SENG 350
- Software Architecture & Design

Shuja Mughal

UML Diagrams

Fall 2024

University of Victoria

UML Diagrams

- Sequence diagrams
- DFD
- CFD
- ERD

Sequence diagrams

Sequence diagrams

- Sequence diagrams are used to model the dynamic aspects of a software system
  - They help you to visualize how the system runs.
  - Sequence diagrams are often built from a use case and a class diagram.
    - The objective is to show how a set of objects accomplishes the required interactions with an actor.

University of Victoria

Interactions and messages

Sequence diagrams show how a set of actors and objects communicate with each other to perform:

- The steps of a use case, or
- The steps of some other piece of functionality.

The set of steps, taken together, is called an *interaction*.

Sequence diagrams can show several different types of communication.

- E.g. method calls, messages sent over the network
- These are all referred to as *messages*.

University of Victoria

Elements found in Sequence diagrams

- Instances of classes
  Shown as boxes with the class and object identifier underlined

- Actors
  Use the stick-person symbol as in use case diagrams

- Messages
  Shown as arrows from actor to object, or from object to object

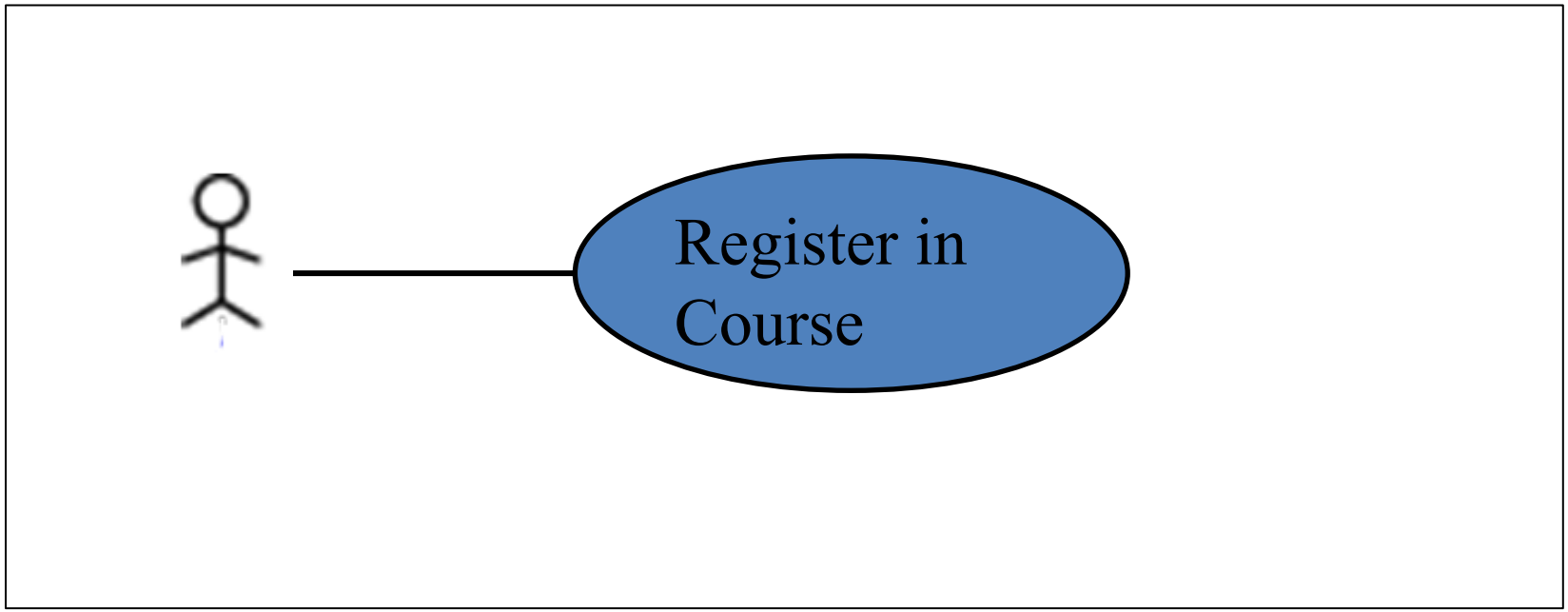
University of Victoria

Creating Sequence diagrams

You should develop a class diagram and a use case model before creating a Sequence diagrams.

University of Victoria

Sequence diagrams – an example



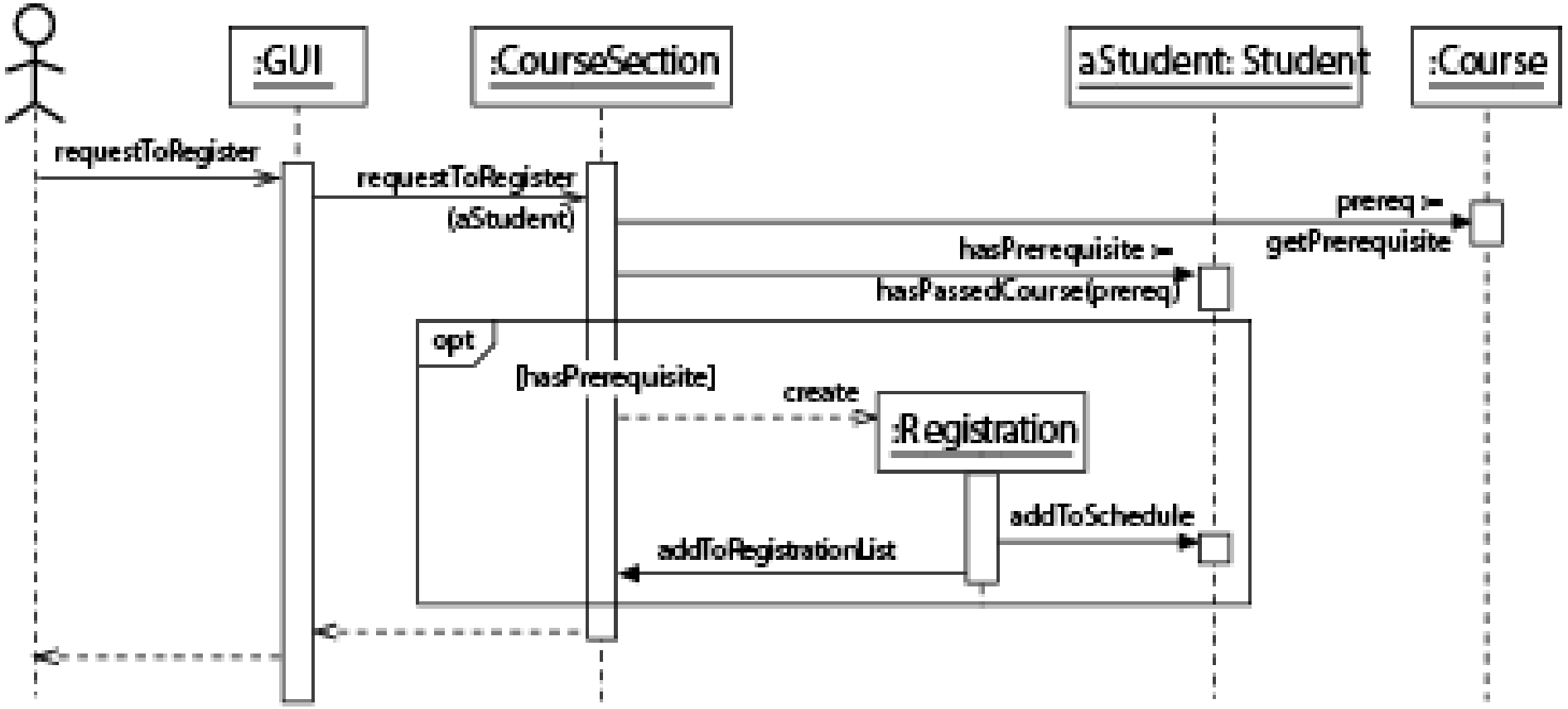
Sequence diagrams

A sequence diagram shows the sequence of messages exchanged by the set of objects performing a specific task

- The objects are arranged horizontally across the diagram.
- An actor who initiates the interaction is often shown on the left.
- The vertical dimension represents time.
- A vertical line, called a *lifeline*, is attached to each object or actor.
- The lifeline becomes a broad box, called an *activation box* during the *live activation* period.
- A message is represented as an arrow between the activation boxes of the sender and receiver.

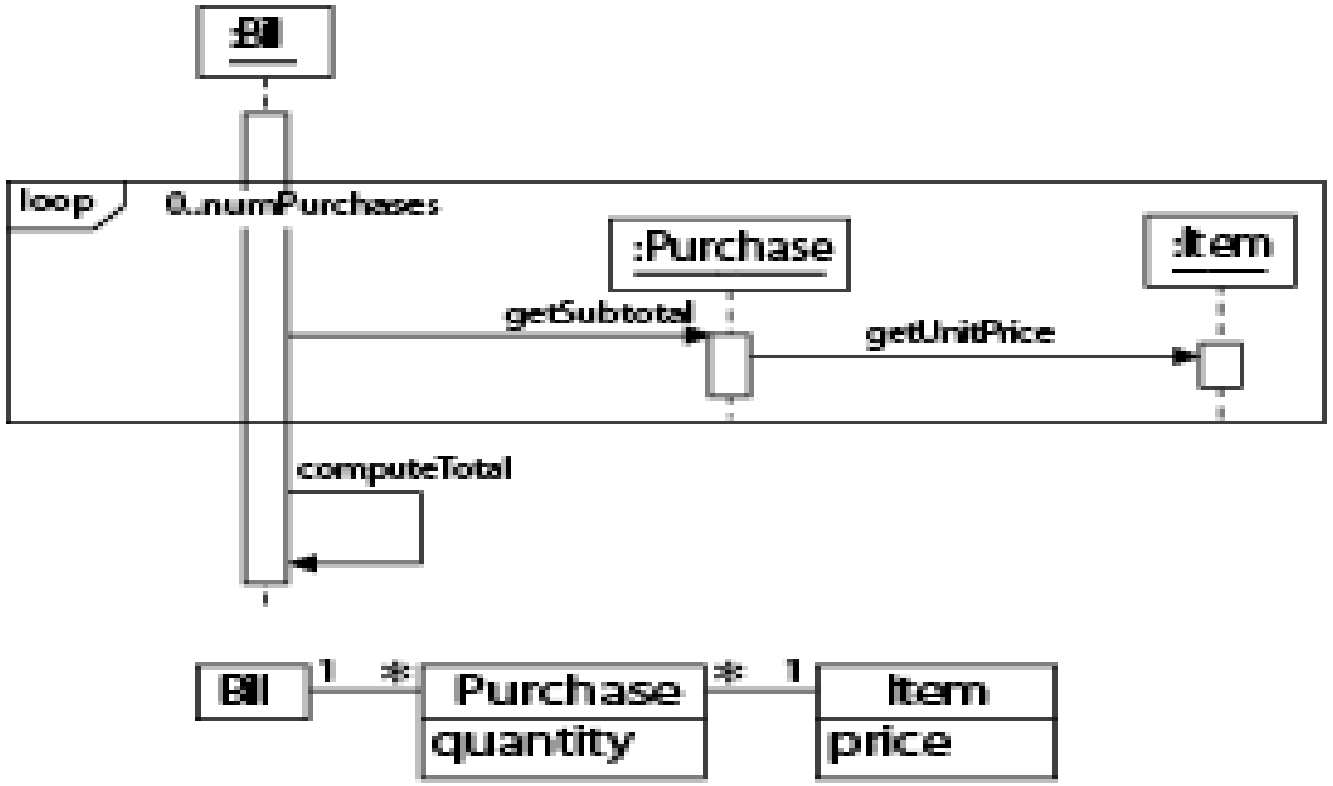
University of Victoria

Sequence diagrams – same example, more details



Sequence diagrams – an example with replicated messages

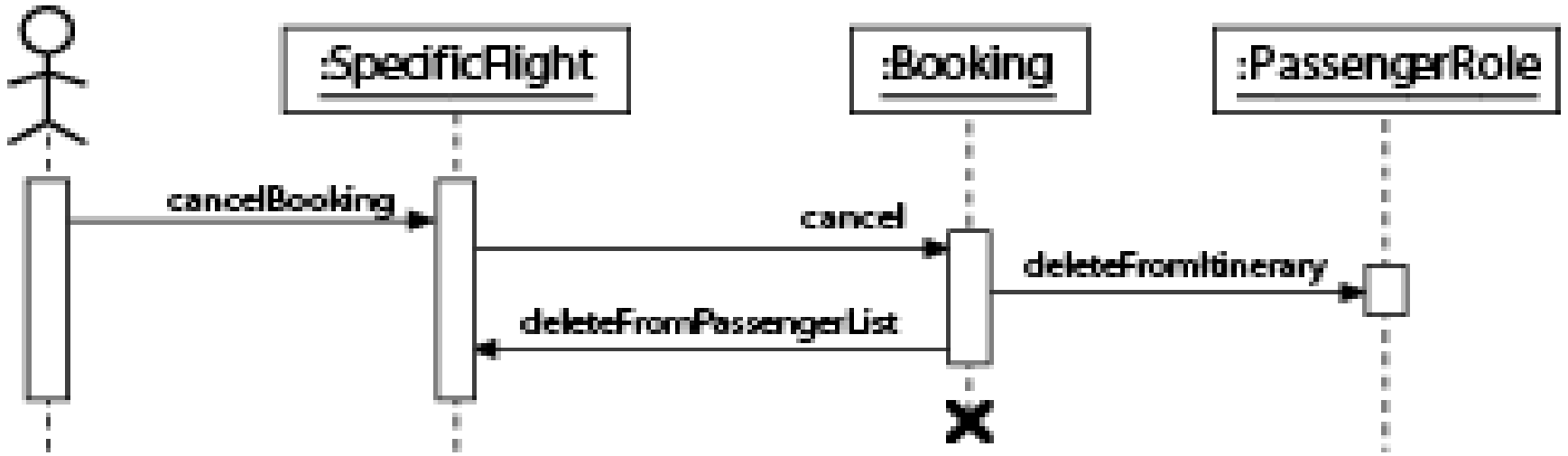
An iteration over objects is indicated by an asterisk preceding the message name



University of Victoria

Sequence diagrams – an example with object deletion

If an object's life ends, this is shown with an X at the end of the lifeline



University of Victoria

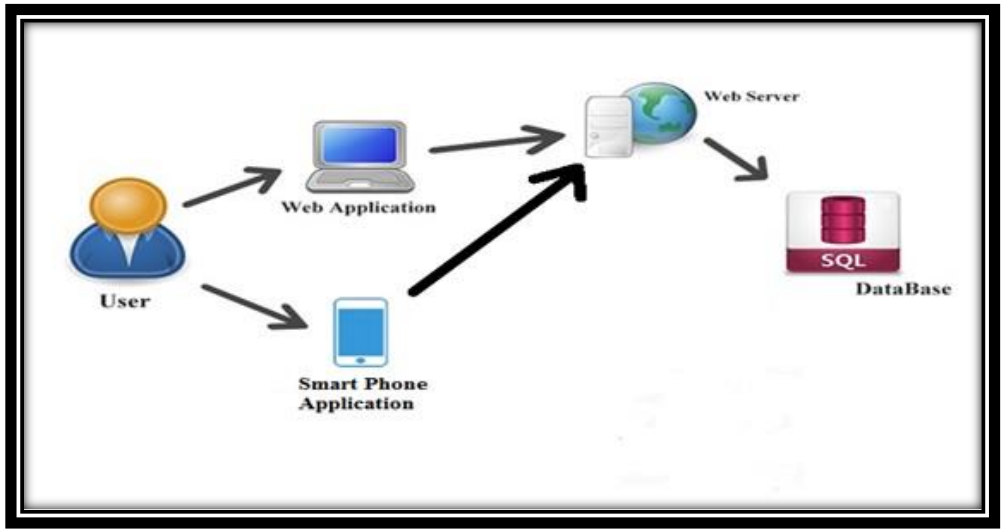
Case Study

University of Victoria

Jens H. Weber

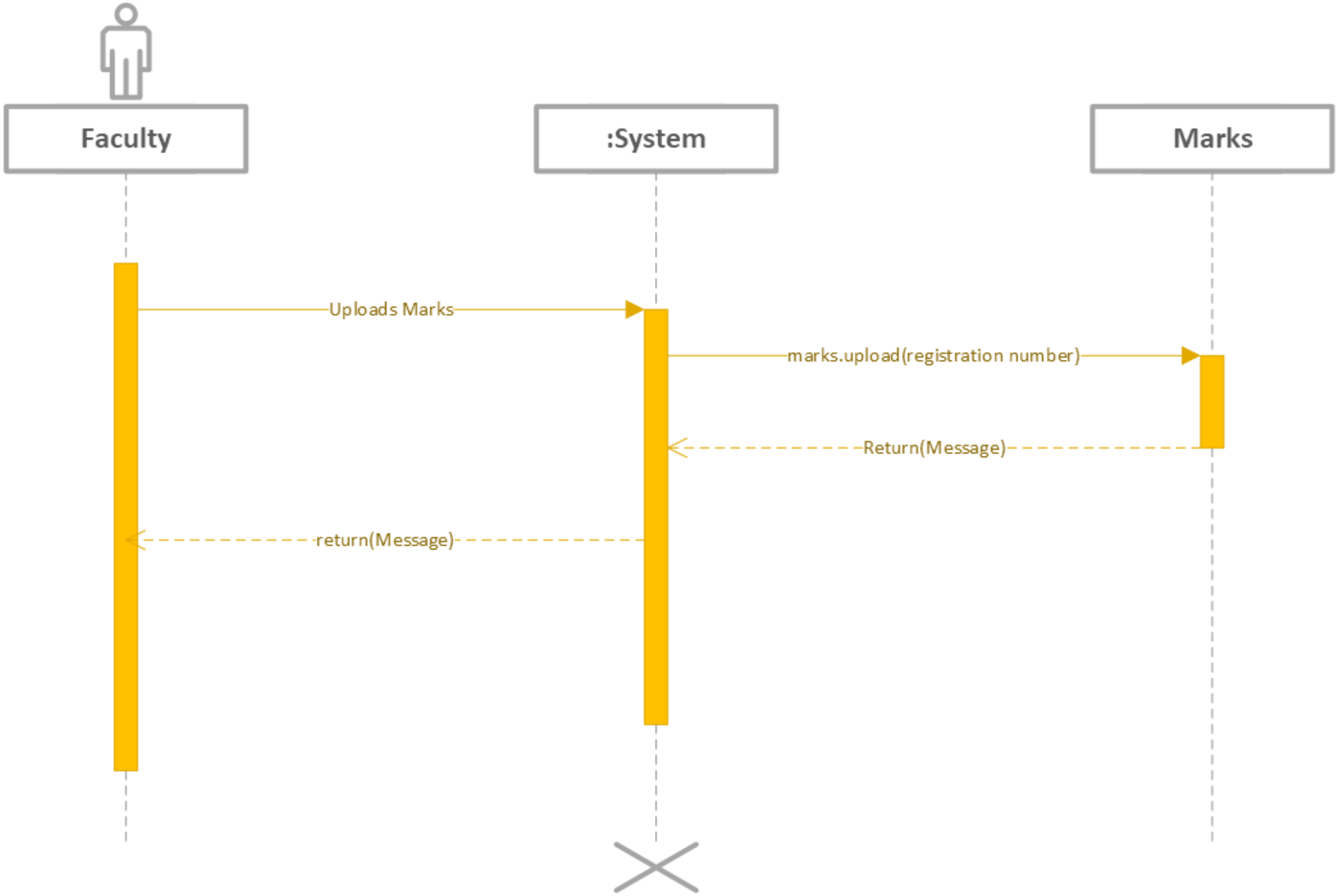
Case Study

Online School Portal (OSP) is a smartphone and web-based application that targets schools. It facilitates students, teachers, administration and parents. This system provides a platform that automates all the work, which is much more efficient than the legacy systems. In this software, each stakeholder can perform certain tasks to get their job done. Moreover, the application provides ease of use, implementing features of human-computer interaction. OSP includes a website and a smartphone application. The smartphone application and website are combined with a mutual server and a database.



University of Victoria

What are the mistakes in this?



Data Flow Diagrams

Data Flow Diagrams

A structured analysis technique employs visual representations of the data moving through the organization, the paths through which the data moves, and the processes that produce, use, and transform data.

Why Data Flow Diagrams?

- Can diagram the **organization** or the **system**
- Can diagram the **current** or **proposed** situation
- Can facilitate **analysis** or **design**
- Provides a good bridge from analysis to design
- Facilitates communication with the user at all stages

University of Victoria

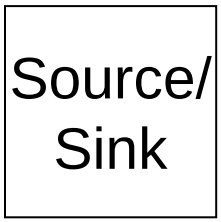
Types of DFDs

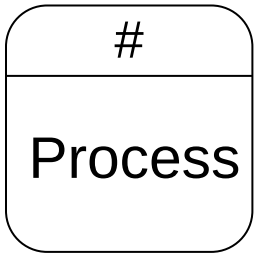
Current - how data flows now
Proposed - how we'd like it to flow
Logical - the "essence" of a process
Physical - the implementation of a process
Partitioned physical - system architecture or high-level design

Levels of Detail

Context level diagram - shows just the inputs and outputs of the system
Level 0 diagram - decomposes the process into the major subprocesses and identifies what data flows between them
Level 1 diagram - increasing levels of detail
Level N diagram - lowest level of decomposition

Four Basic Symbols





University of Victoria

Creating Data Flow Diagrams

Steps:

1. Create a list of activities
2. Construct Context Level DFD
   (identifies external entities and processes)
3. Construct Level 0 DFD
   (identifies manageable sub process)
4. Construct Level 1- n DFD
   (identifies actual data flows and data stores)

University of Victoria

DFD Naming Guidelines

- External Entity → Noun
- Data Flow → Names of data
- Process → verb phrase
  - a system name
  - a subsystem name
- Data Store → Noun

Creating Data Flow Diagrams

Lemonade Stand Example

University of Victoria

Creating Data Flow Diagrams

Example

The operations of a simple lemonade stand will be used to demonstrate the creation of dataflow diagrams.

Steps:

1. Create a list of activities
   - Old way: no Use-Case Diagram
   - New way: use Use-Case Diagram
2. Construct Context Level DFD
   (identifies sources and sink)
3. Construct Level 0 DFD
   (identifies manageable sub processes)
4. Construct Level 1- n DFD
   (identifies actual data flows and data stores)

University of Victoria

Creating Data Flow Diagrams

Example

Think through the activities that take place at a lemonade stand.

1. Create a list of activities

Customer Order
Serve Product
Collect Payment
Produce Product
Store Product

University of Victoria

Creating Data Flow Diagrams

Example

Also think of the additional activities needed to support the basic activities.

1. Create a list of activities

Customer Order
Serve Product
Collect Payment
Produce Product
Store Product
Order Raw Materials
Pay for Raw Materials
Pay for Labor

University of Victoria

Creating Data Flow Diagrams

Example

Group these activities in some logical fashion, possibly functional areas.

1. Create a list of activities

Customer Order
Serve Product
Collect Payment

Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

University of Victoria

Creating Data Flow Diagrams

Example

Create a context level diagram identifying the sources and sinks (users).

Customer Order
Serve Product
Collect Payment

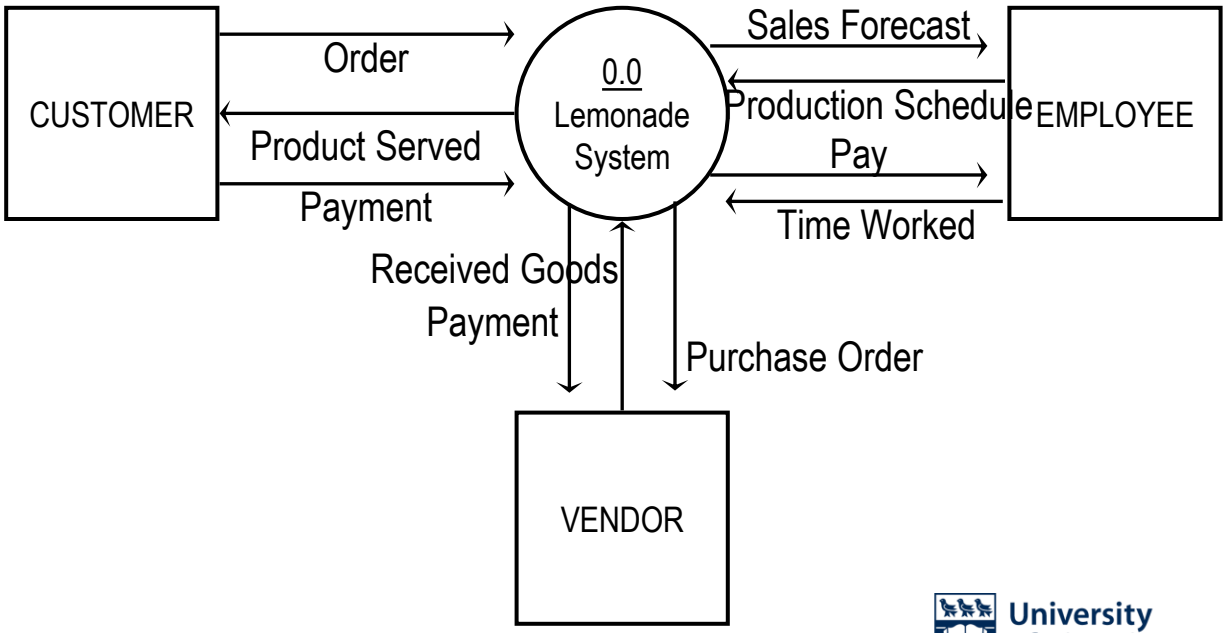
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

2. Construct Context Level DFD (identifies sources and sink)

Context Level DFD



University of Victoria

Creating Data Flow Diagrams

Example

Create a level 0 diagram identifying the logical subsystems that may exist.

Customer Order
Serve Product
Collect Payment

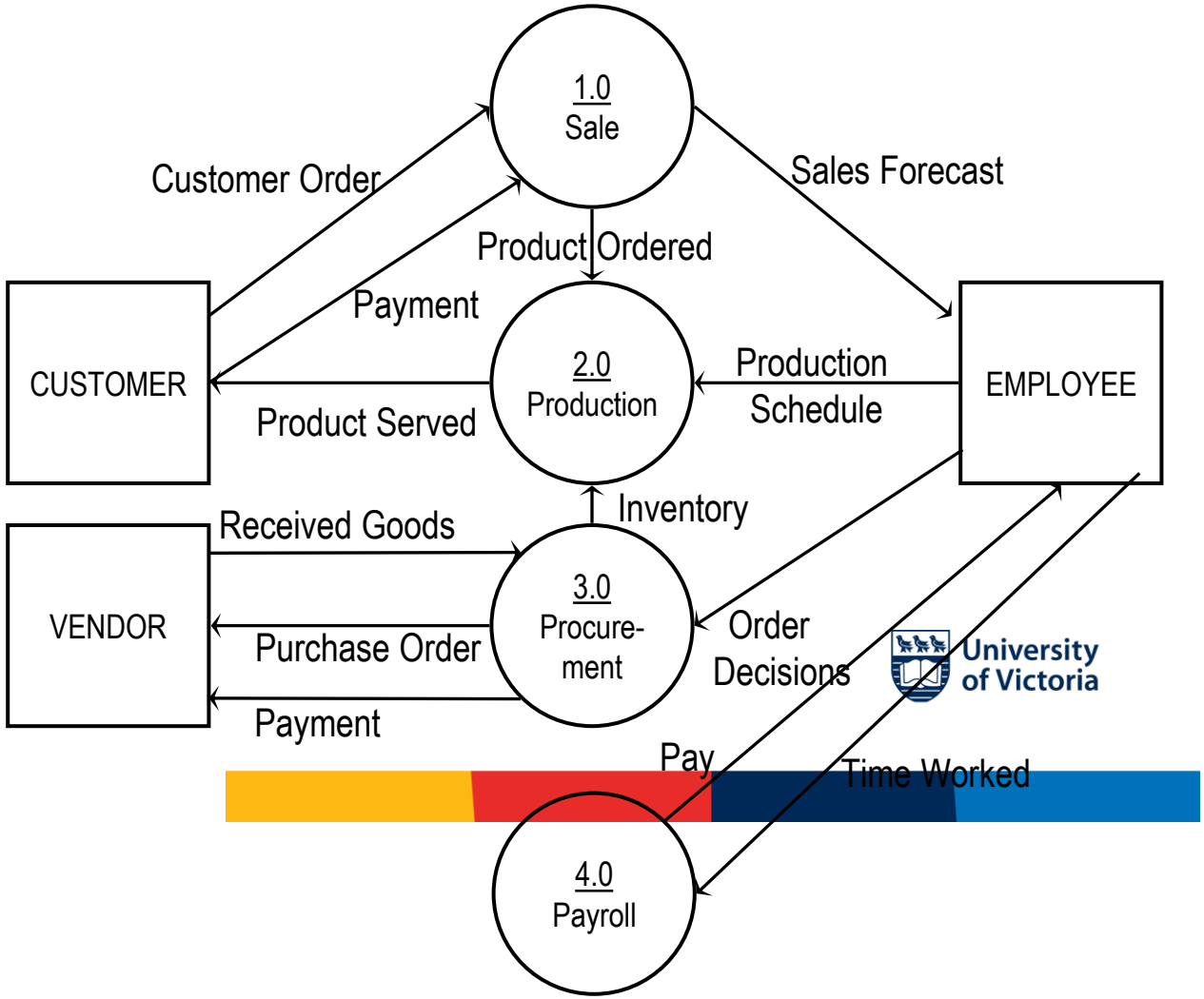
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

3. Construct Level 0 DFD
(identifies manageable sub processes)

Level 0 DFD



University of Victoria

Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

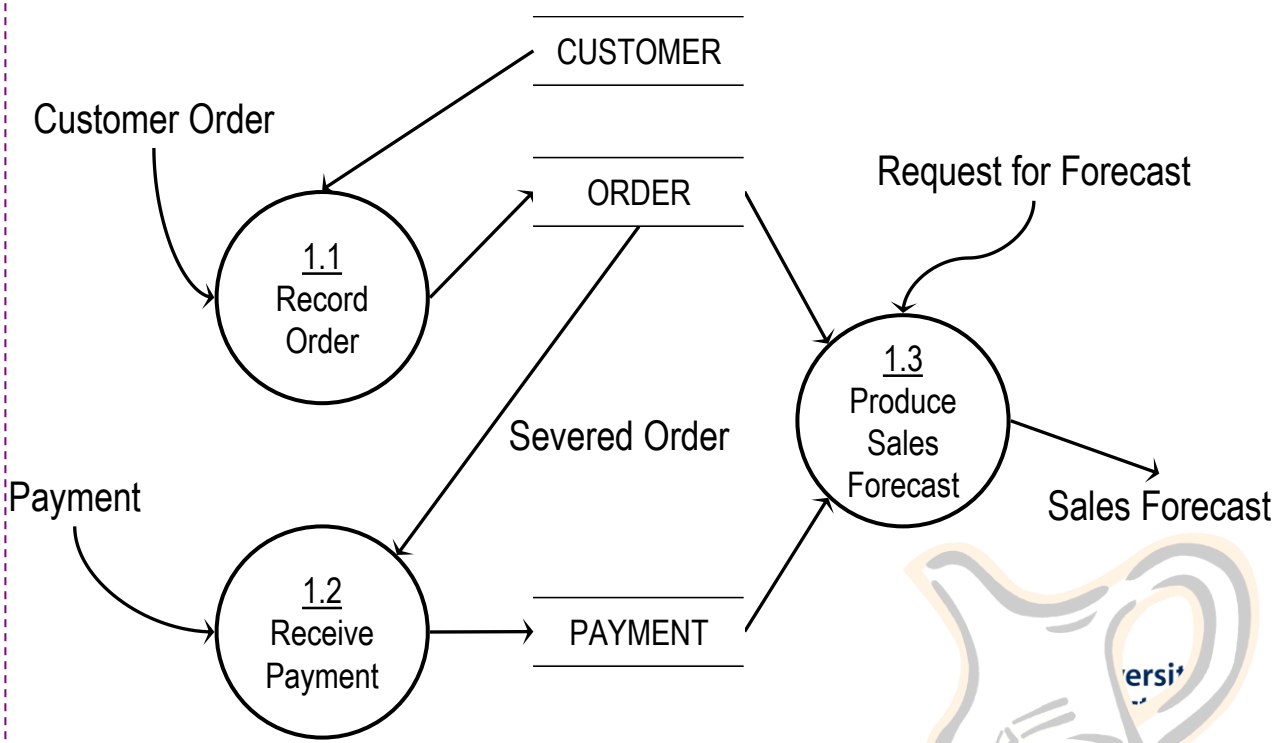
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1- n DFD
(identifies actual data flows and data stores)

Level 1 DFD



Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

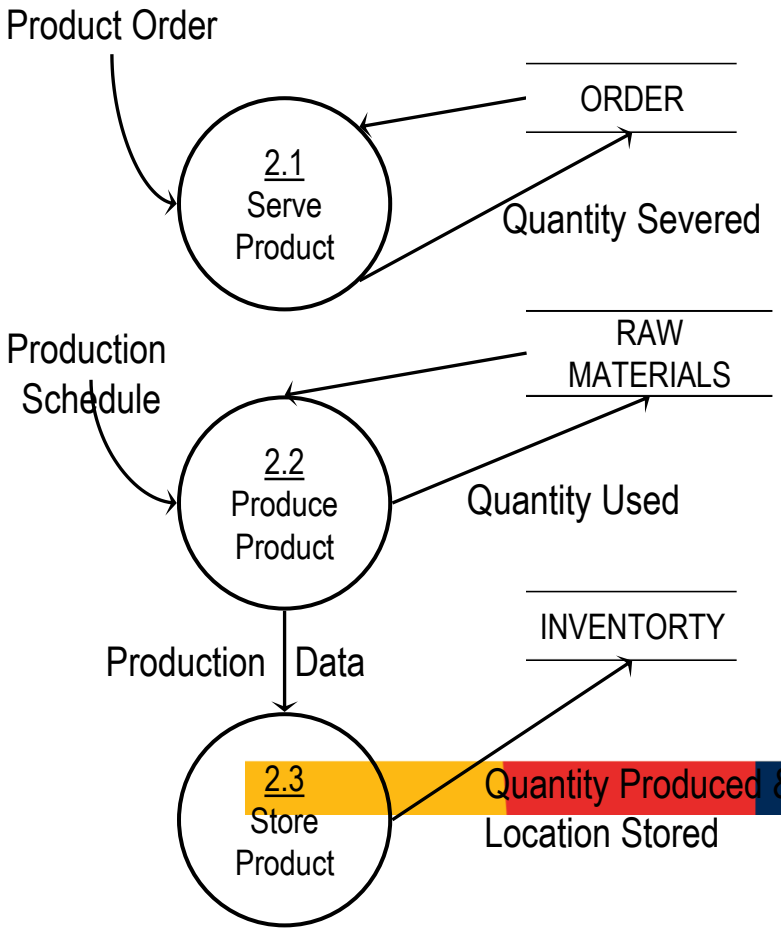
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1 (continued)

Level 1 DFD



Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

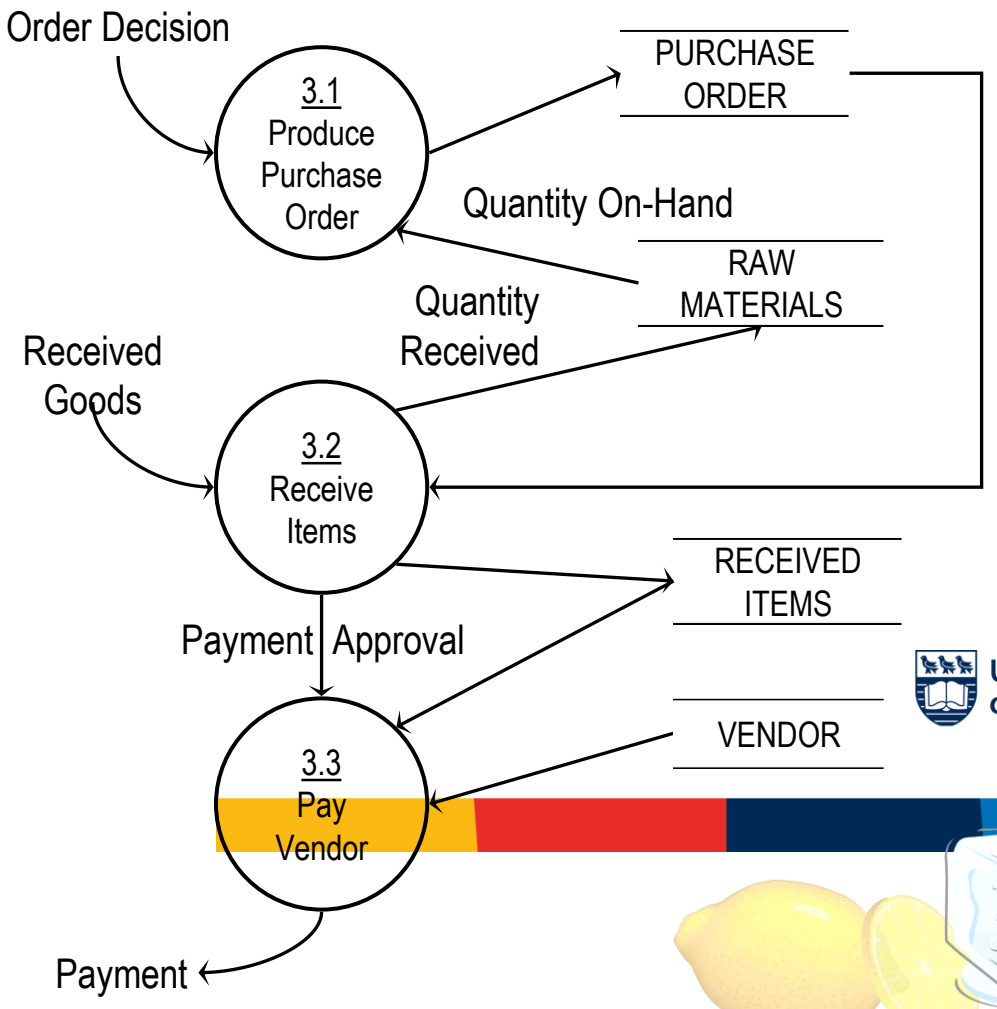
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1 (continued)

Level 1 DFD



Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

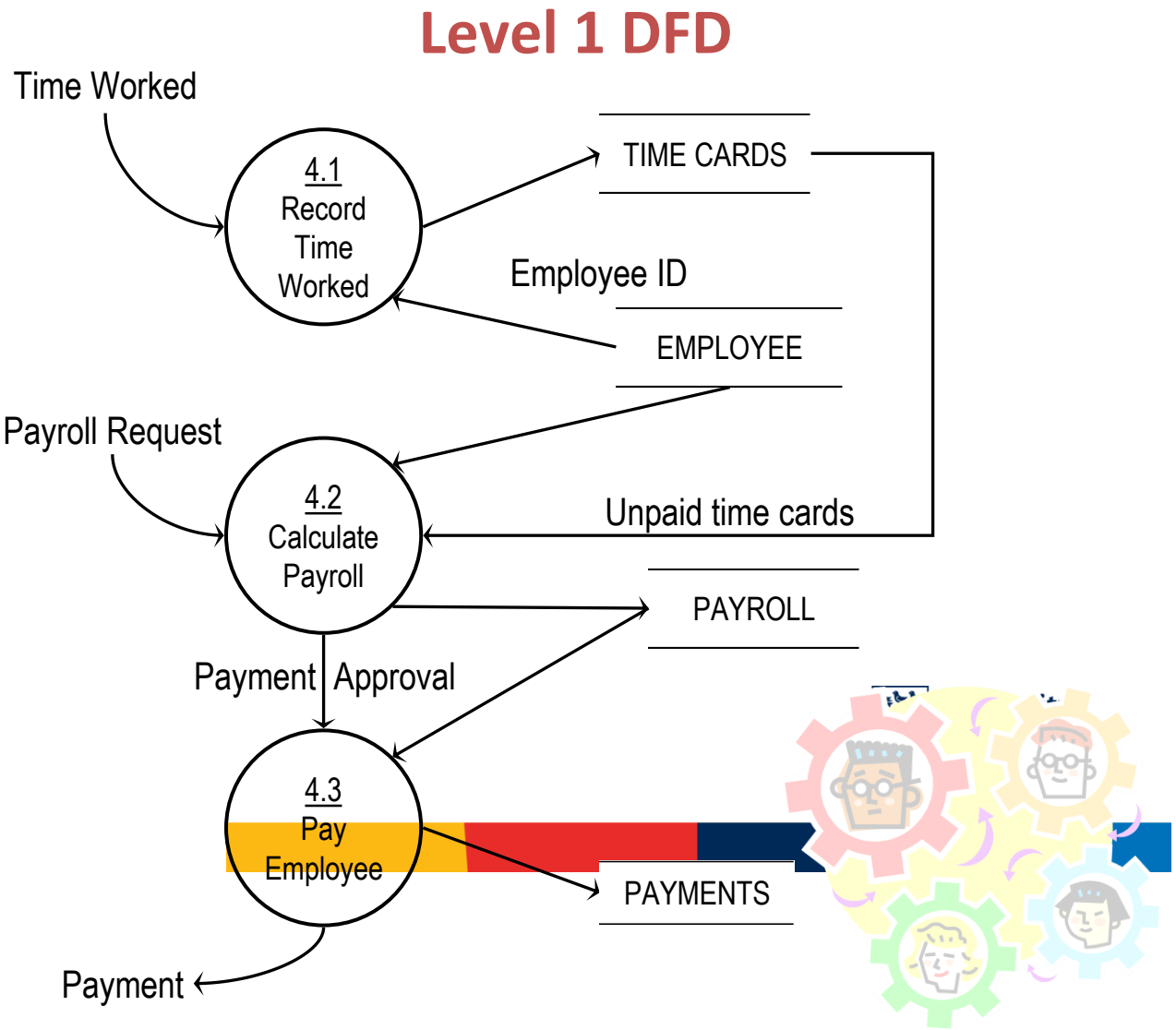
Customer Order
Serve Product
Collect Payment

Produce Product
Store Product

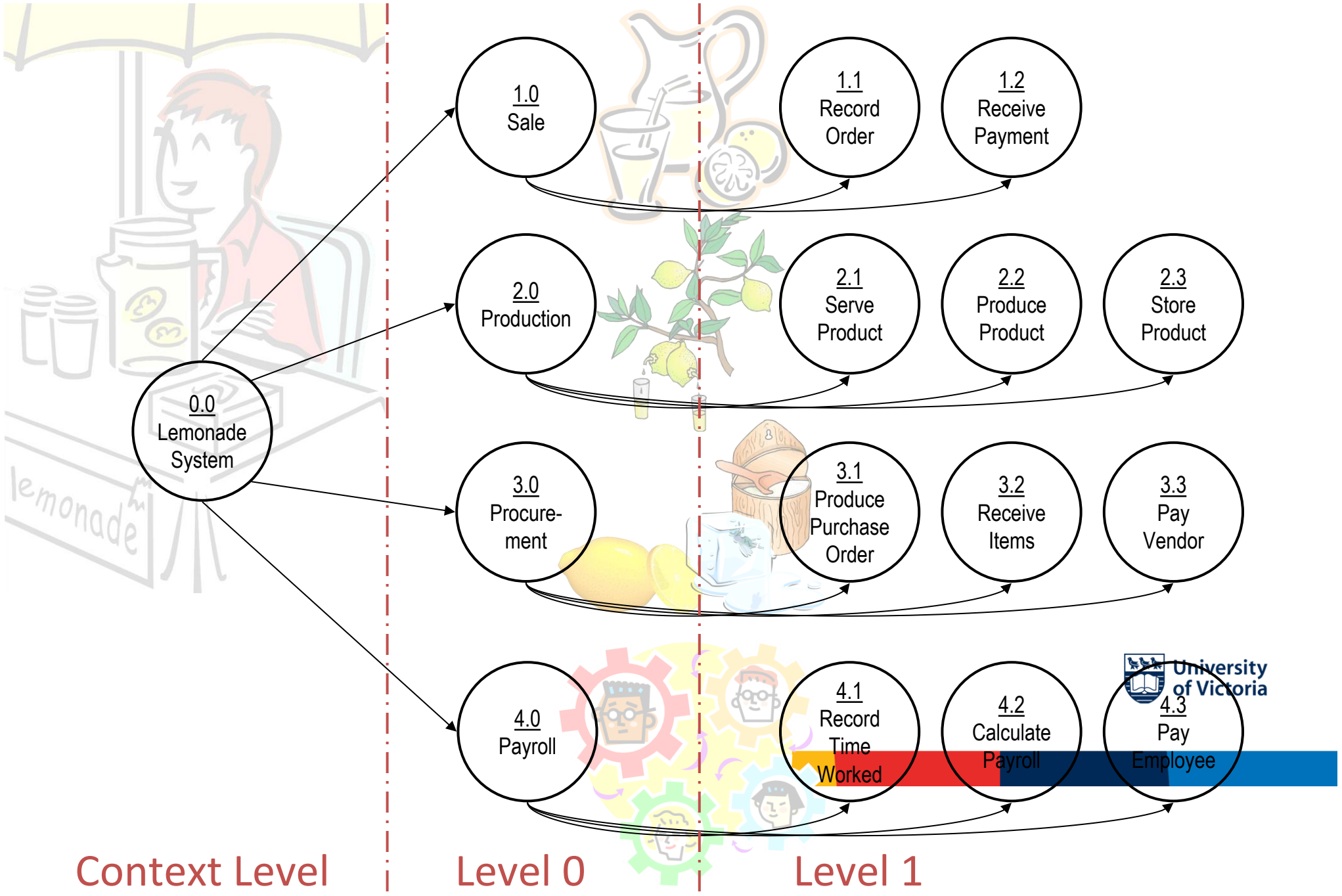
Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1 (continued)



Process Decomposition

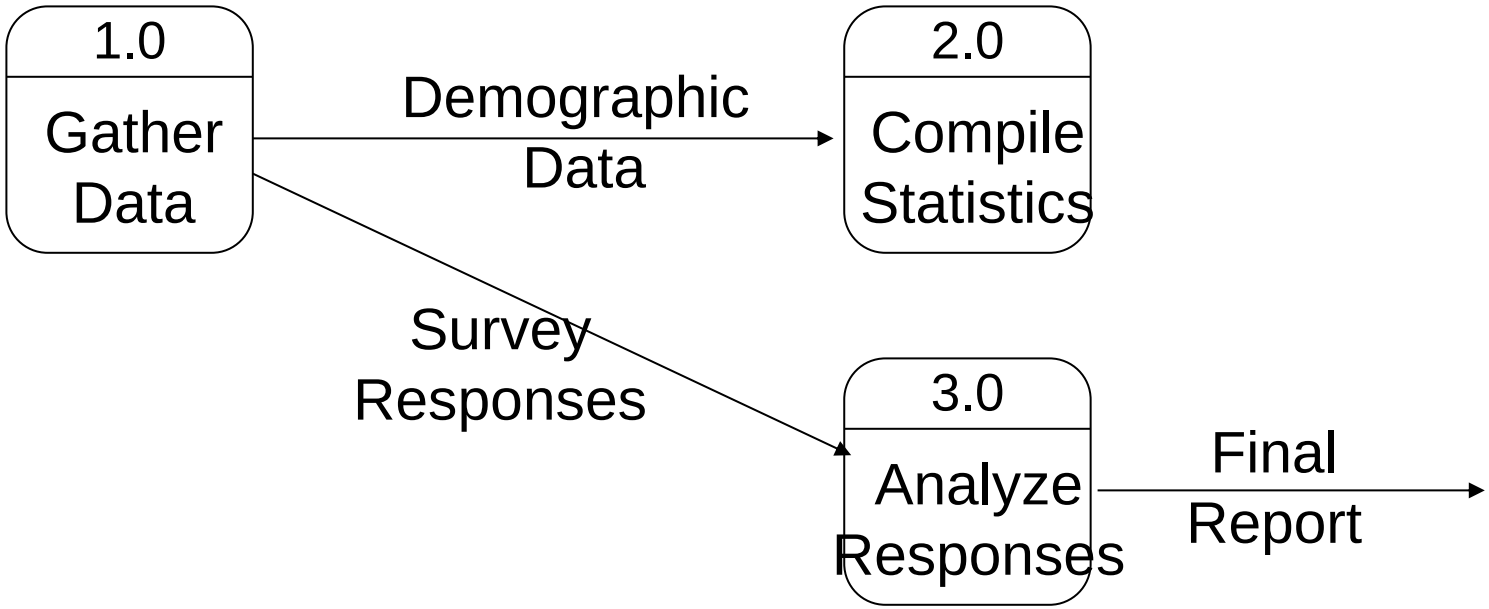


University of Victoria

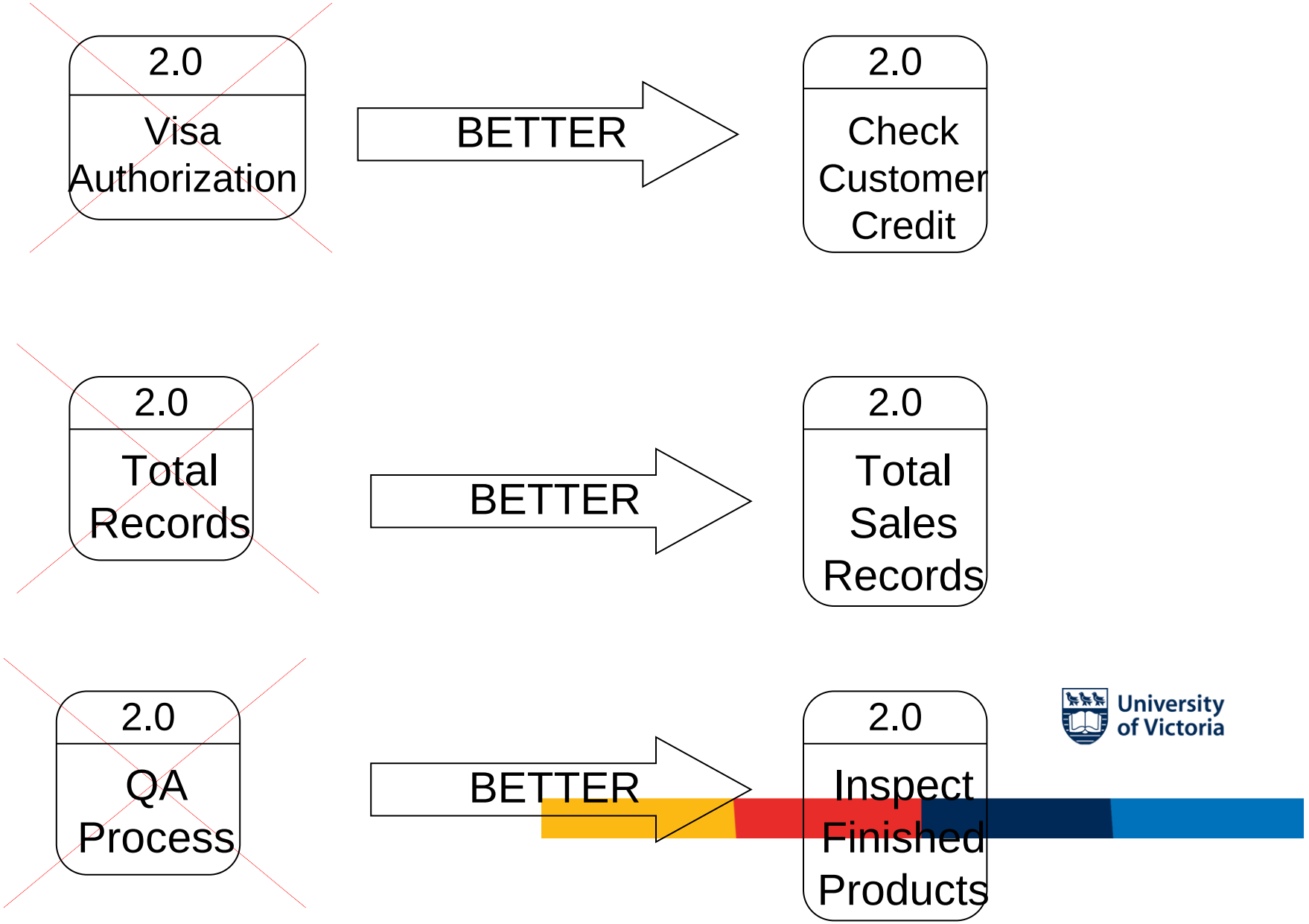
Data Flow Diagramming Rules

Processes

- a process must have at least one input
- a process must have at least one output
- a process name (except for the context level process) should be a verb phrase
  - usually three words: verb, modifier, noun
  - on a physical DFD, could be a complete sentence



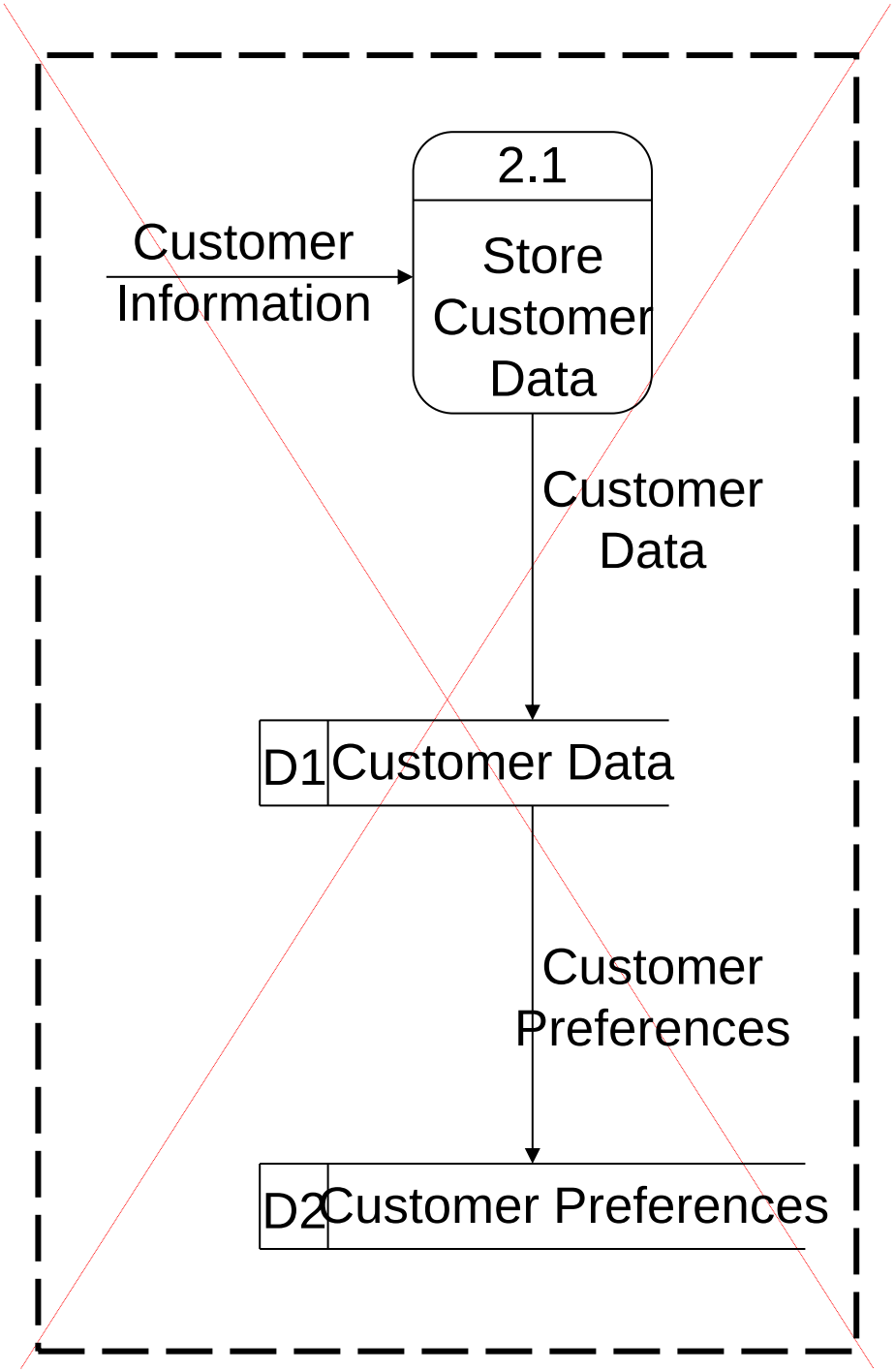
University of Victoria

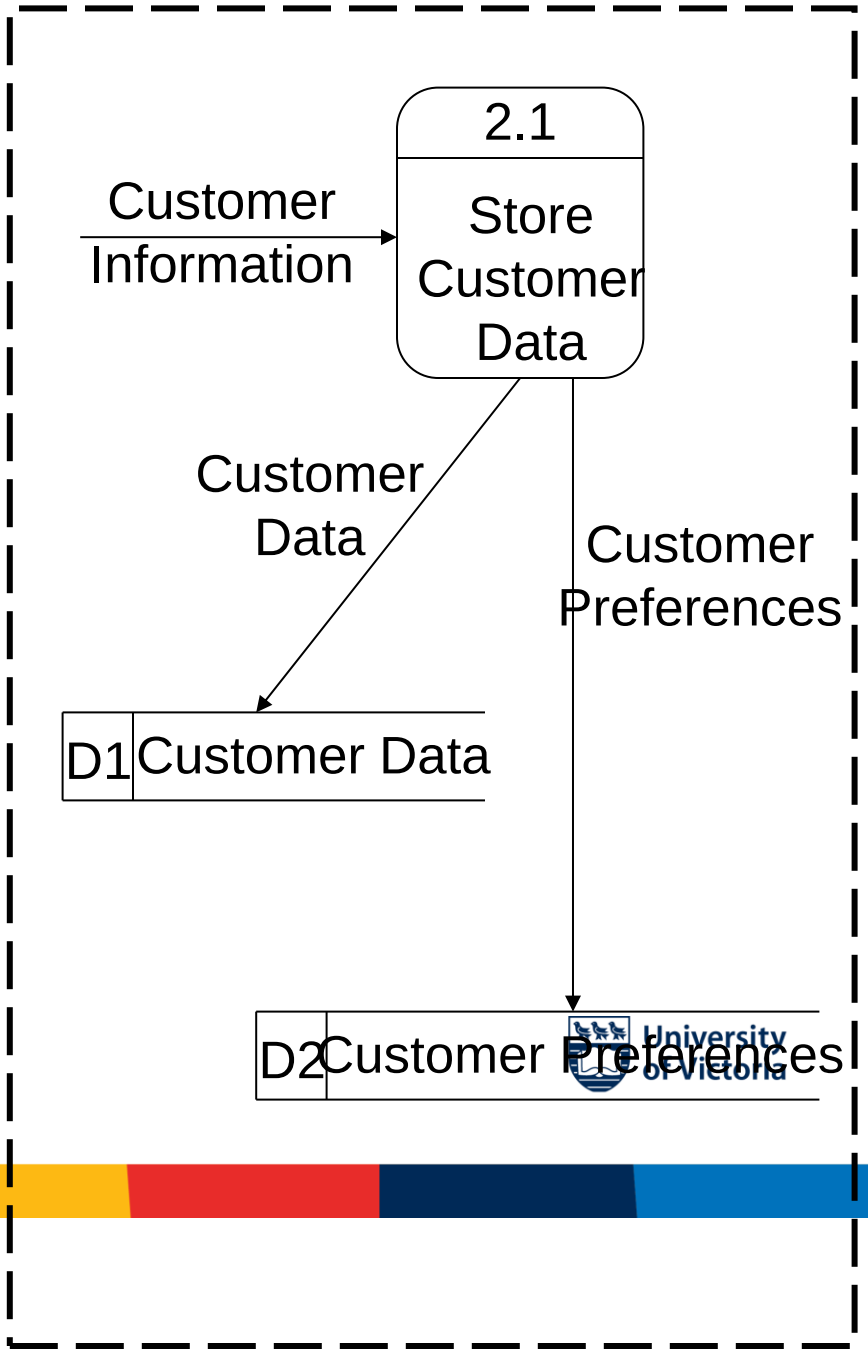

2.0
Visa Authorization
BETTER
2.0
Check Customer Credit
2.0
Total Records
BETTER
2.0
Total Sales Records
2.0
QA Process
BETTER
2.0
Inspect Finished Products
University of Victoria

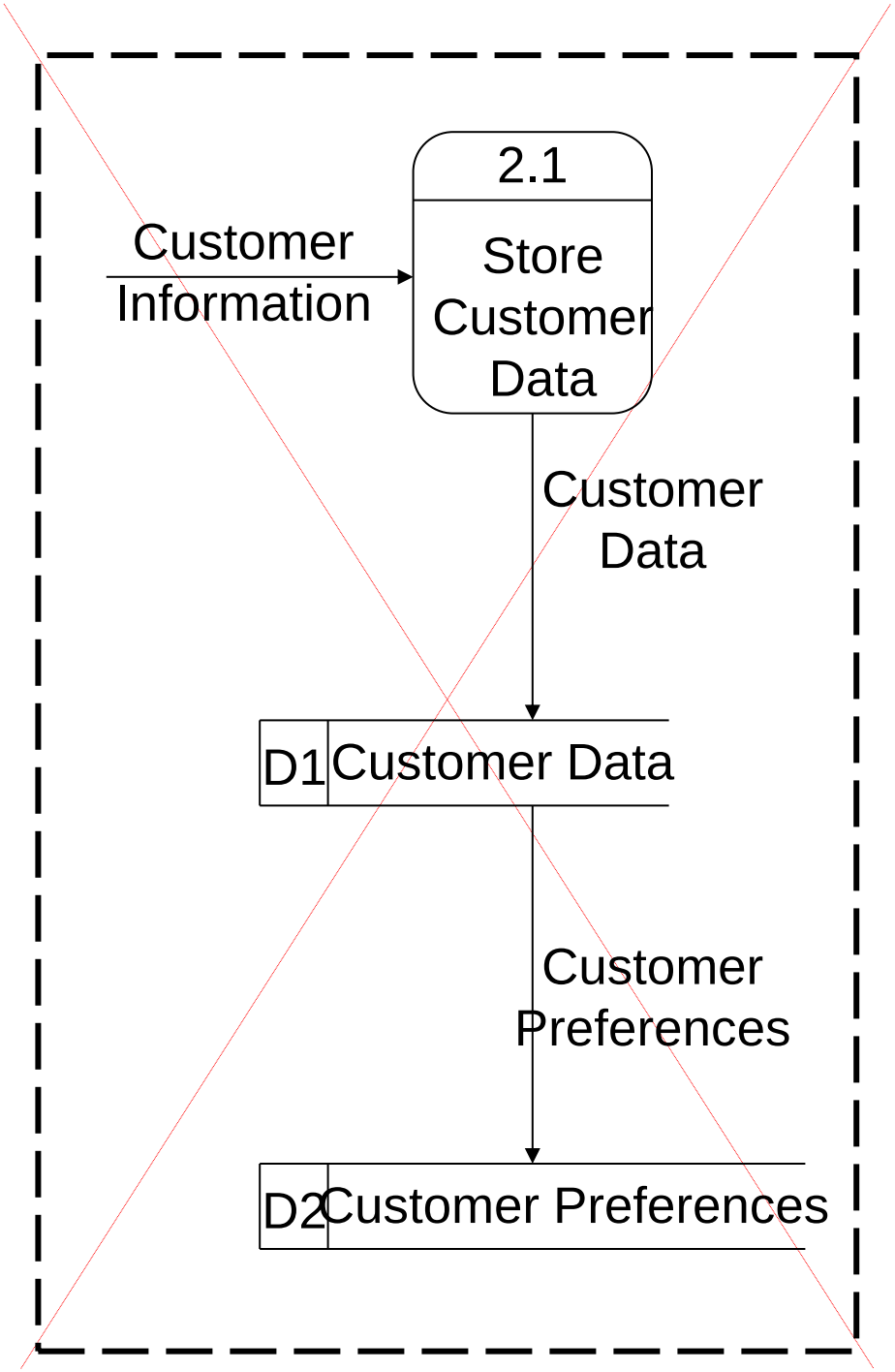
Data Flow Diagramming Rules

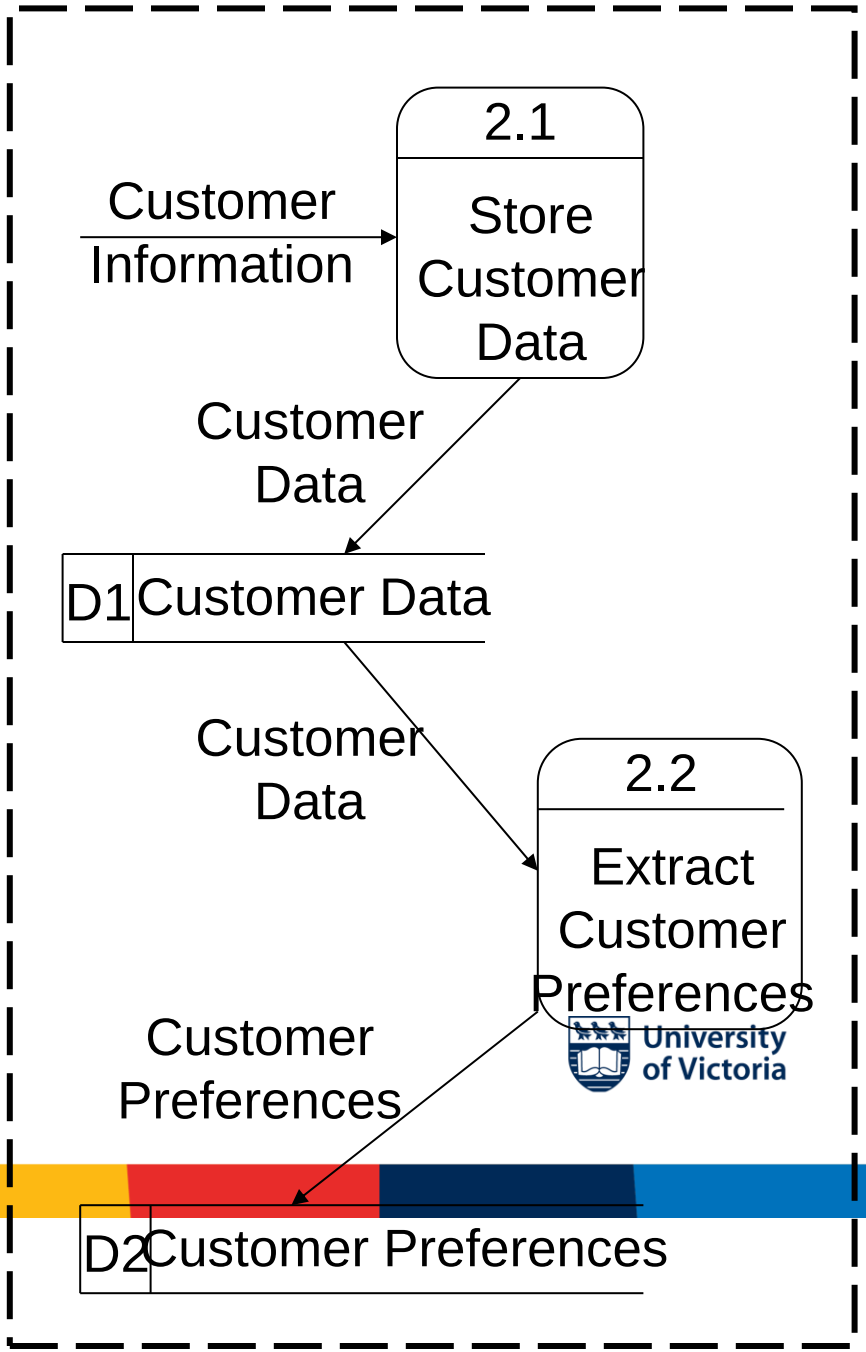
Data stores and sources/sinks

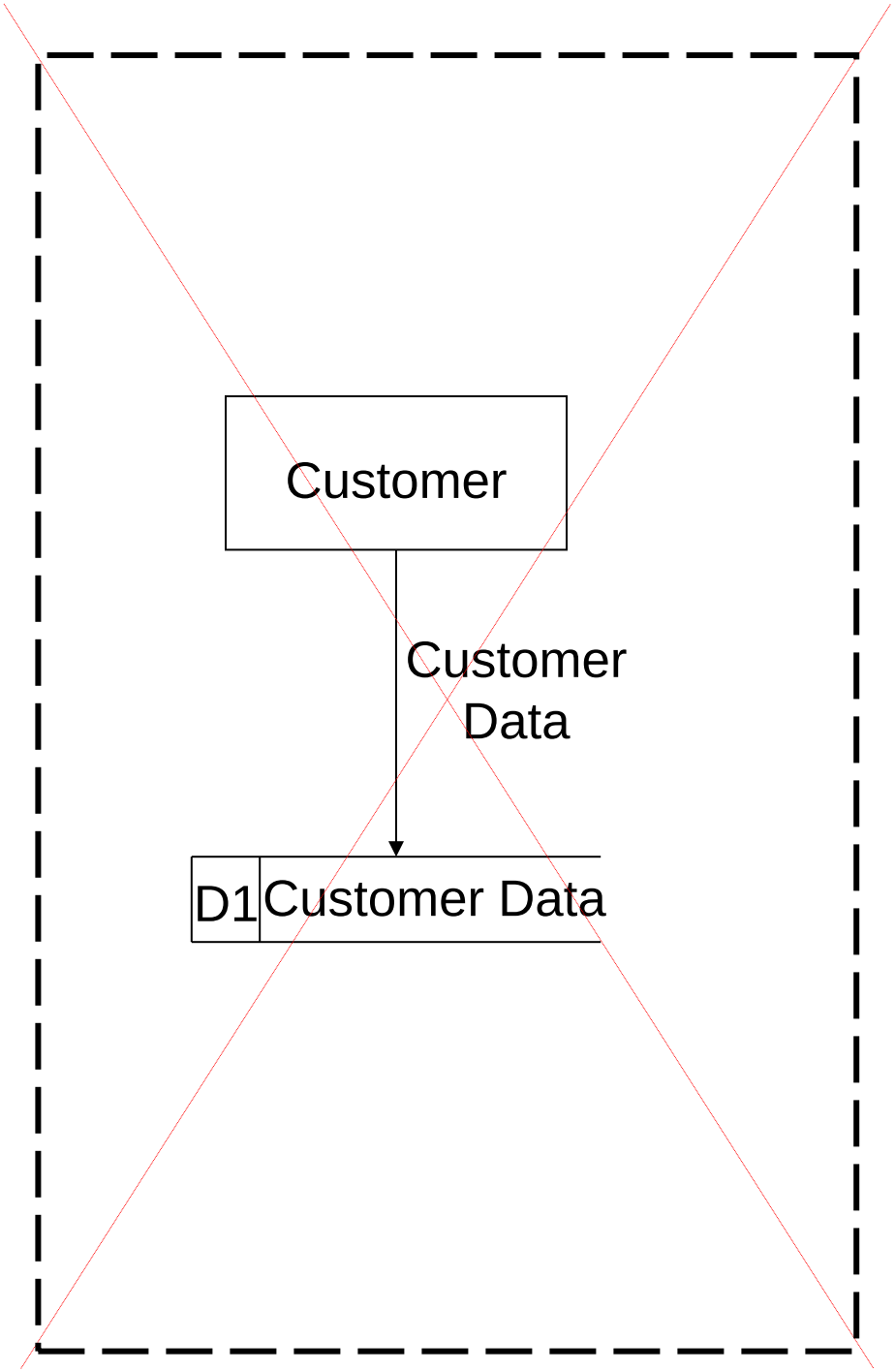
- no data flows between two data stores; there must be a process in between
- no data flows between two sources/sinks
  such a data flow is not of interest, or
  there is a process that moves that data

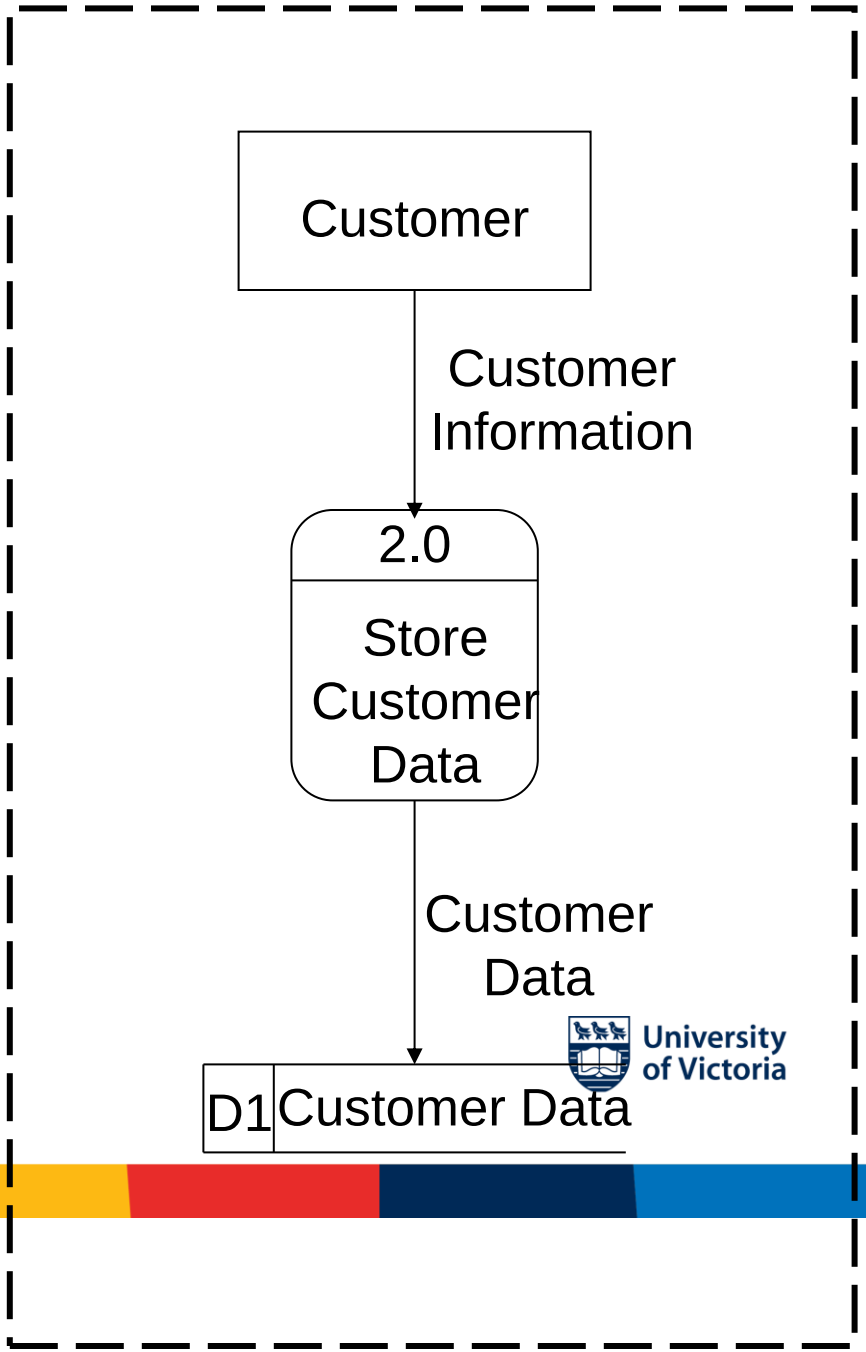

2.1
Store Customer Data
Customer Information
Customer Data
Customer Preferences
D1 Customer Data
D2 Customer Preferences


2.1
Store Customer Data
Customer Information
Customer Data
Customer Preferences
D1 Customer Data
D2 Customer Preferences
University of Victoria

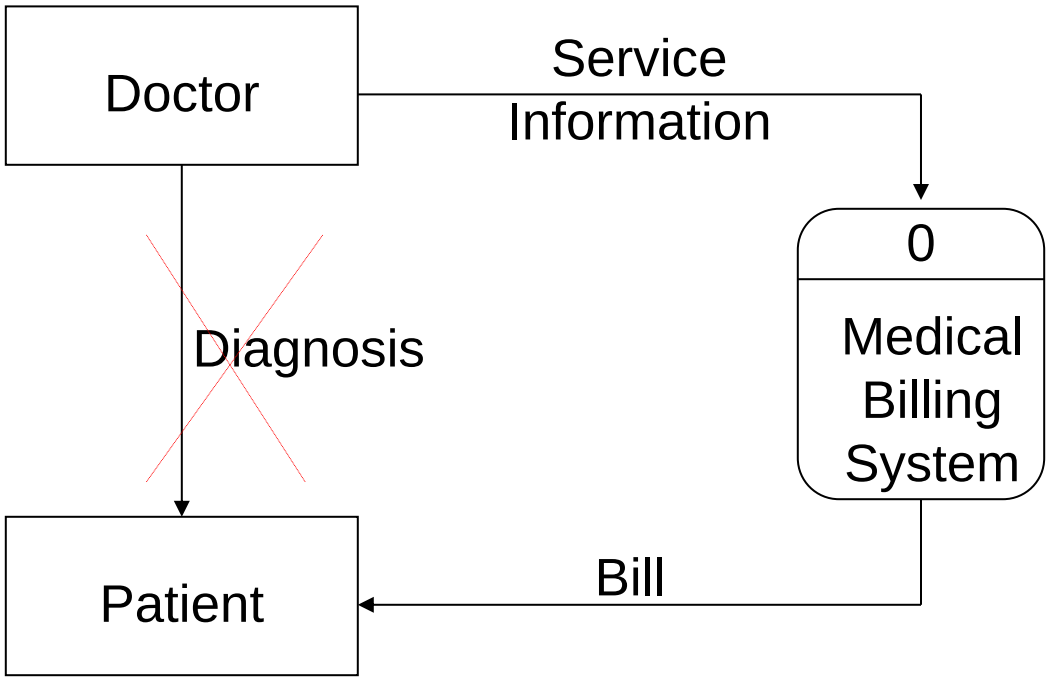

2.1
Customer Information
Store Customer Data
Customer Data
D1 Customer Data
Customer Preferences
D2 Customer Preferences


2.1
Customer Information
Store Customer Data
Customer Data
D1 Customer Data
Customer Data
2.2
Extract Customer Preferences
Customer Preferences
D2 Customer Preferences
University of Victoria





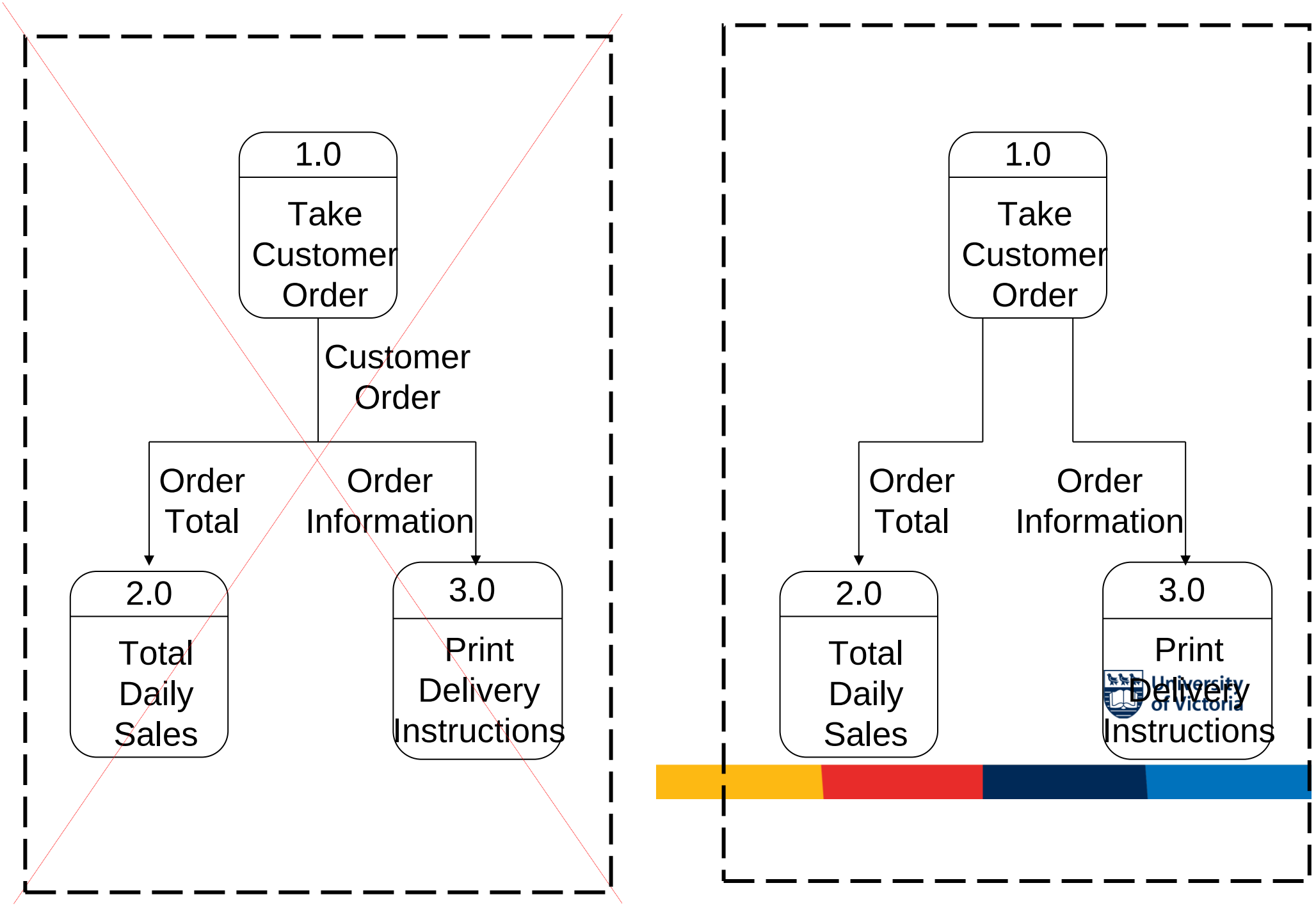
University of Victoria

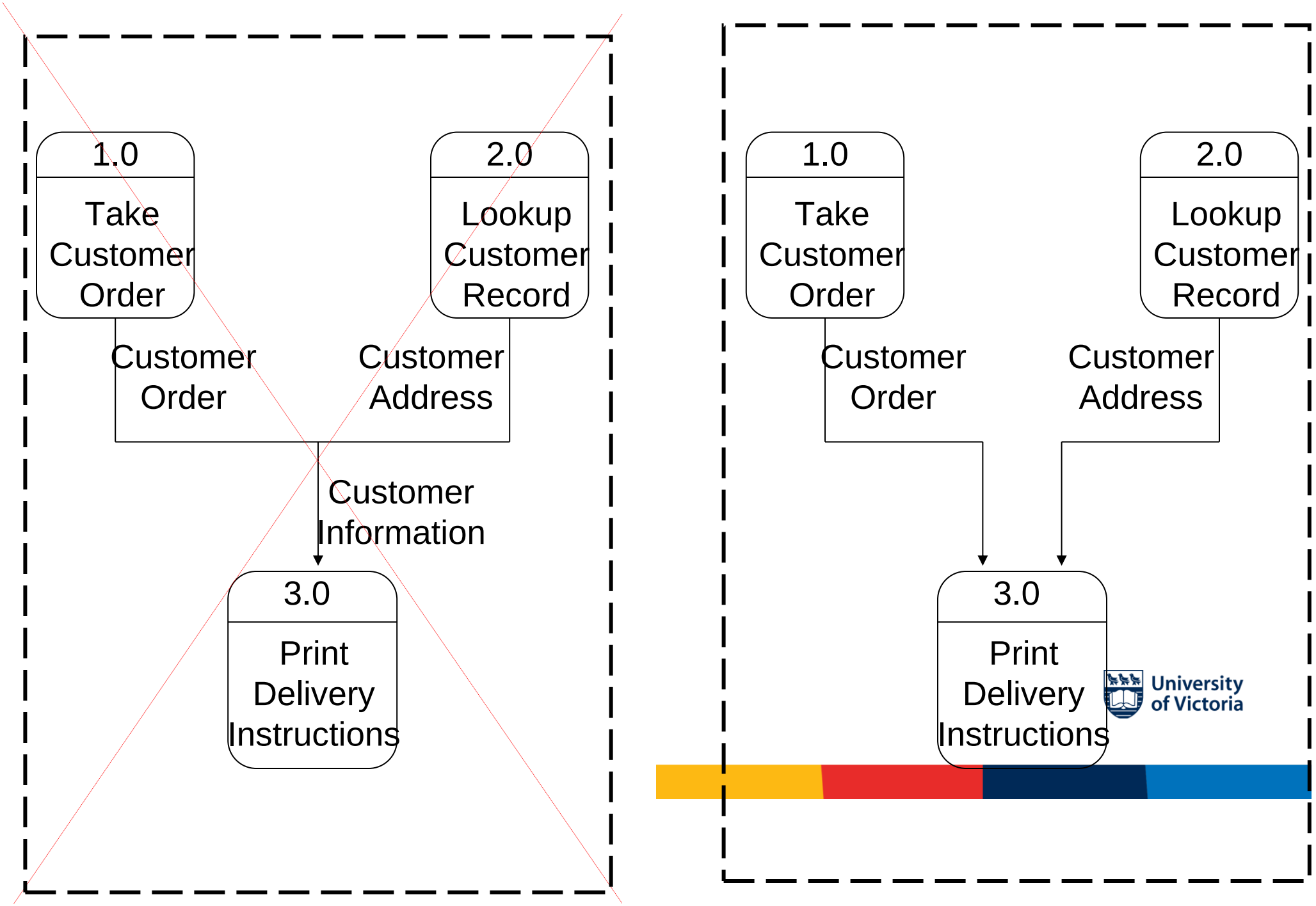


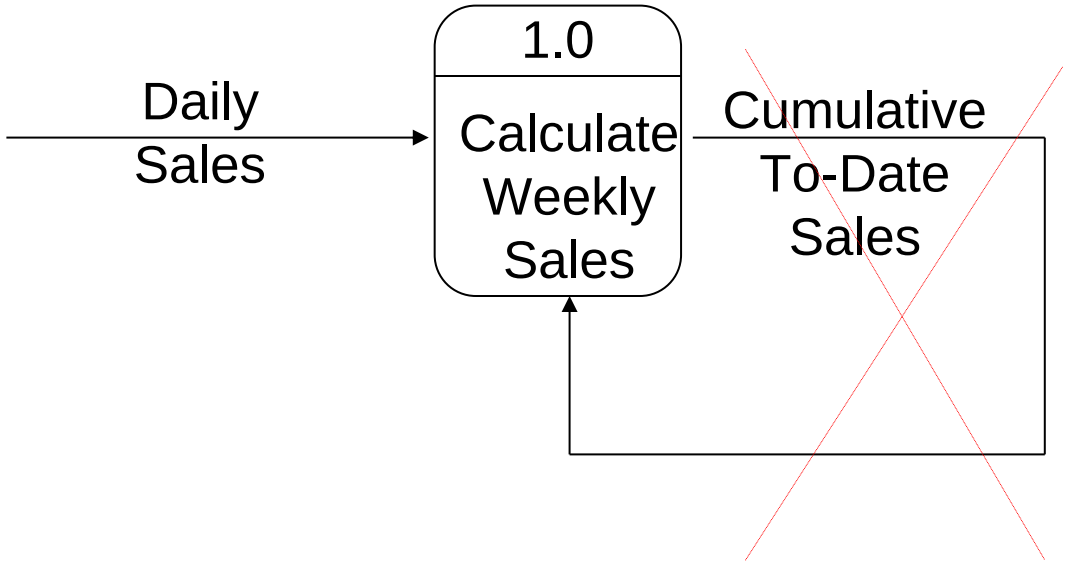
Data Flow Diagramming Rules

Data flows

- data flows are unidirectional
- a data flow may fork, delivering exactly the same data to two different destinations
- two data flows may join to form one only if the original two are exactly the same
- no recursive data flows
- data flows (and data stores and sources/sinks) are labelled with noun phrases


1.0
Take Customer Order
Customer Order
Order Total
Order Information
2.0
Total Daily Sales
3.0
Print Delivery Instructions
1.0
Take Customer Order
Order Total
Order Information
2.0
Total Daily Sales
3.0
Print Delivery Instructions


1.0
Take Customer Order
2.0
Lookup Customer Record
Customer Order
Customer Address
Customer Information
3.0
Print Delivery Instructions
1.0
Take Customer Order
2.0
Lookup Customer Record
Customer Order
Customer Address
3.0
Print Delivery Instructions
University of Victoria

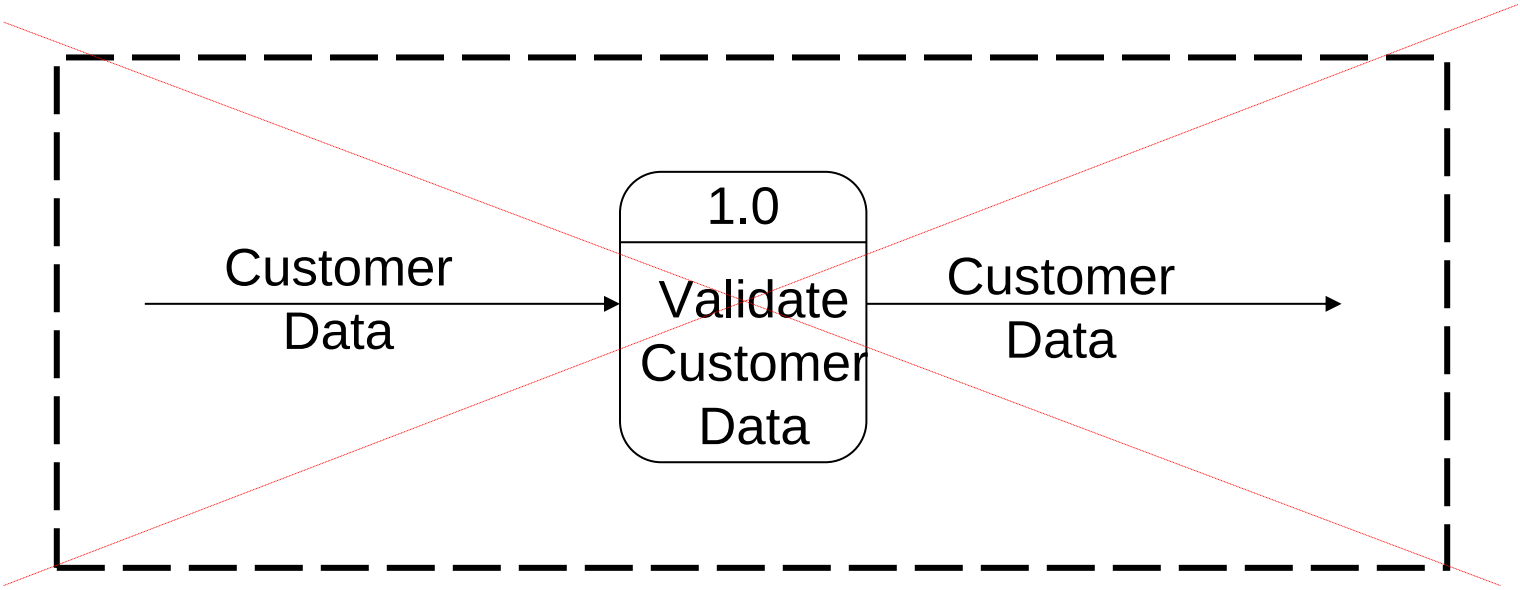


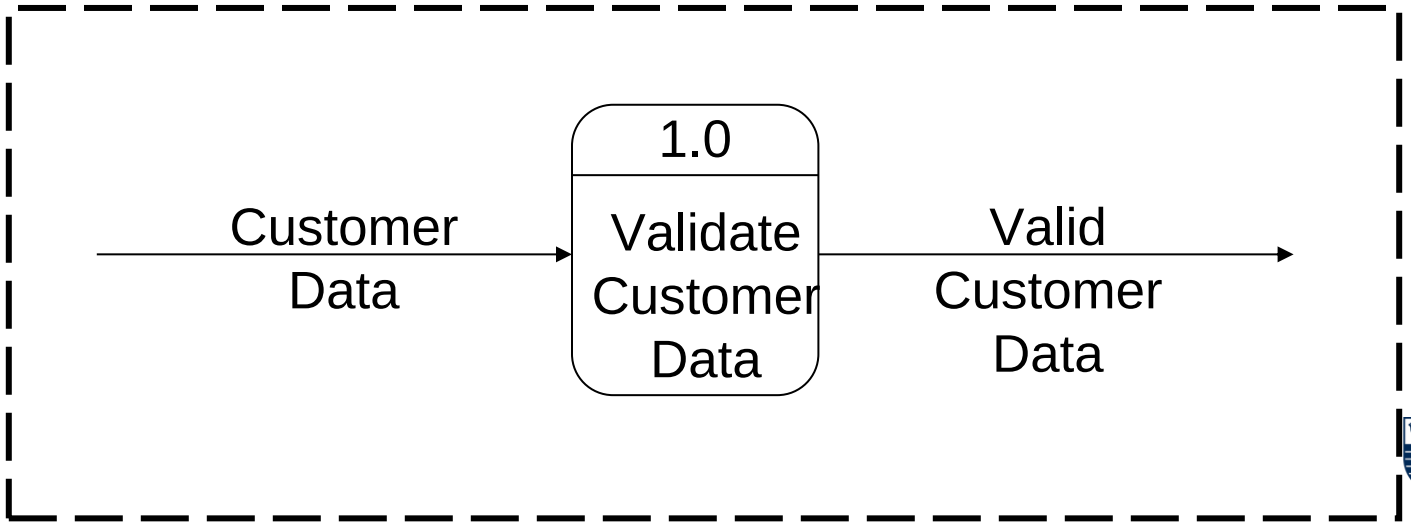
University of Victoria

Data Flow Diagramming Guidelines

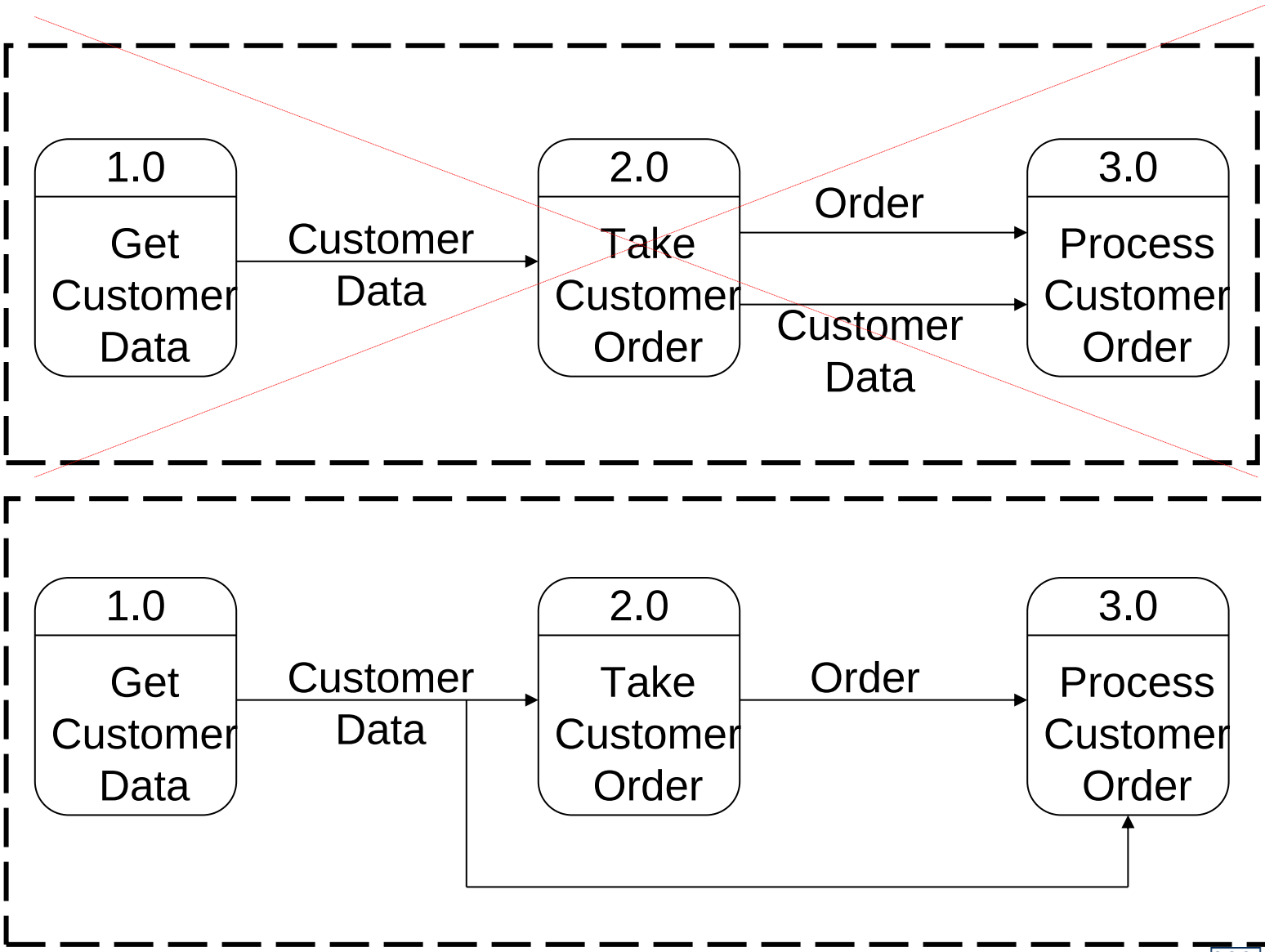
The inputs to a process are different from the outputs

Every object in a DFD has a unique name

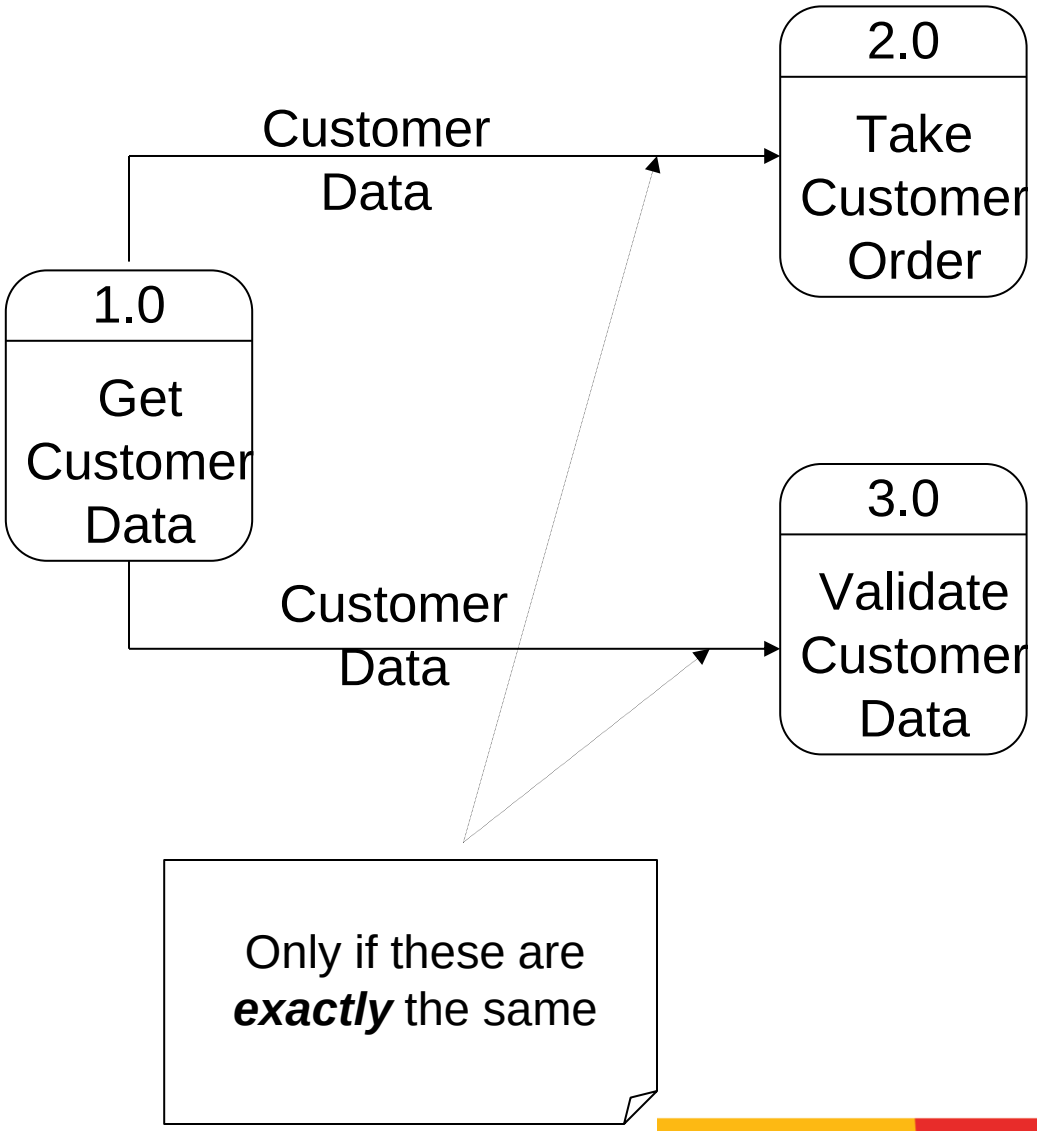




University of Victoria



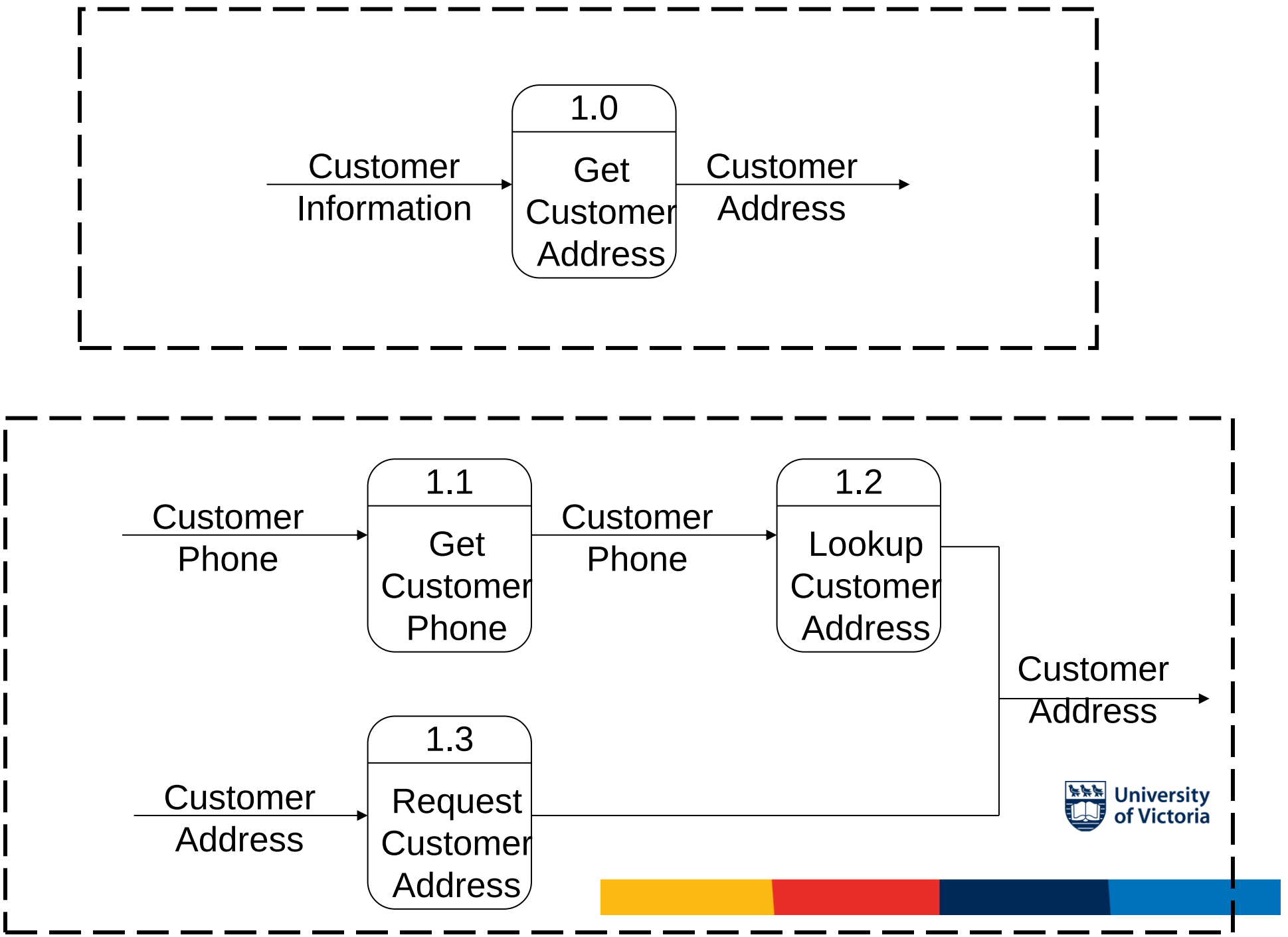
University of Victoria

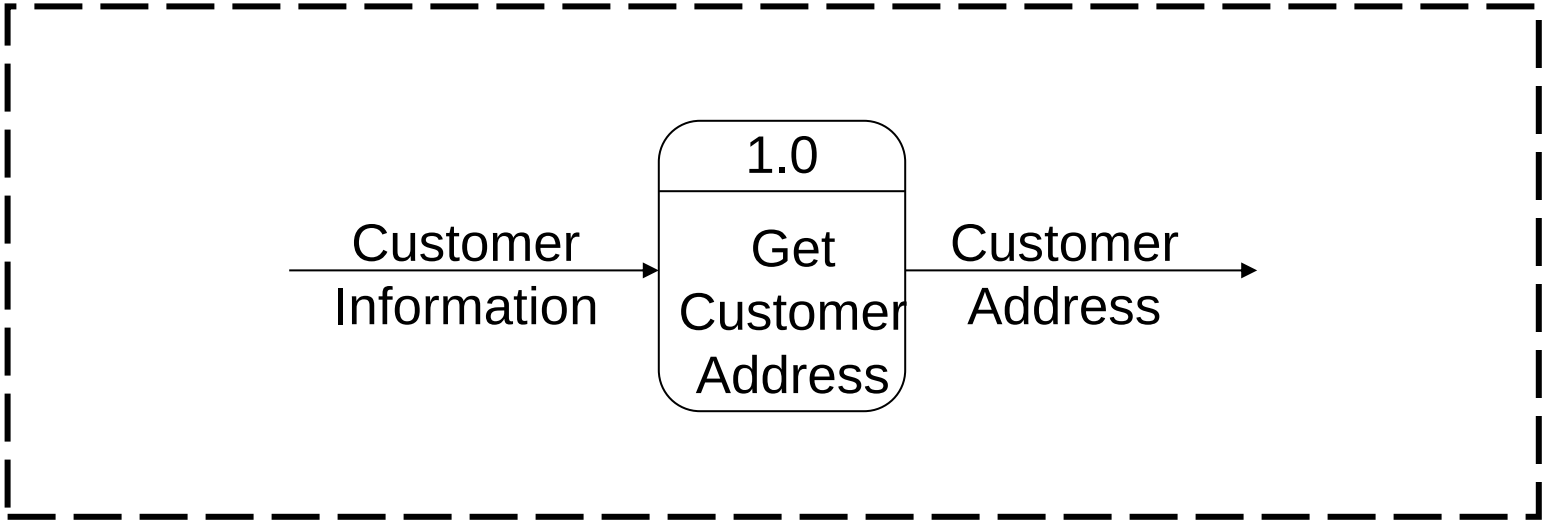


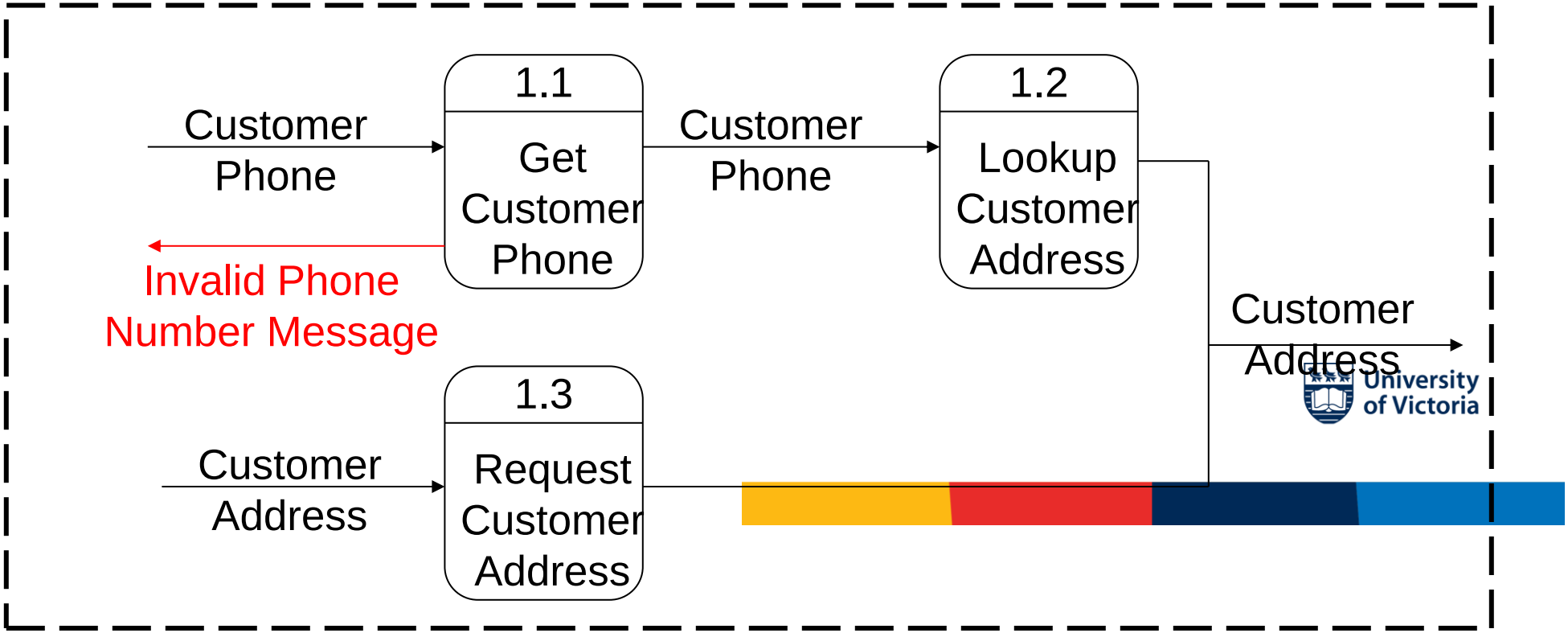
University of Victoria

Data Flow Diagramming Guidelines

- A data flow at one level may be decomposed at a lower level
- All data coming into and out of a process must be accounted for
- On low-level DFDs, new data flows can be added to represent exceptional situations


1.0
Customer Information
Get Customer Address
Customer Address
1.1
Customer Phone
Get Customer Phone
Customer Phone
1.2
Lookup Customer Address
Customer Address
1.3
Customer Address
Request Customer Address
University of Victoria

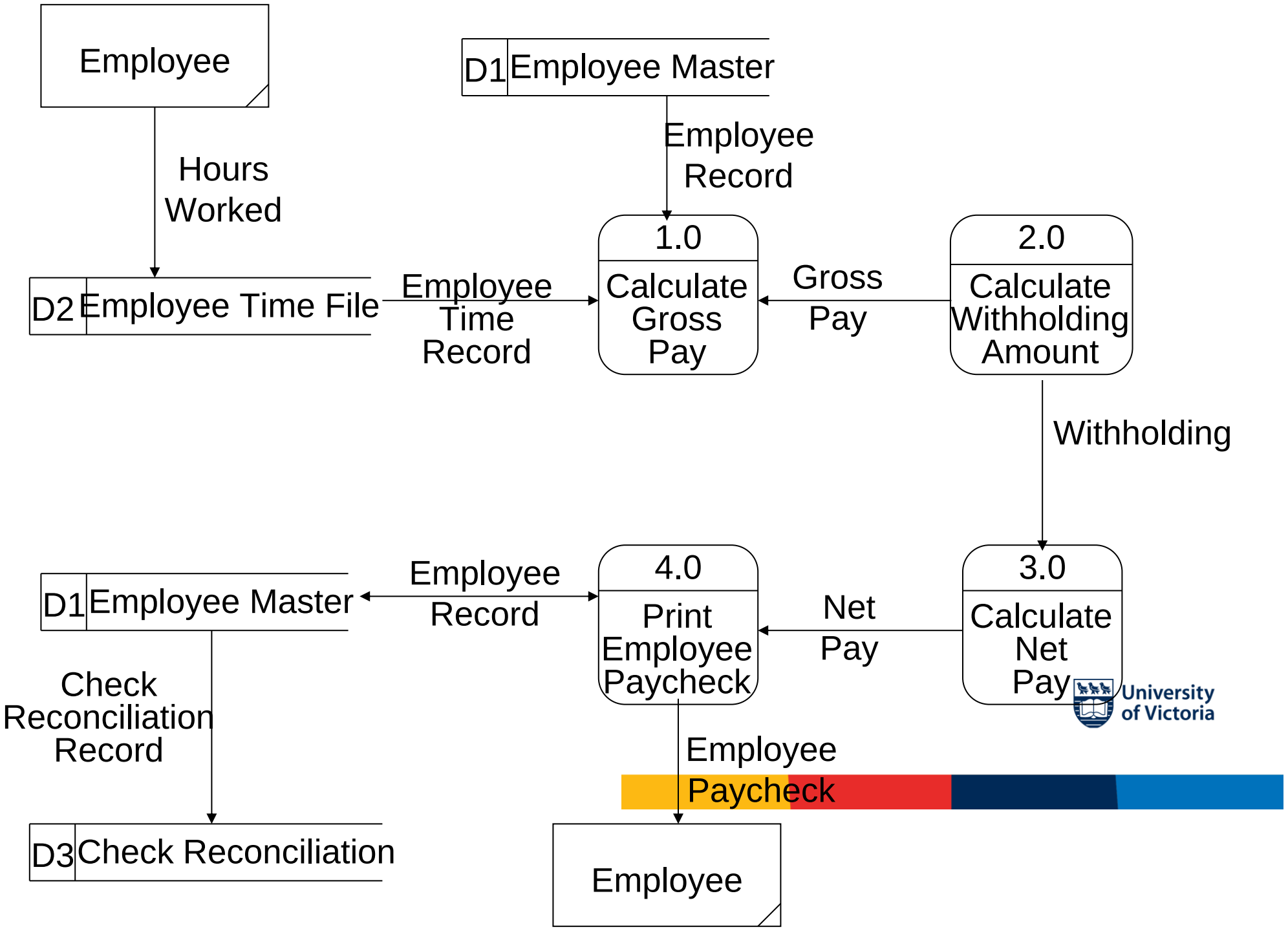


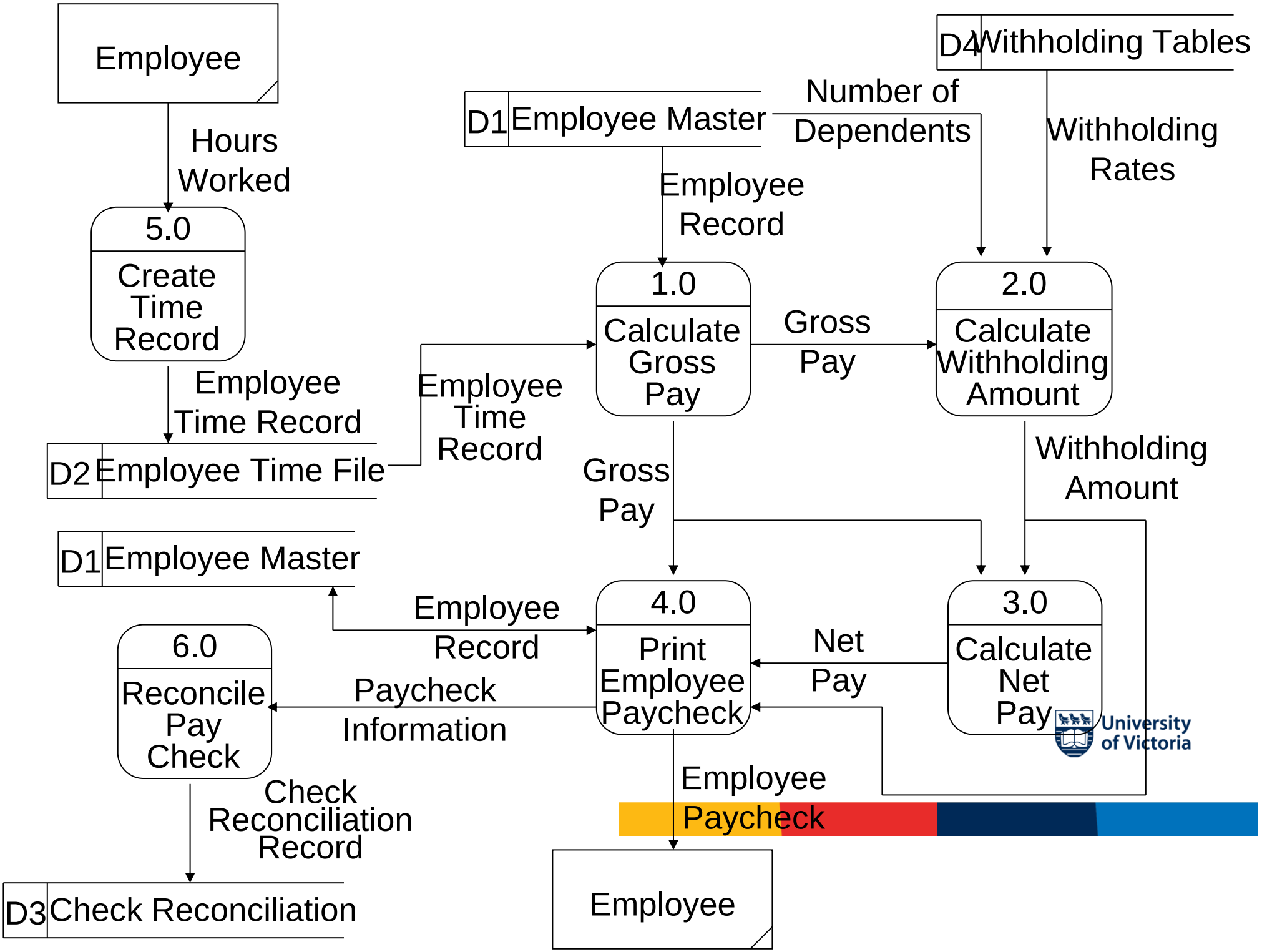


University of Victoria

Data Elements

- Indivisible pieces of data
- Data flows and data stores are made up of data elements
- Like attributes on an ER diagram
- The data elements of a data flow flowing in or out of a data store must be a subset of the data elements in that data store


Employee
Hours Worked
D1 Employee Master
Employee Record
D2 Employee Time File
Employee Time Record
1.0
Calculate Gross Pay
Gross Pay
2.0
Calculate Withholding Amount
Withholding
D1 Employee Master
Employee Record
4.0
Print Employee Paycheck
Net Pay
3.0
Calculate Net Pay
Check Reconciliation Record
D3 Check Reconciliation
Employee Paycheck
Employee


Employee
Hours Worked
5.0
Create Time Record
Employee Time Record
D2 Employee Time File
D1 Employee Master
D4 Withholding Tables
Number of Dependents
Withholding Rates
Employee Record
1.0
Calculate Gross Pay
Employee Time Record
Gross Pay
2.0
Calculate Withholding Amount
Withholding Amount
Gross Pay
D1 Employee Master
Employee Record
4.0
Print Employee Paycheck
3.0
Calculate Net Pay
Net Pay
6.0
Reconcile Pay Check
Paycheck Information
Check Reconciliation Record
D3 Check Reconciliation
Employee Paycheck
Employee
University of Victoria

Upcoming Lecture

- CFD
- ERD
- Component/Package Diagram
- Deployment Diagram

University of Victoria