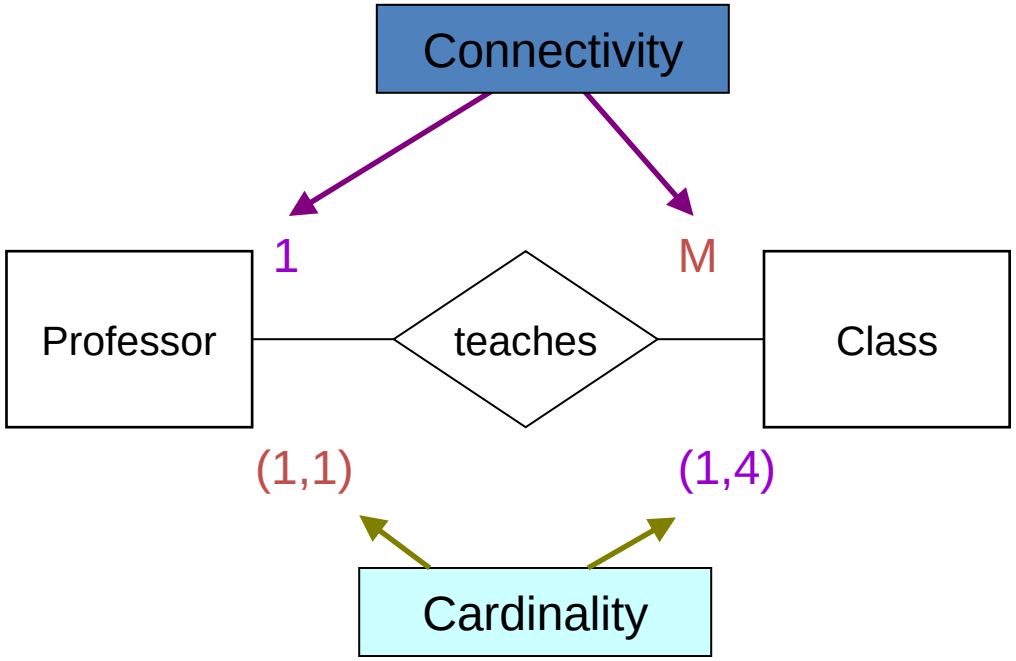


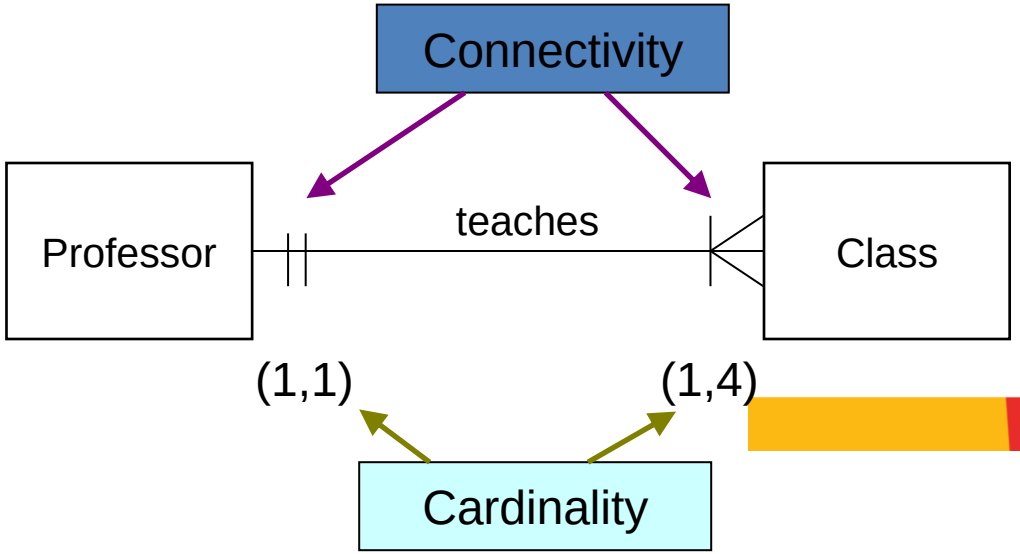
A professor teaches class OR
A class is taught by professor

How Many??

University of Victoria

Cardinality and Connectivity





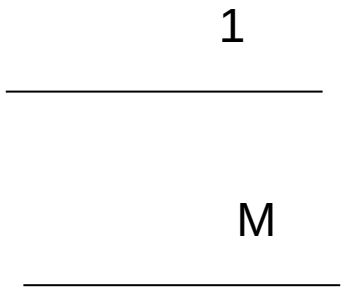
University of Victoria

Connectivity

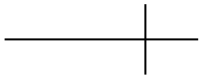
Chen Model

1 to represent one.

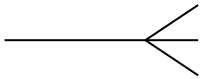
M to represent many



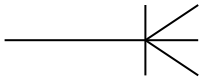
Crow's Foot



One



many



One or many

Mandatory one , means (1,1)

Optional? – we'll see after this

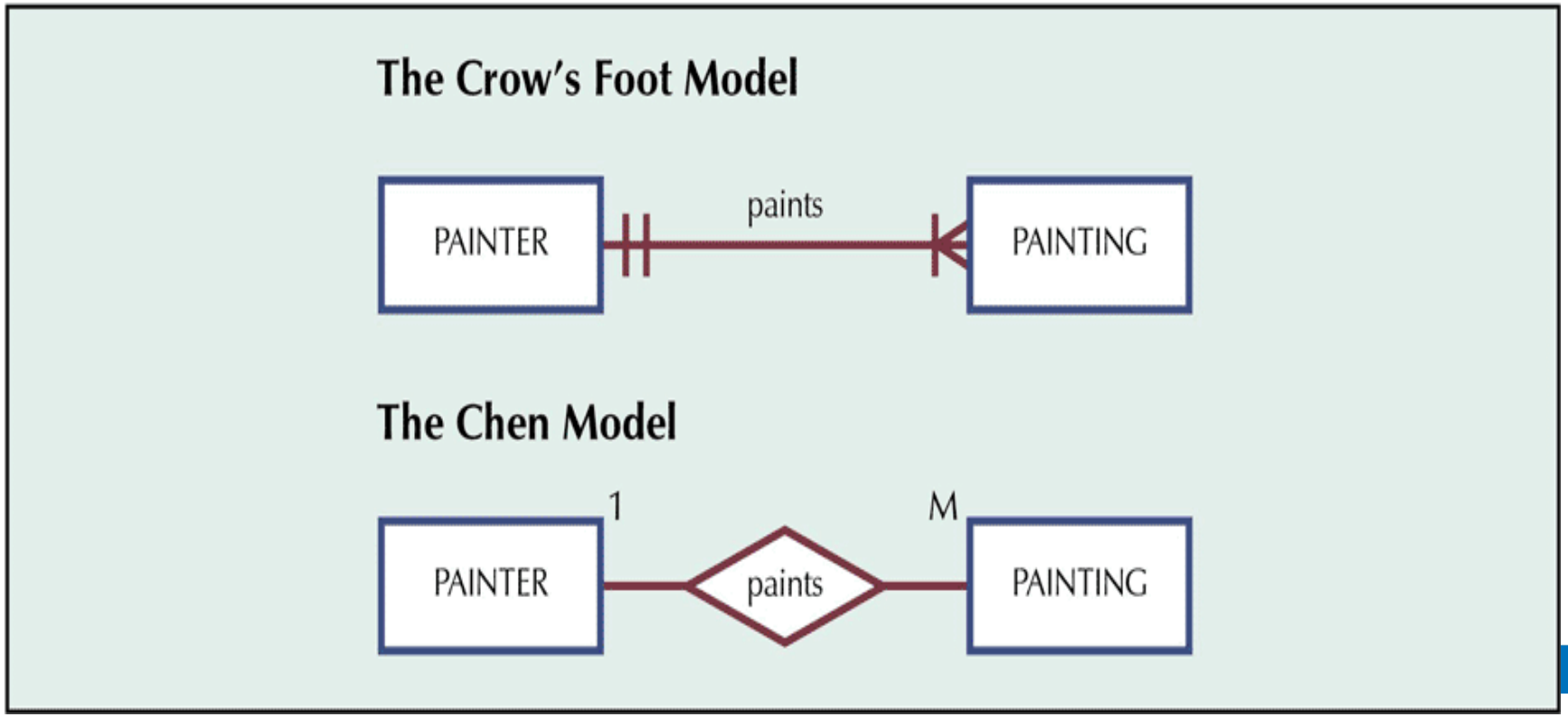
University of Victoria

Binary Relationships

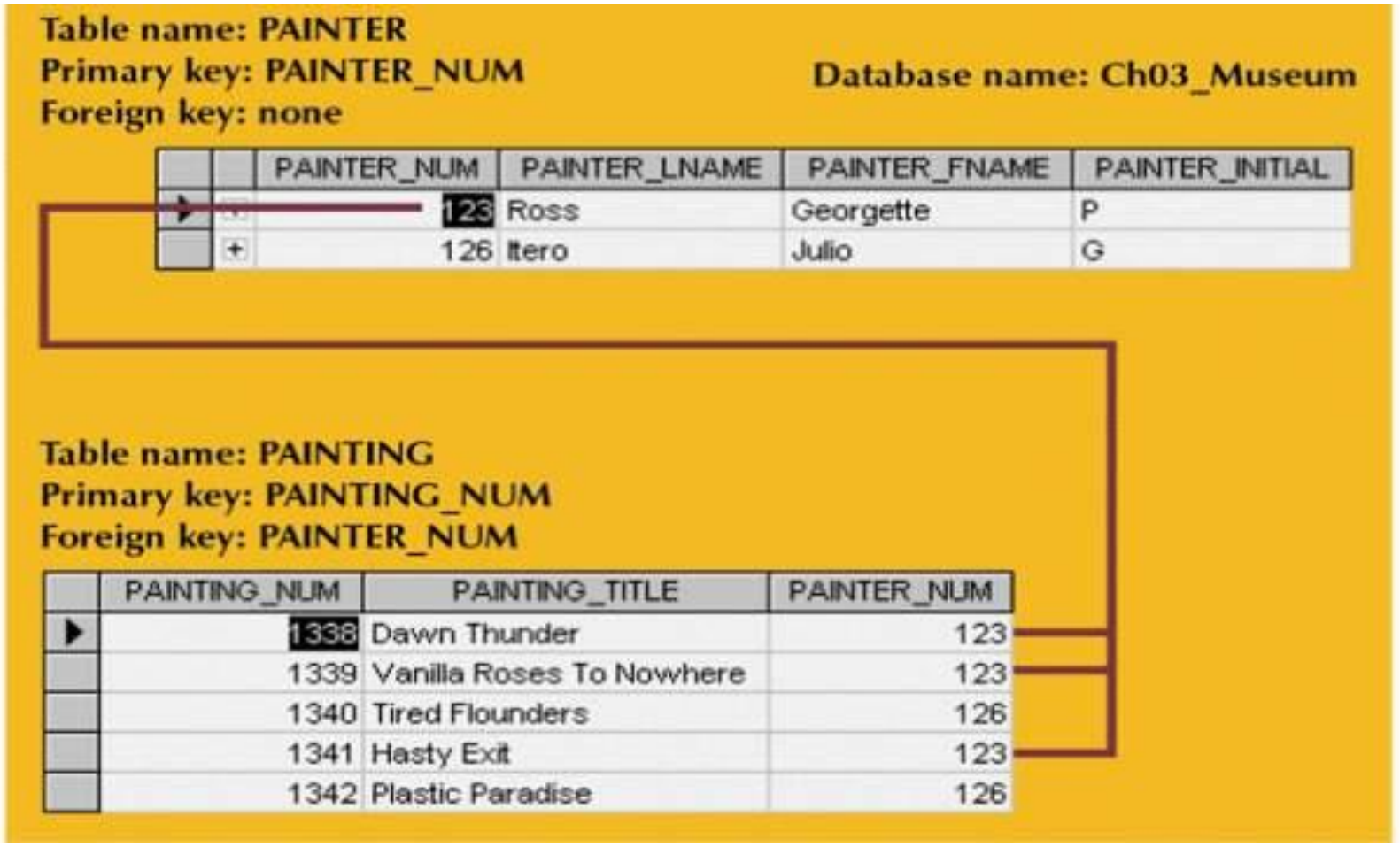
1:M relationship

Relational modeling ideal

Should be the norm in any relational database design



The 1: M relationship between PAINTER and PAINTING



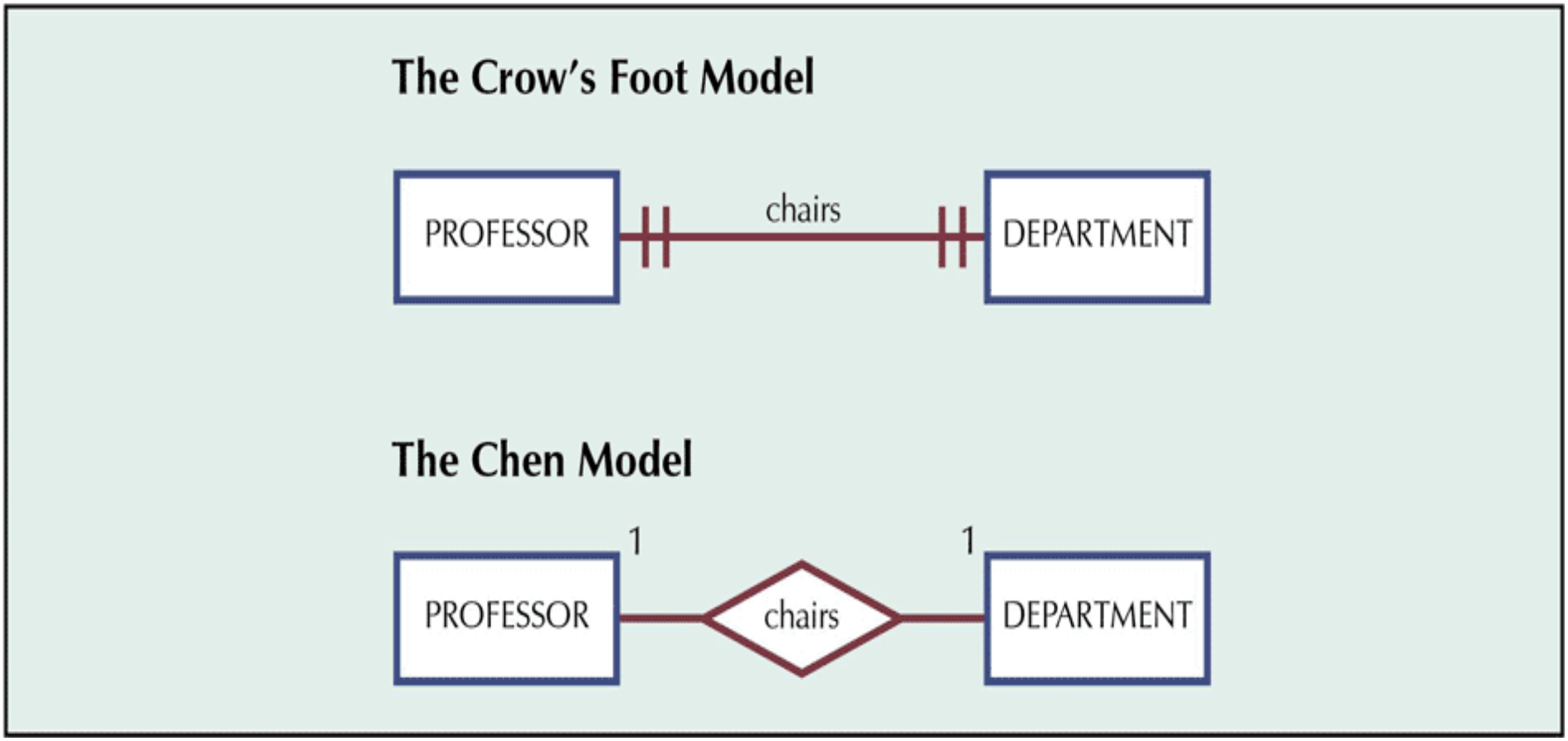
The Implemented 1:M relationship between PAINTER and PAINTING

University of Victoria

Binary Relationships

Relationship

- A single entity instance in one entity class is related to a single entity instance in another entity class
- Could indicate that two entities actually belong in the same table



The 1:1 Relationship Between PROFESSOR and DEPARTMENT

University of Victoria

Table name: PROFESSOR **Database name: Ch03_TinyCollege**
Primary key: EMP_NUM
Foreign key: DEPT_CODE

EMP_NUM	DEPT_CODE	PROF_OFFICE	PROF_EXTENSION	PROF_HIGH_DEGREE
103	HIST	DRE 156	6783	Ph.D.
104	ENG	DRE 102	5561	MA
105	ACCT	KLR 229D	8665	Ph.D.
106	MKT/MGT	KLR 126	3899	Ph.D.
110	BIOL	AAK 160	3412	Ph.D.
114	ACCT	KLR 211	4436	Ph.D.
155	MATH	AAK 201	4440	Ph.D.
160	ENG	DRE 102	2248	Ph.D.
162	CIS	KLR 203E	2359	Ph.D.
191	MKT/MGT	KLR 409B	4016	DBA
195	PSYCH	AAK 297	3550	Ph.D.
209	CIS	KLR 333	3421	Ph.D.
228	CIS	KLR 300	3000	Ph.D.
297	MATH	AAK 194	1145	Ph.D.
299	ECON/FIN	KLR 284	2851	Ph.D.
301	ACCT	KLR 244	4683	Ph.D.
335	ENG	DRE 208	2000	Ph.D.
342	SOC	BBG 208	5514	Ph.D.
387	BIOL	AAK 230	8665	Ph.D.
401	HIST	DRE 156	6783	MA
425	ECON/FIN	KLR 284	2851	MBA
435	ART	BBG 185	2278	Ph.D.

The Implemented 1:1 Relationship Between PROFESSOR and DEPARTMENT

Table name: DEPARTMENT
Primary key: DEPT_CODE
Foreign key: EMP_NUM

DEPT_CODE	DEPT_NAME	SCHOOL_CODE	EMP_NUM	DEPT_ADDRESS	DEPT_EXTENSION
ACCT	Accounting	BUS	114	KLR 211, Box 52	3119
ART	Fine Arts	A&SCI	435	BBG 185, Box 128	2278
BIOL	Biology	A&SCI	387	AAK 230, Box 415	4117
CIS	Computer Info. Systems	BUS	209	KLR 333, Box 56	3245
ECON/FIN	Economics/Finance	BUS	299	KLR 284, Box 63	3126
ENG	English	A&SCI	160	DRE 102, Box 223	1004
HIST	History	A&SCI	103	DRE 156, Box 284	1867
MATH	Mathematics	A&SCI	297	AAK 194, Box 422	4234
MKT/MGT	Marketing/Management	BUS	106	KLR 126, Box 55	3342
PSYCH	Psychology	A&SCI	195	AAK 297, Box 438	4110
SOC	Sociology	A&SCI	342	BBG 208, Box 132	2008

Binary Relationships

M:N relationships

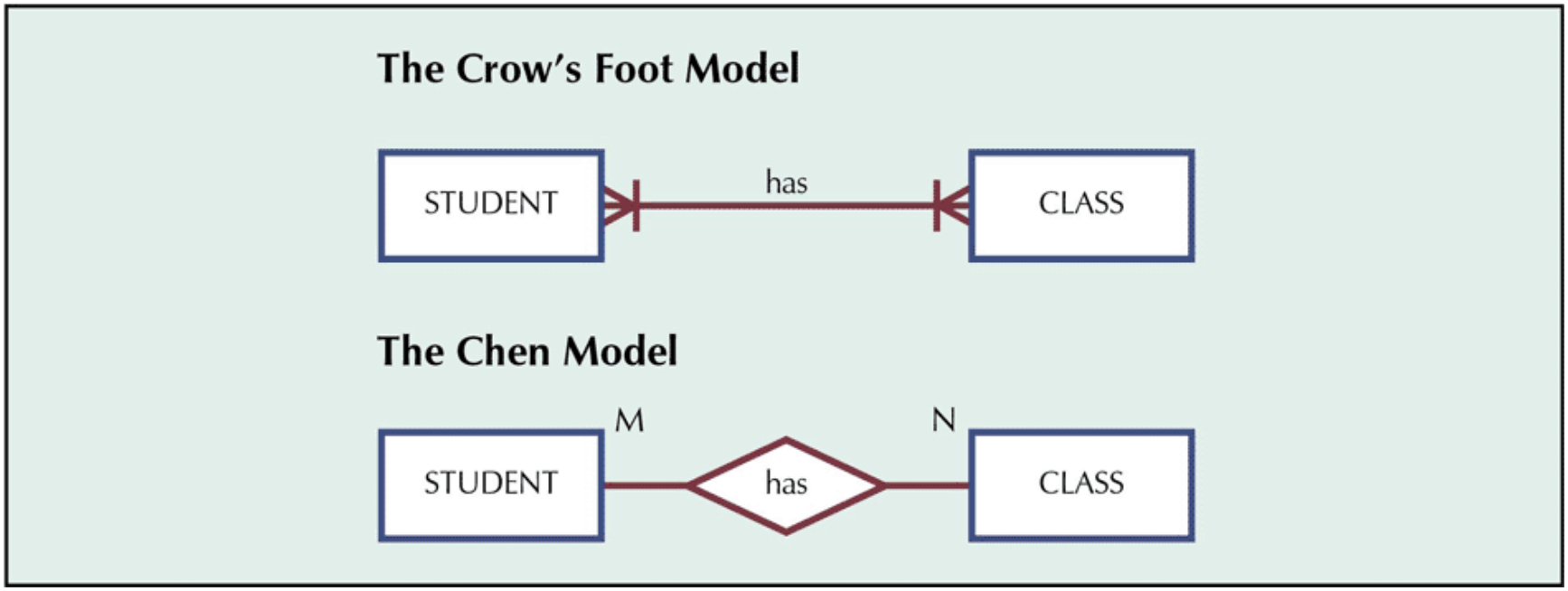
Must be avoided because they lead to *data redundancies*.

This can be implemented by breaking it up to produce a set of 1:M relationships

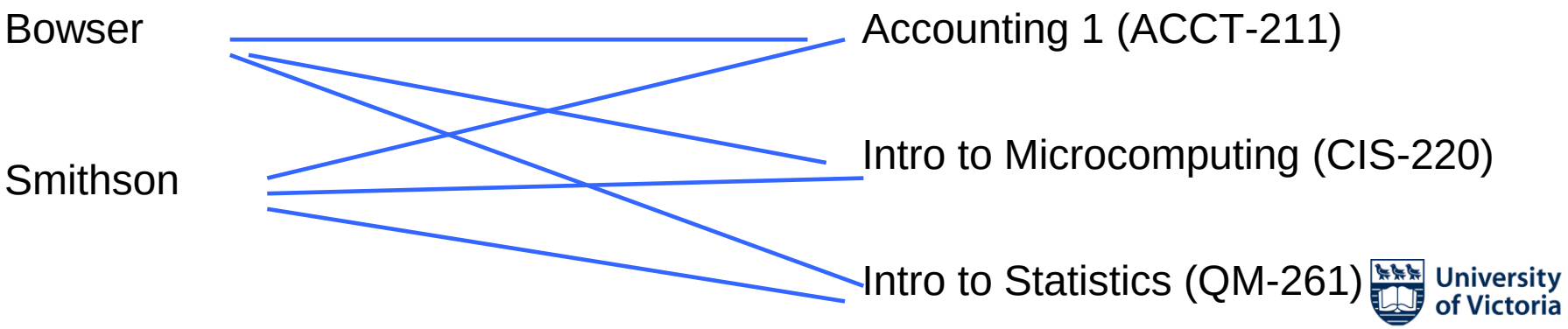
Can avoid problems inherent to M:N relationship by creating a ***composite entity or bridge entity***

This will be used to link the tables that were originally related in a M:N relationship

The composite entity structure includes-as f**oreign keys-at least the primary keys of the tables that are to be linked.**



The M:N Relationship Between STUDENT and CLASS



University of Victoria

This CANNOT be implemented as shown next…..

The tables have **many redundancies!!**

Table name: STUDENT
Primary key: STU_NUM
Foreign key: none

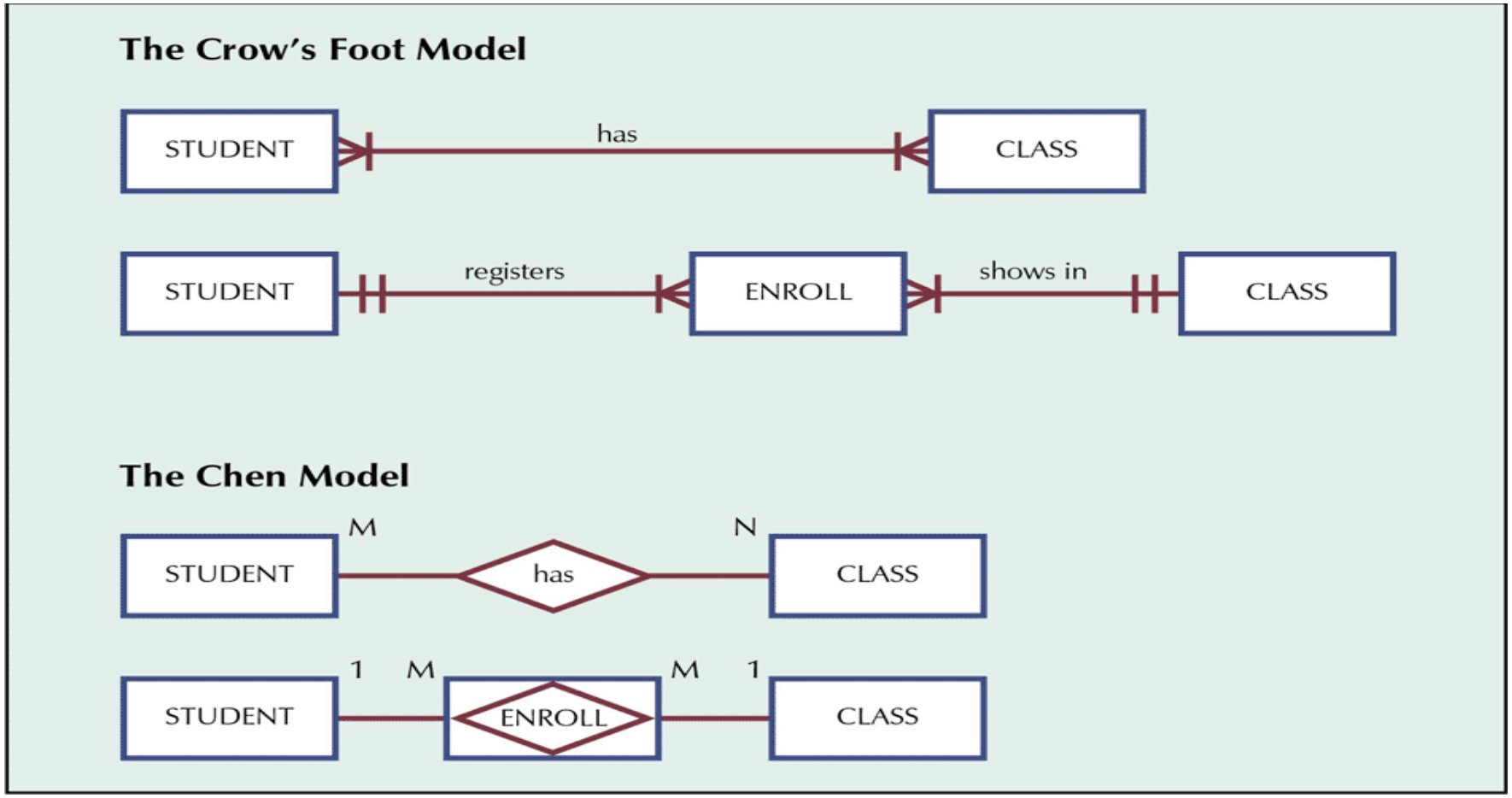
Database name: Ch03_CollegeTry

STU_NUM	STU_LNAME	CLASS_CODE
321452	Bowser	10014
321452	Bowser	10018
321452	Bowser	10021
324257	Smithson	10014
324257	Smithson	10018
324257	Smithson	10021

Table name: CLASS
Primary key: CLASS_CODE
Foreign key: STU_NUM

+ STU_NUM

CLASS_CODE	STU_NUM	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
10014	321452	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10014	324257	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10018	321452	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10018	324257	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10021	321452	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10021	324257	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114



Changing the M:N relationship to TWO 1:M relationships

University of Victoria

Table name: STUDENT
Primary key: STU_NUM
Foreign key: none

	STU_NUM	STU_LNAME
▶	321452	Bowser
	324257	Smithson

The database designer has 2 main options to define **a composite table's primary key:**
either
use the combination of those foreign keys or **create a new primary key.**

Table name: ENROLL
Primary key: CLASS_CODE + STU_NUM
Foreign key: CLASS_CODE, STU_NUM

	CLASS_CODE	STU_NUM	ENROLL_GRADE
▶	10014	321452	C
	10014	324257	B
	10018	321452	A
	10018	324257	B
	10021	321452	C
	10021	324257	C

Foreign keys reference the primary keys in the other tables of which it has a relationship with

Table name: CLASS
Primary key: CLASS_CODE
Foreign key: CRS_CODE

	CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	CLASS_ROOM	PROF_NUM
▶	10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
	10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
	10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114

Converting the M:N relationship into TWO 1:M relationships

How to Evaluate a Data Model?

A good data model has the following:

- Accuracy and completeness
- Non redundancy
- Enforcement of business rules
- Data Reusability
- Stability and Flexibility
- Communication Effectiveness
- Simplicity

A Common Mistake

Modeling the **business processes** or **functions** instead of the **data**.

What data we want to keep??

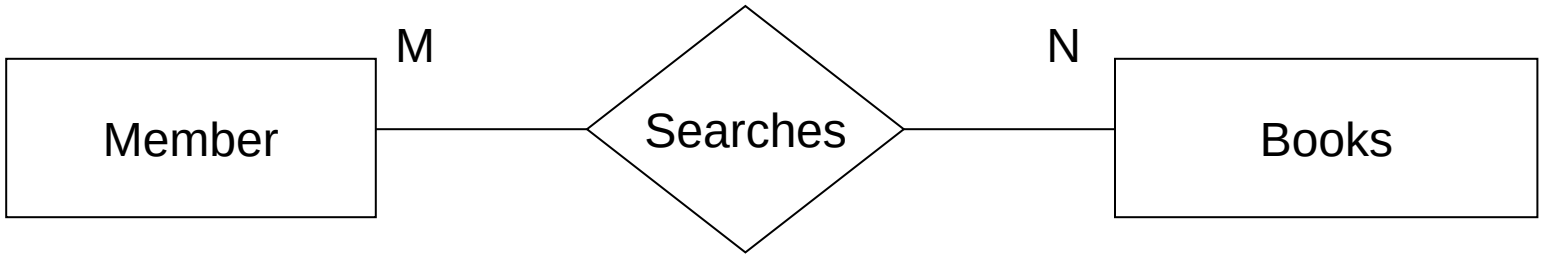
We are interested in modeling the data,

NOT the processes or functions that use

or generate those data.

University of Victoria

Example:



Is this part of the data requirement?

Are we interested to know the books searched by the members?

If answer is NO, then DO NOT include that as a relationship.

Use other appropriate diagramming techniques to capture the business processes such as Data Flow Diagram.

Do not mix up the use of ER Modeling with DFD.

University of Victoria

Component Diagram

Component Diagram

Component is an encapsulated, reusable, and replaceable part of the software.

Two stereotypes:

- «component»
- «subsystem» for larger pieces of systems

University of Victoria

Component Diagram

A UML component is a building block of the system. It is represented as a rectangle with a tabbed rectangle symbol inside

Components have different lifetimes:

- Some exist only at design time
  - Classes, associations
- Others exist until compile time
  - Source code, pointers
- Some exist at link or only at runtime
  - Linkable libraries, executables, addresses

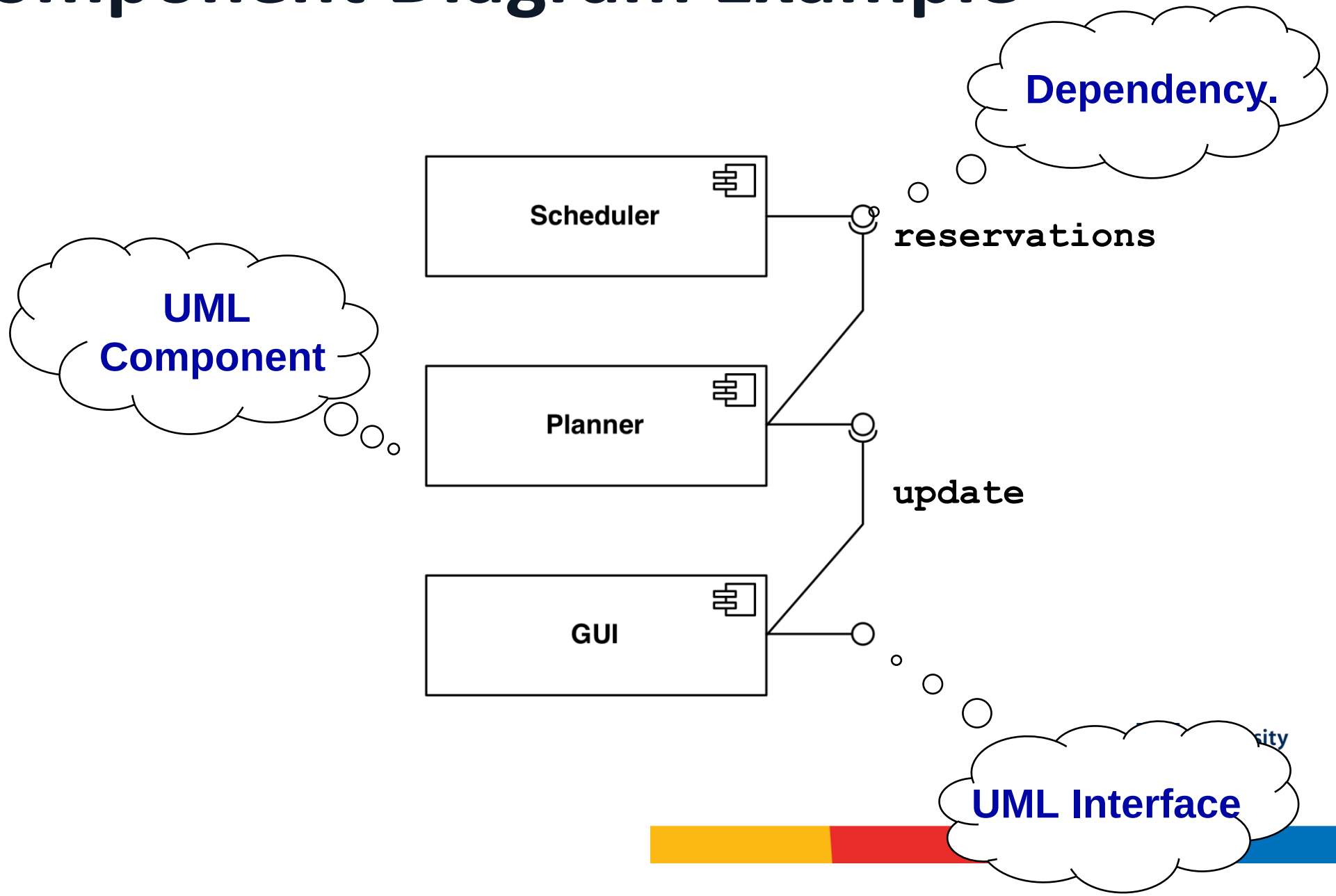
Component Diagram

- Illustrates dependencies between components at design time, compilation time and runtime source code, linkable libraries, executable
- Used to model the top-level view of the system

UML Component Diagram

- The dependencies (edges in the graph) are shown as lines with arrows from the client component to the supplier component:
  - The lines are often also called connectors
  - Components can be connected by “lollipops” and “grabbers”
- It is also called a “software wiring diagram“ because it shows how the software components are wired together in the overall application.

Component Diagram Example



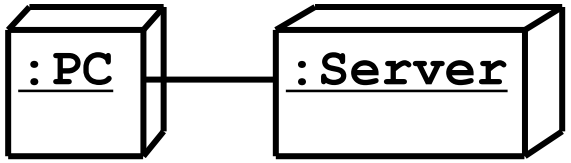
Deployment Diagram

Deployment Diagram

- Illustrates the distribution of components at run-time.
- Deployment diagrams use nodes and connections to depict the physical resources in the system.

Deployment Diagram

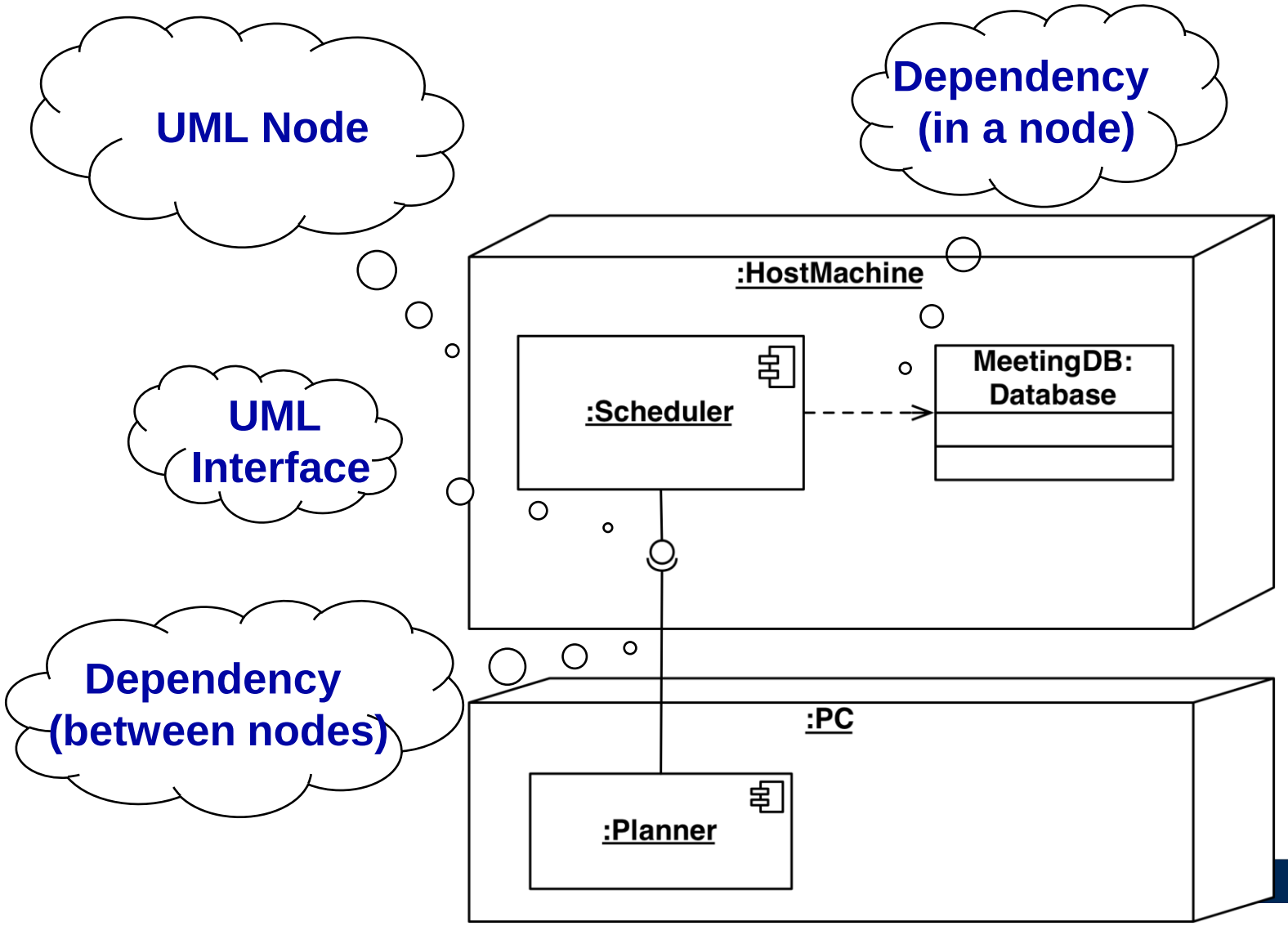
- Shows how the system is finally deployed in a given real-world situation



- A **deployment diagram** is a graph of nodes and connections ("communication associations")
  - Nodes are shown as 3-D boxes
  - Connections between nodes are shown as solid lines
  - Nodes may contain components

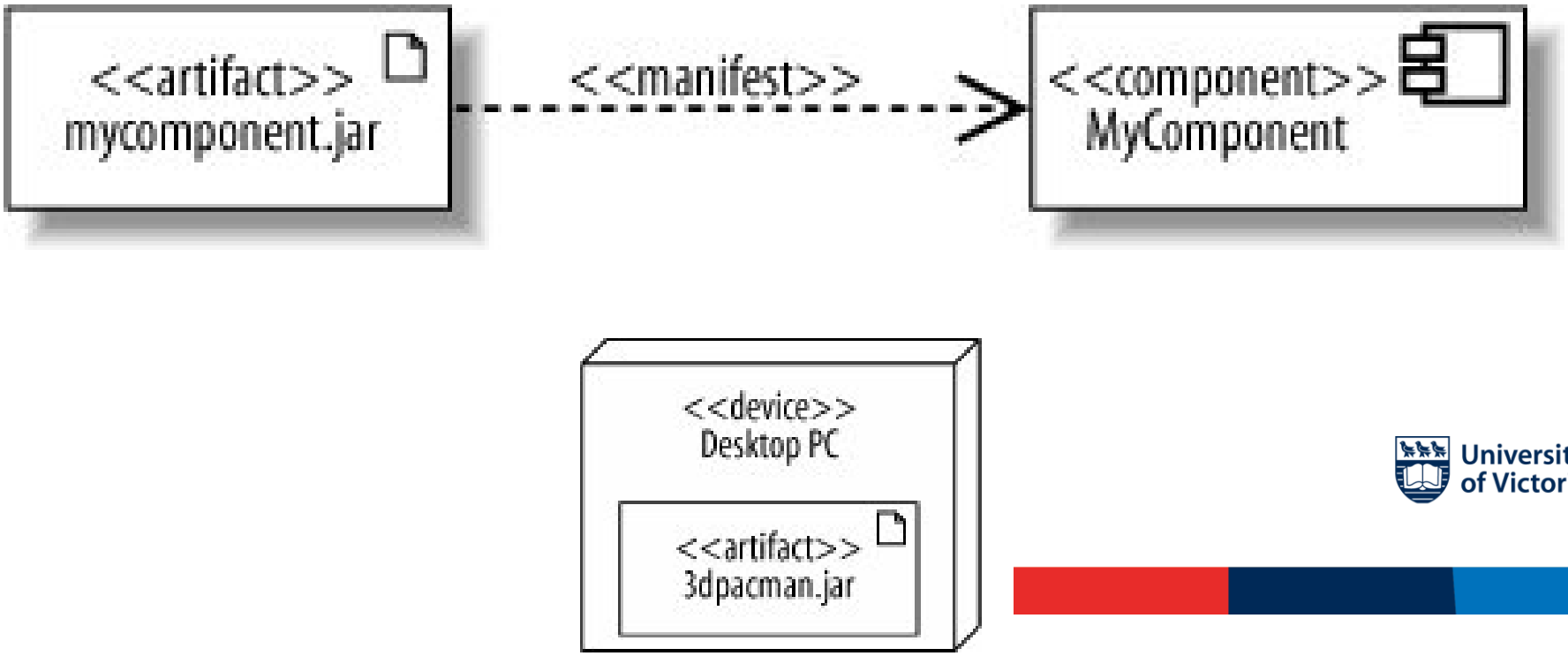
University of Victoria

Deployment Diagram Example



Deployment Diagram

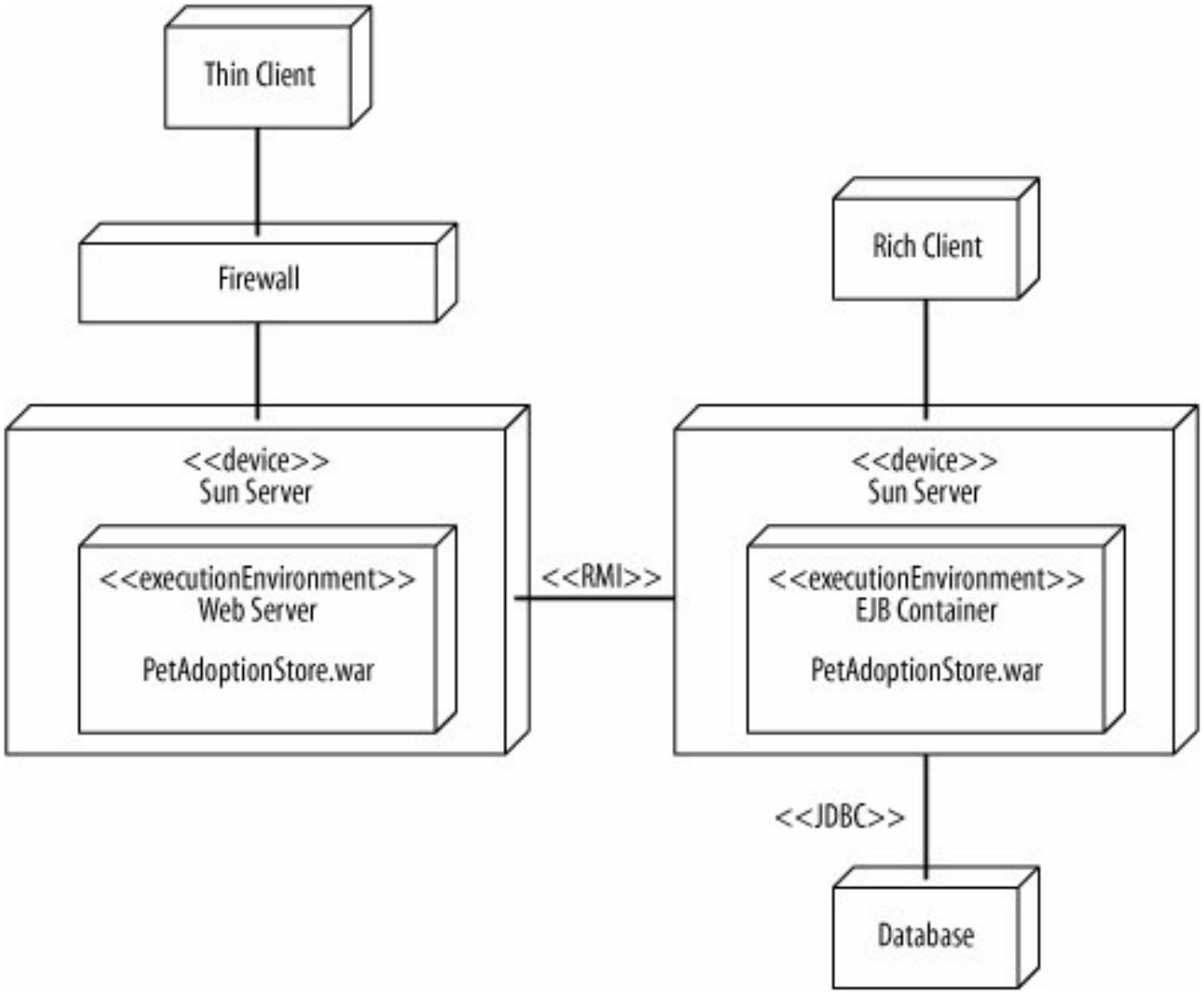
Linking deployment diagrams to component diagrams by mapping components to hardware:



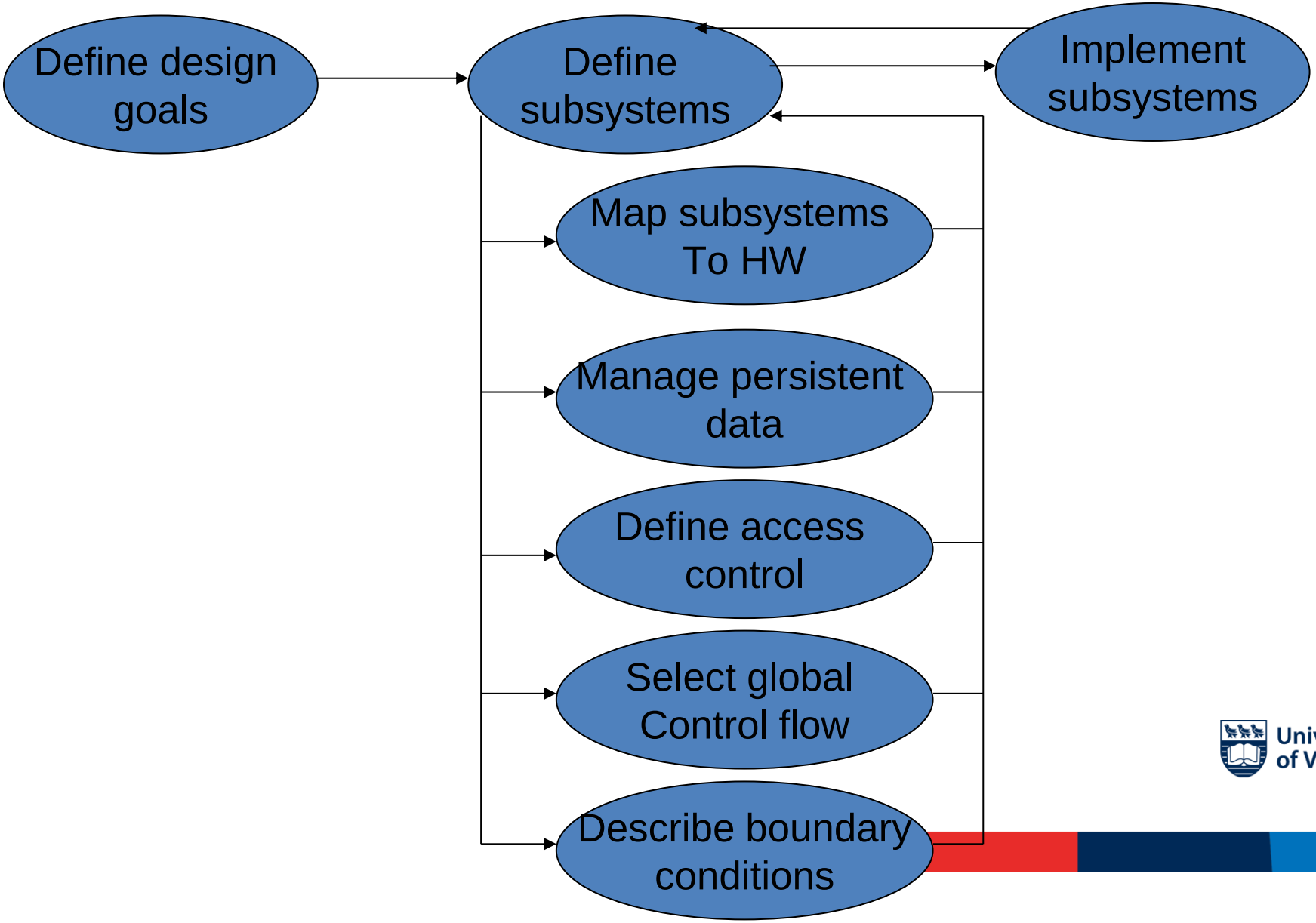
University of Victoria

Deployment Diagram

Devices, execution environments and communication: multiple levels of detail possible.



Activities of System Design



University of Victoria

Hardware Software Mapping

This system design activity addresses two questions:

- How shall we realize the subsystems: With hardware or with software?
- How do we map the object model onto the chosen hardware and/or software?
  - Mapping the Objects:
    Processor, Memory, Input/Output
  - Mapping the Associations:
    Network connections

Mapping Objects onto Hardware

Control Objects -> Processor

- Is the computation rate too demanding for a single processor?
- Can we get a speedup by distributing objects across several processors?
- How many processors are required to maintain a steady state load?

Entity Objects -> Memory

- Is there enough memory to buffer bursts of requests?

Boundary Objects -> Input/Output Devices

- Do we need an extra piece of hardware to handle the data generation rates?
- Can the desired response time be realized with the available communication bandwidth between subsystems?

University of Victoria

Friday is important!!

University of Victoria