Data Flow Diagram

Introduction

- DFD is the abbreviation for Data Flow Diagram.
- It is a graphical tool, useful for communicating with users, managers, and other personnel. It is useful for analyzing existing as well as proposed systems.
- DFD represents the flow of data of a system or a process
- It gives insight into the inputs and outputs of each entity in a system and the process itself.

What it does

- A DFD gives an overview of:
  - What data the system processes.
  - What transformations are performed.
  - What data is stored.
  - What results are produced, etc.
- DFDs can be hierarchically organized, which helps in progressively partitioning and analyzing large systems.

Characteristics

The DFD belongs to structured-analysis modeling tools.

- Its purpose: Represent the flow of data within a system.
- Its components: Processes, data stores, data flow, and external entities.
- its Focus: Emphasizes data movement and processing.

A DFD is not a flowchart. flow charts having:

- Their purpose: Illustrate the sequence of steps or activities in a process.
- Their components: Start/end symbols, process boxes, decision diamonds, and connectors.
- Their focus: Emphasizes the flow of control in a process.

Cont.

DFDs also are not part of Unified Modeling Language (UML), DFD predates UML, but they are both useful tools of system analysis and design UML having:

- Their purpose: A standardized modeling language for software engineering.
- Their components: Use case diagrams, class diagrams, sequence diagrams, etc.
- Their focus: Offers a comprehensive set of diagrams for various aspects of software design.

Cont.

- DFDs are commonly used during problem analysis.
- DFDs are quite general and are not limited to problem analysis for software requirements specification.
- DFDs are very useful in understanding a system and can be effectively used during analysis.
- It views a system as a function that transforms the inputs into desired outputs.
- The DFD aims to capture the transformations that take place within a system to the input data so that eventually the output data is produced.
- The processes are shown by named circles and data flows are represented by named arrows entering or leaving the bubbles.
- A rectangle represents a source or sink, and it is a net originator or consumer of data. A source sink is typically outside the main system of study.

Components

- ***Processes:***
  - Represented by circles or ovals.
  - Depict activities or transformations that occur within the system.
  - Processes manipulate data and produce information.
- ***Data Stores:***
  - Represented by rectangles.
  - Signify where data is stored within the system.
  - Examples include databases, files, or even physical storage locations.

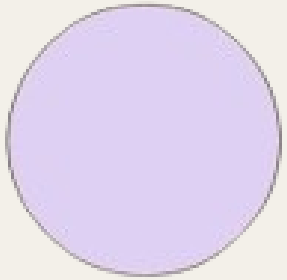
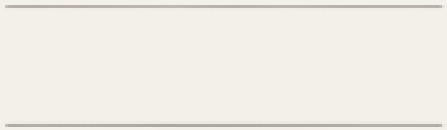
Cont.

- ***Data Flow:***

  - Represented by arrows.
  - Illustrate the movement of data between processes, data stores, and external entities.
  - Show the direction in which data is transferred.

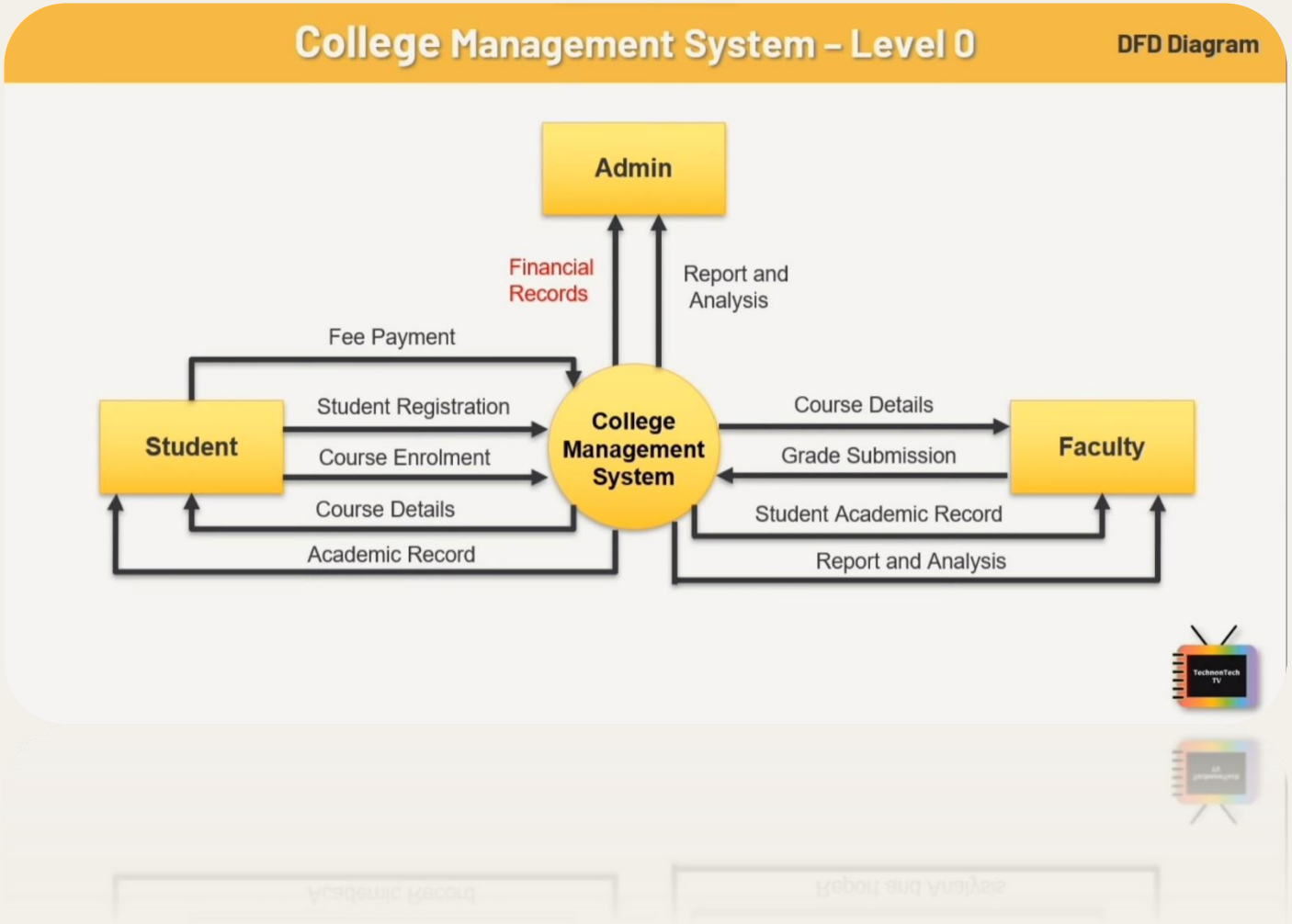
- ***External Entities:***

  - Represented by squares.
  - Depict sources or destinations of data that are outside the system.
  - External entities interact with the system by providing or receiving data.

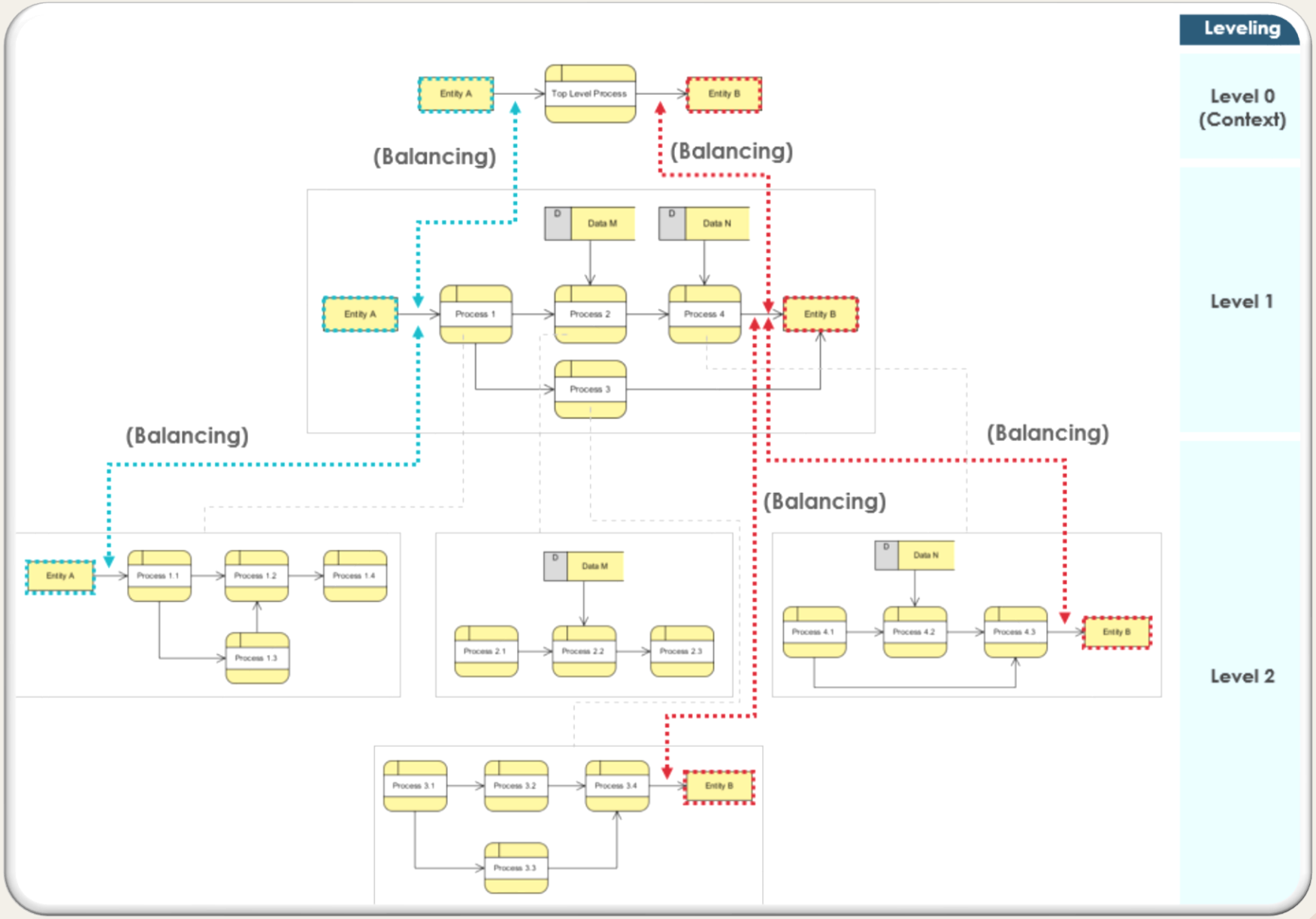
	Yourdon & Coad	Gane & Sarson
External Entity		
Process		
Data Store		
Data Flow		

How to create a data flow diagram?

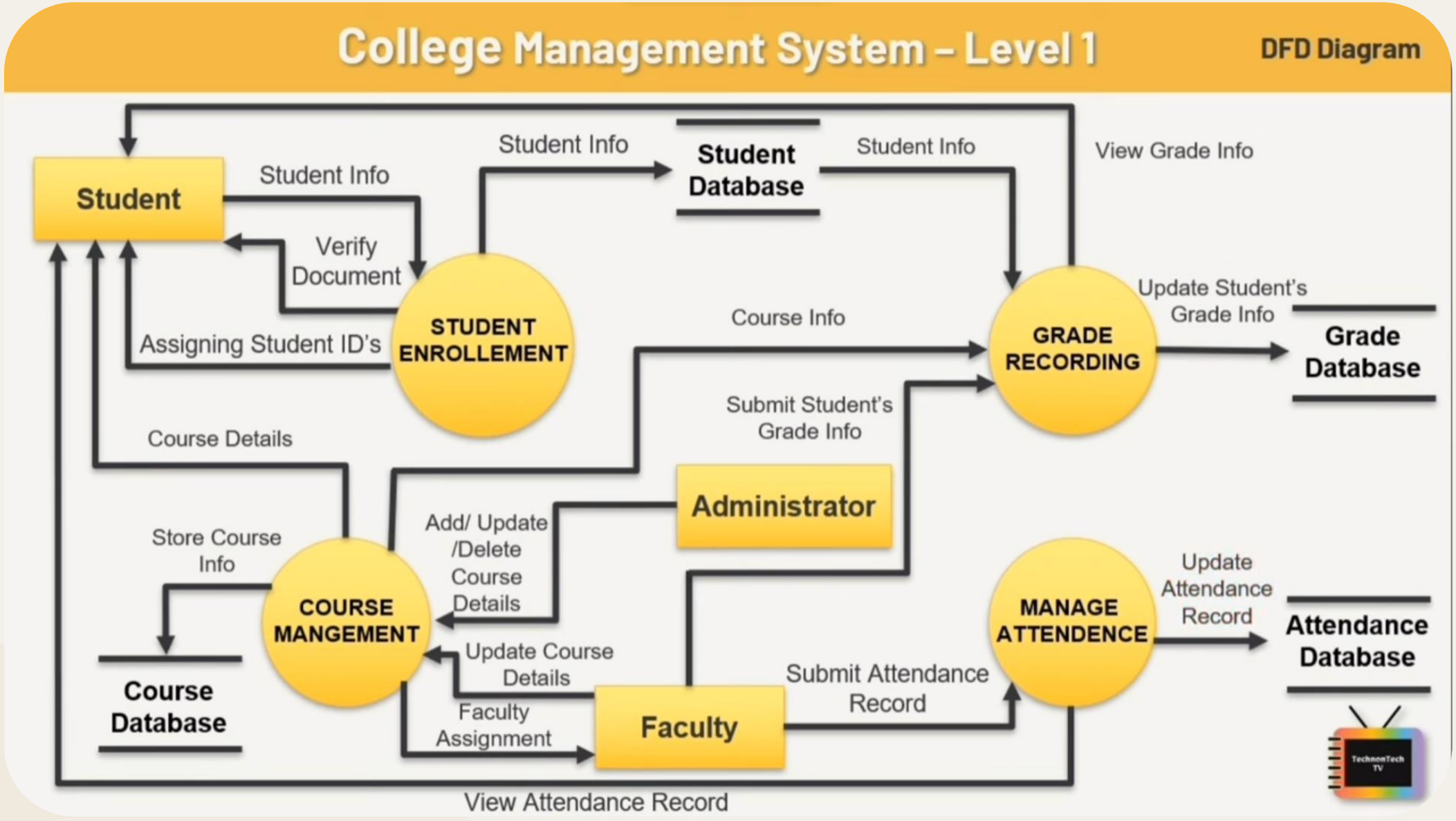
- **Select a system or process.**
- **Categorize related business activities.**
- **Context Diagram :**
  - ✓ Create a high-level overview with a single process node.
  - ✓ Connect external entities to show input and output flow.
  - ✓ Center node represents the general process.



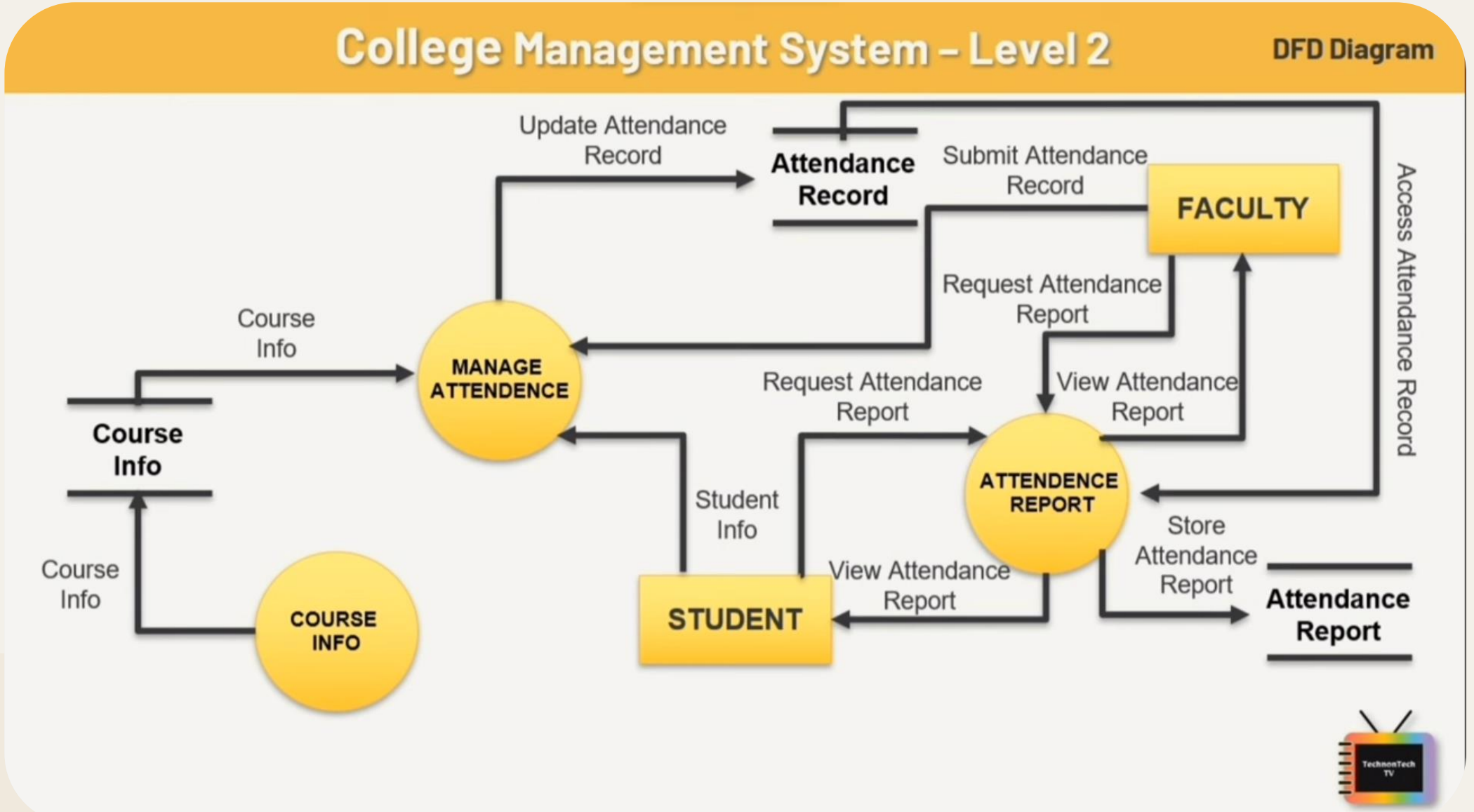
Cont. Process Decomposition



Cont. Level 1 DFD:

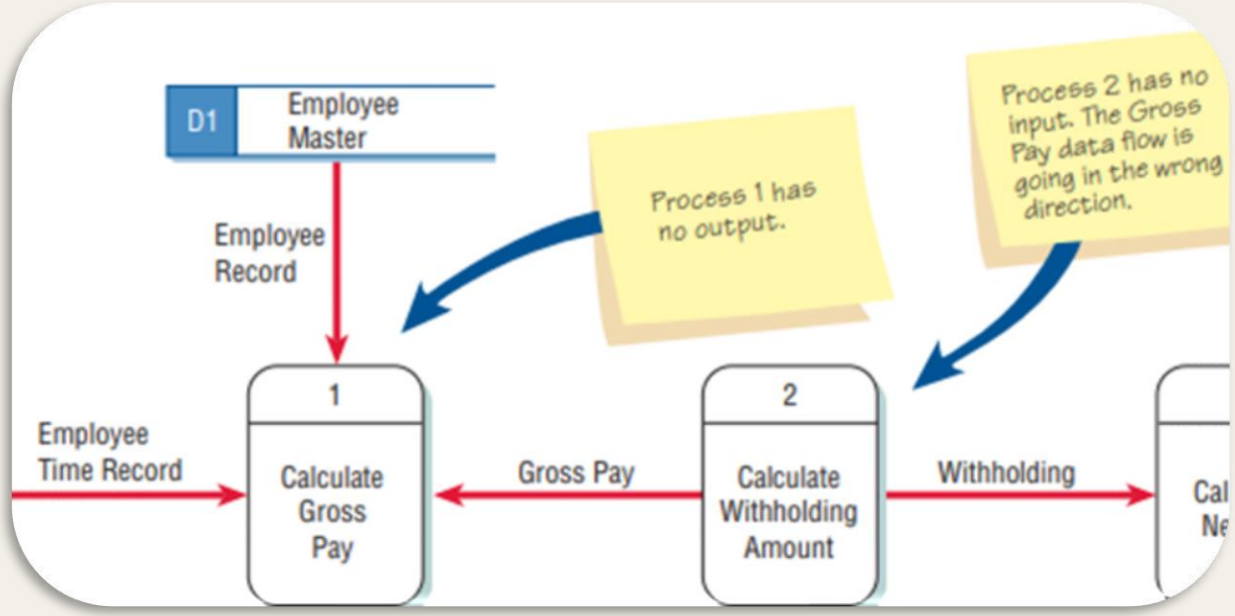


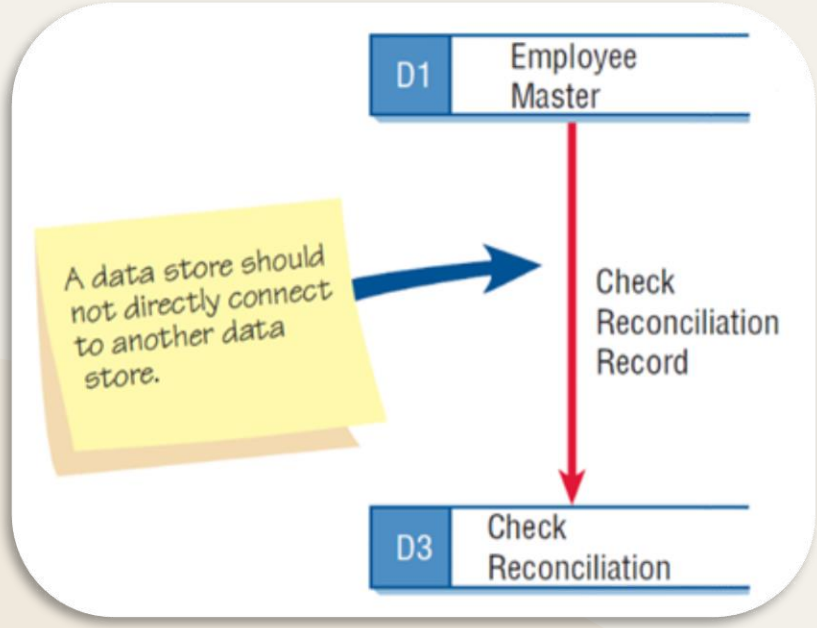
Cont. Level 2 DFD:



Cont. Common Mistakes :

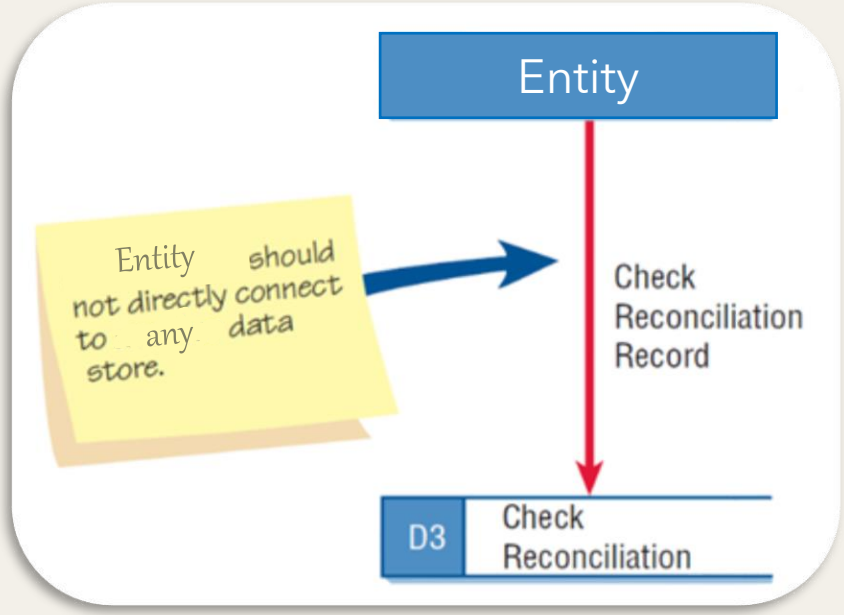
- Forgetting to include a data flow (input or output) or pointing an arrow in the wrong direction.





- Connecting data stores directly to each other.

Cont. Common Mistakes :



- Connecting data stores and external entities directly to each other.

• Incorrectly labeling processes or data flow.

✓ A process should indicate the system name or use the verb-adjective-noun format.

✓ Each data flow should be described with a noun.

• Including more than nine processes on a data flow diagram.

✓ Solution : group processes that work together into a subsystem and place them in a child diagram.

- Creating unbalanced decomposition (or explosion) in child diagrams.

✓ Each child diagram should have the same input and the same output data flows as the parent process.

pros

Clarity and Simplicity:

Clear and simple representation of complex systems.

Communication:

Effective communication tool for both technical and non-technical stakeholders.

Visualization of Processes:

Visual representation of data flow and processes in a system.

System Decomposition:

Supports breaking down complex systems into manageable subprocesses or modules.

Decision Support:

Assists in informed decision-making about system modifications.

Cost-Effective Analysis:

Identifies bottlenecks and inefficiencies early in the development cycle

Cons

Lack of detail:

DFDs provide a high-level overview and may not capture intricate system details.

Difficulty in managing changes:

Modifying DFDs to reflect system changes can be time-consuming and effort-intensive.

Lack of support for dynamic behavior:

DFDs do not capture the system's dynamic behavior or time-dependent interactions.

Incomplete representation of external entities:

DFDs may not provide detailed information about external entities and their interactions.

Difficulty in maintaining consistency:

Maintaining consistency across multiple DFDs can be challenging, especially in complex systems.

Thank you