# Data Flow Diagram
# State Transition Diagram

# DFD introduction

- ***DFD Introduction***

  - Data Flow Diagrams (DFDs) are **a graphical/representation** of systems and systems components.

  - They show the **functional relationships** of the values computed by a system, including **input values, output values, and internal data stores**.

  - It's a graph showing the flow of data values from their sources in objects through processes/functions that transform them to their destinations in other objects.

# Data Flow Diagramming Rules

## Data Flow

- **Unidirectional flows** – Data moves in one direction only
- **Forking allowed** – Same data can go to multiple destinations
- **Joining allowed** *only if identical* – Data must be exactly the same
- **No recursion** – A process cannot send data to itself
- **Use noun phrases** – For naming data flows, stores, and sources/sinks (like **Customer Order) – (Send Order)**

# Data Flow Diagramming Rules

**Data Stores & Sources/Sinks – Key Rules**

- **No direct flow between two data stores**
  - Must use a process in between
- **No direct flow between a data store and a source/sink**
  - A process must handle the data
- **No flow between two sources/sinks**
  - It's either not useful, or a process is missing
- **Always show a process** to read from or write to a data store
- **Processes represent system work – without them, nothing is happening**

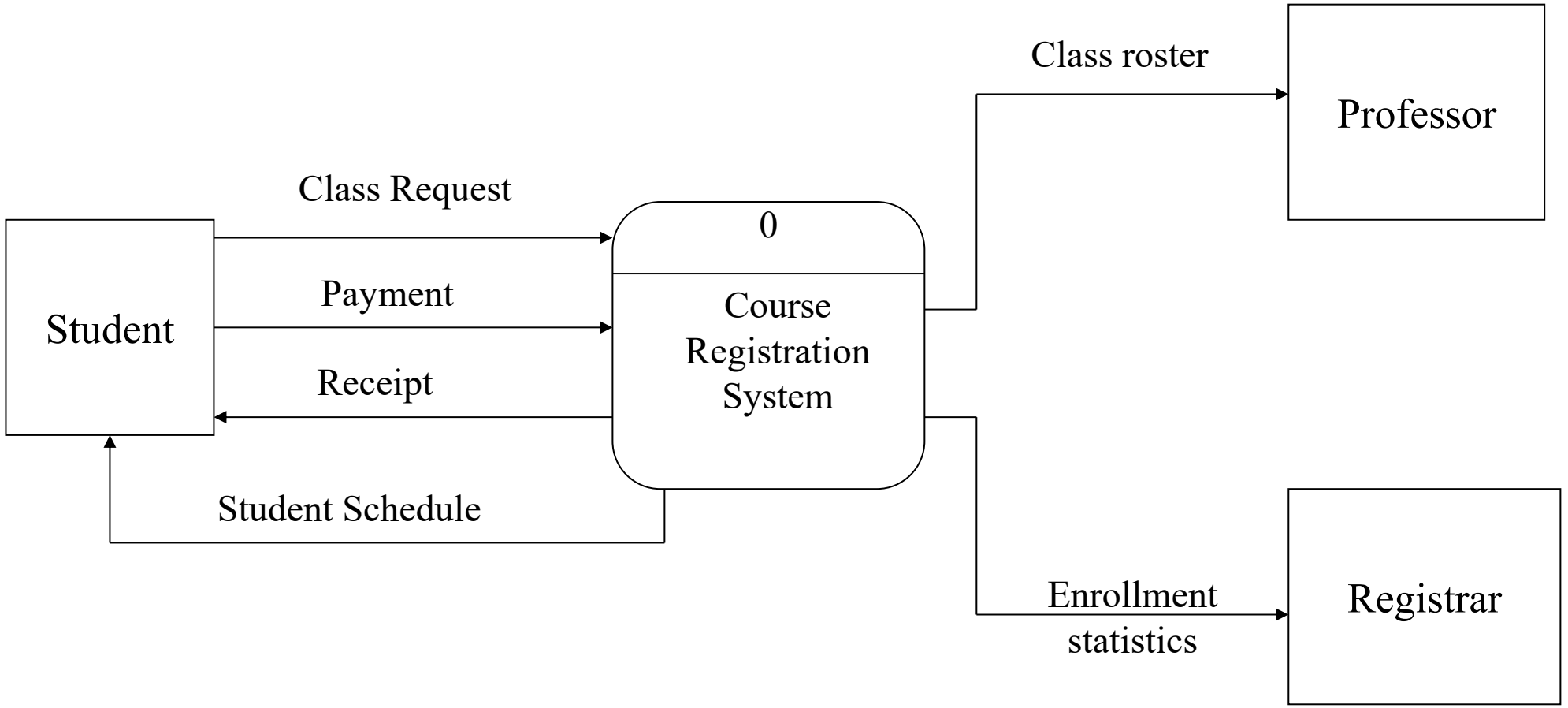
# Data Flow Diagramming Rules

- **Processes**
  - a process must have at least one input
  - a process must have at least one output
  - Context level process → Just the system name (no verb needed)
  - Lower-level process → Should describe the action (verb phrase: “Update Customer Record”)
  - Physical DFD → **Can use longer phrases for more detail**.
  - “Send email to finance department”
  - “Print student registration confirmation form”

# Practice

- Develop a context diagram for a course registration system

# Course Registration: Context level Diagram

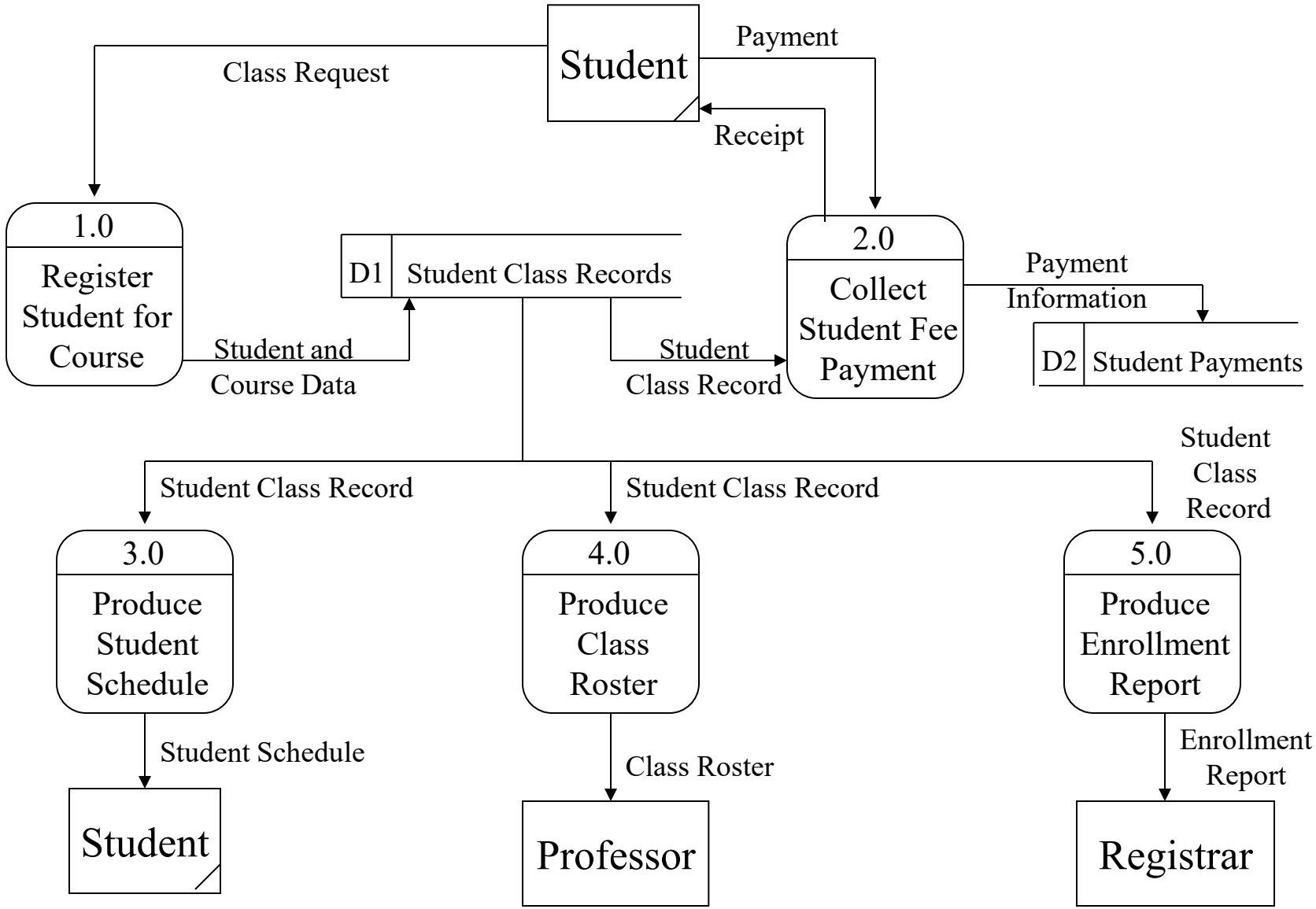


# Level 0 Diagram

**Exploding a Process (DFD Decomposition)**

- **Process is “exploded”**
  - Break one big process into smaller sub-processes
- **Sources, sinks, and data flows are repeated**
  - Keep external entities and flows from the context diagram
- **Process is broken down into sub-processes**
  - Label sub-processes like 1.1, 1.2, 1.3, etc.
- **Lower-level data flows and data stores are added**
  - Show internal data movement and storage not shown before
- **Gives more detail**
  - Helps understand how the system really works inside

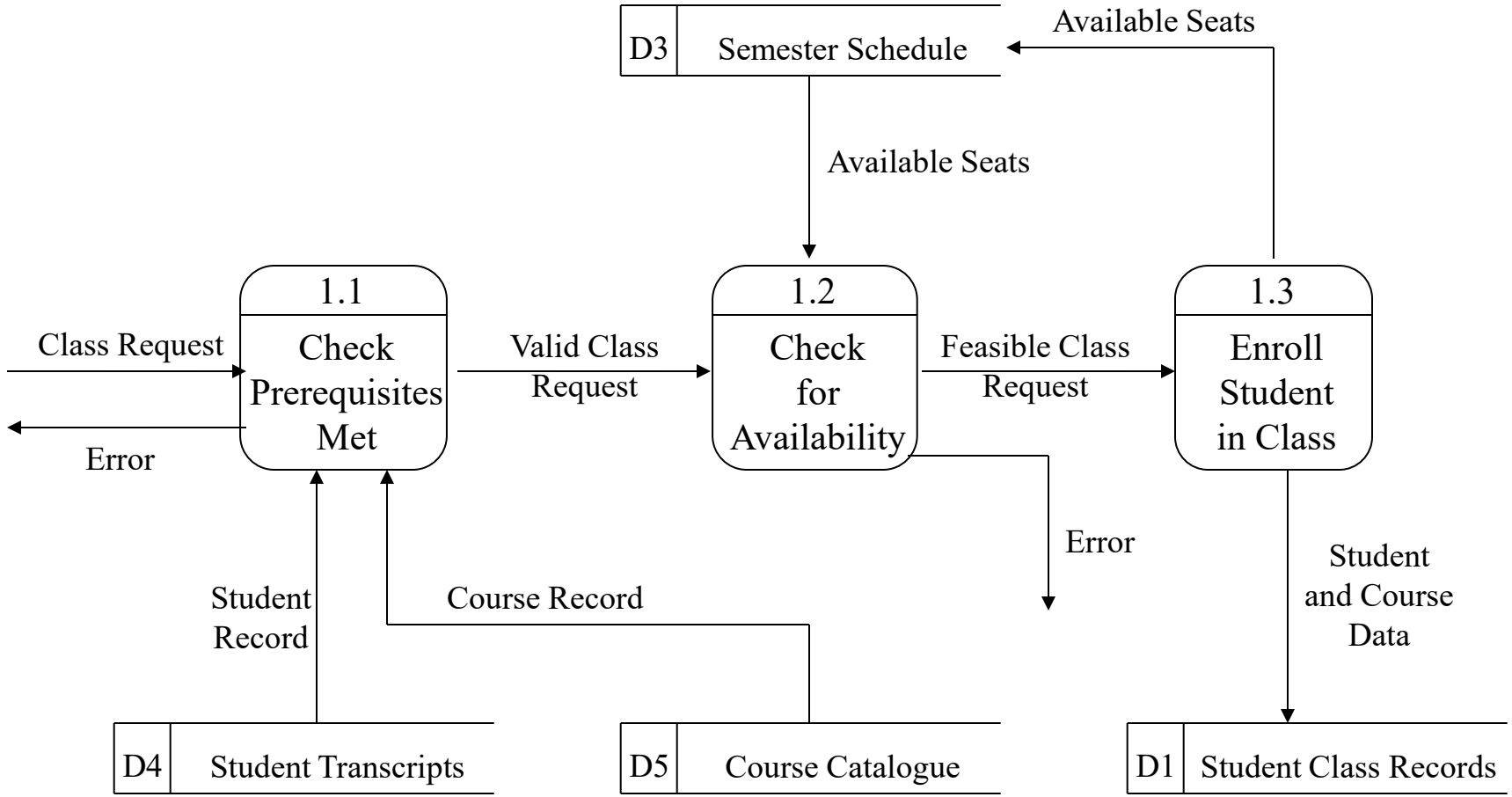
# Course Registration: Current Logical Level 0 Diagram



# Child Diagrams

- “Explode” one process in level 0 diagram
- Break down into lower-level processes, using numbering scheme
- Must include all data flow into and out of “parent” process in level 0 diagram
- Don’t include sources and sinks
- May add lower-level data flows and data stores

# Course Registration: Current Logical Child Diagram



# Class Activity: "Draw a DFD for a Student Attendance System"

**Scenario:**

- Teachers record student attendance daily.
- The system stores attendance data and generates reports for each student.
- Students and parents can view their attendance records online.
- Admin staff can generate monthly or yearly attendance summaries.

# Step-by-Step Breakdown:

## Part 1: Identify DFD Elements

- **External Entities:**
  - Teacher
  - Student
  - Parent
  - Admin Staff

# Step-by-Step Breakdown:

- **Processes:**
  - Record Attendance
  - View Attendance
  - Generate Reports
- **Data Stores:**
  - Student Database
  - Attendance Records

# Step-by-Step Breakdown:

- **Data Flows:**
  - Attendance Sheet
  - Attendance Report
  - Student Info
  - Summary Request
- **Part 2: Level 0 DFD**
- Single process: “Attendance Management System”
- Connect all external entities with relevant data flows

# Step-by-Step Breakdown:

**Part 3: Level 1 DFD**

- Expand the main process into sub-processes:
- Record Daily Attendance
- Store Attendance Data
- View Student Attendance
- Generate Admin Reports

**Include:**

- How each process interacts with data stores
- How external users interact with specific processes

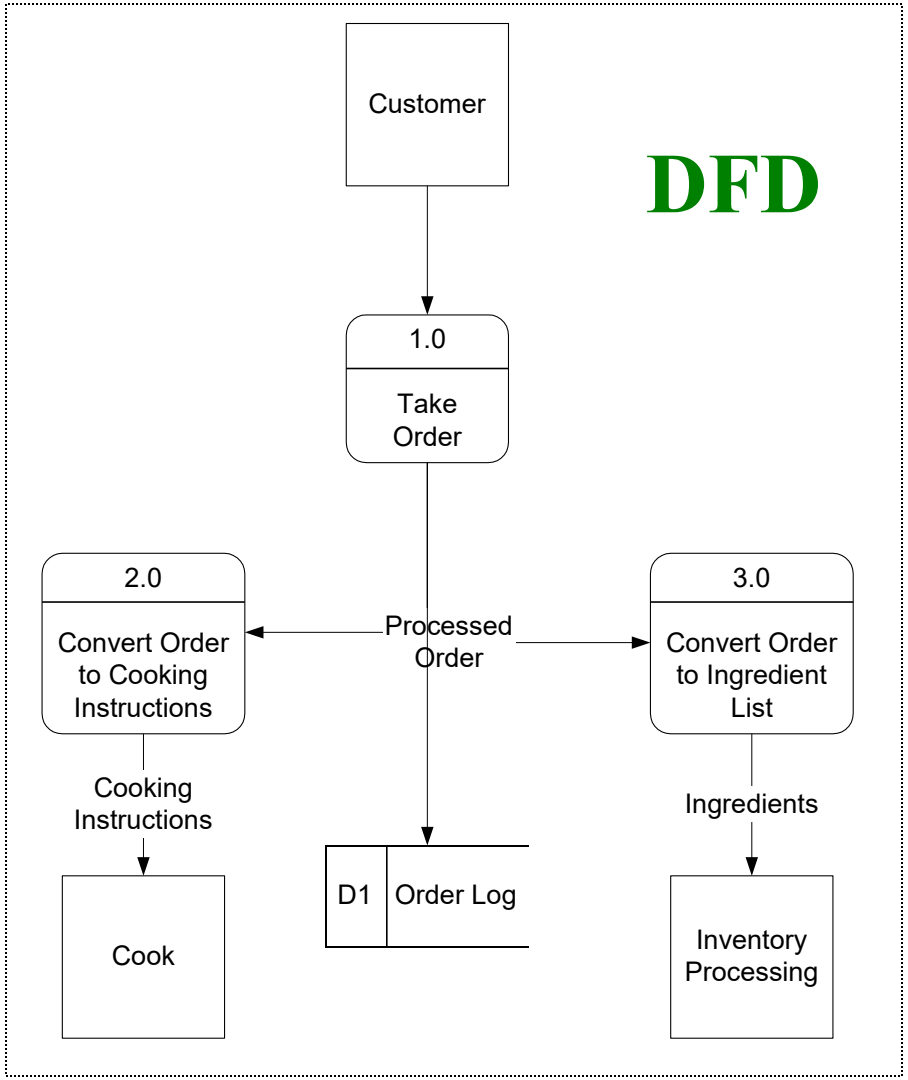
# DFDs and ERDs

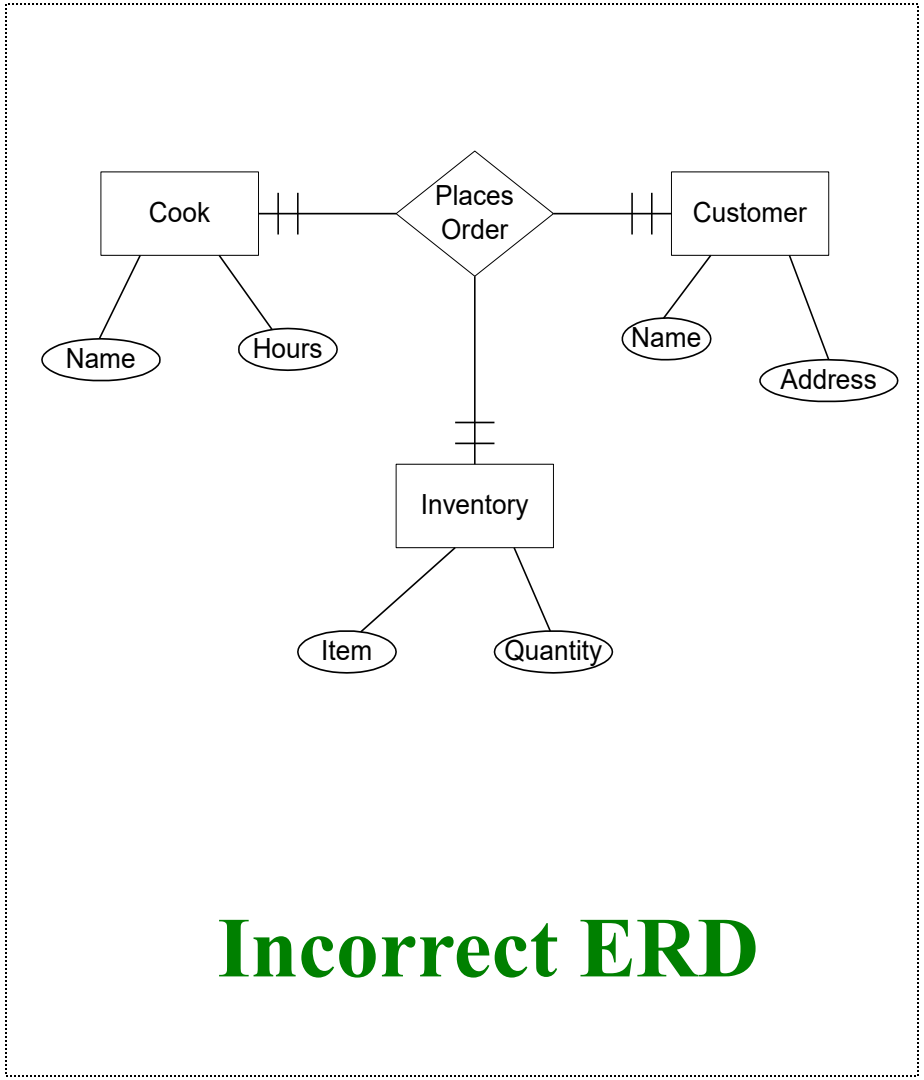
- DFDs and ERDs are both used to model systems, but they show two very different perspectives on the system
- A DFD shows what the system ***does*** as well as the ***data*** that the system manipulates
- An ERD shows **only** the ***data*** that the system manipulates.

# DFDs and ERDs (cont.)

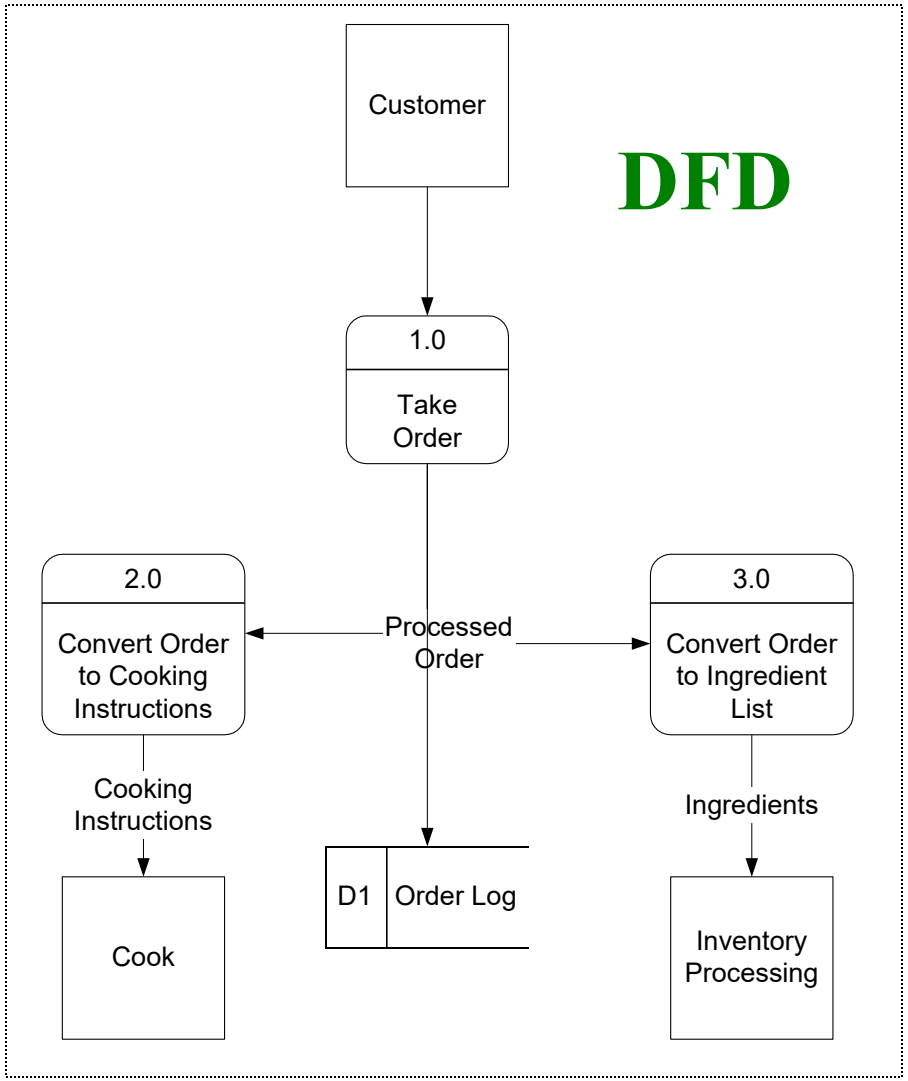
- Entities on an ERD often (but not always) correspond to data stores on a DFD
- Attributes on an ERD usually correspond to data elements (listed in the data dictionary) that make up the data store and data flows on a DFD
- Relationships on an ERD **do not** correspond to processes on a DFD.
- Sources and sinks on a DFD usually **do not** show up as entities on an ERD

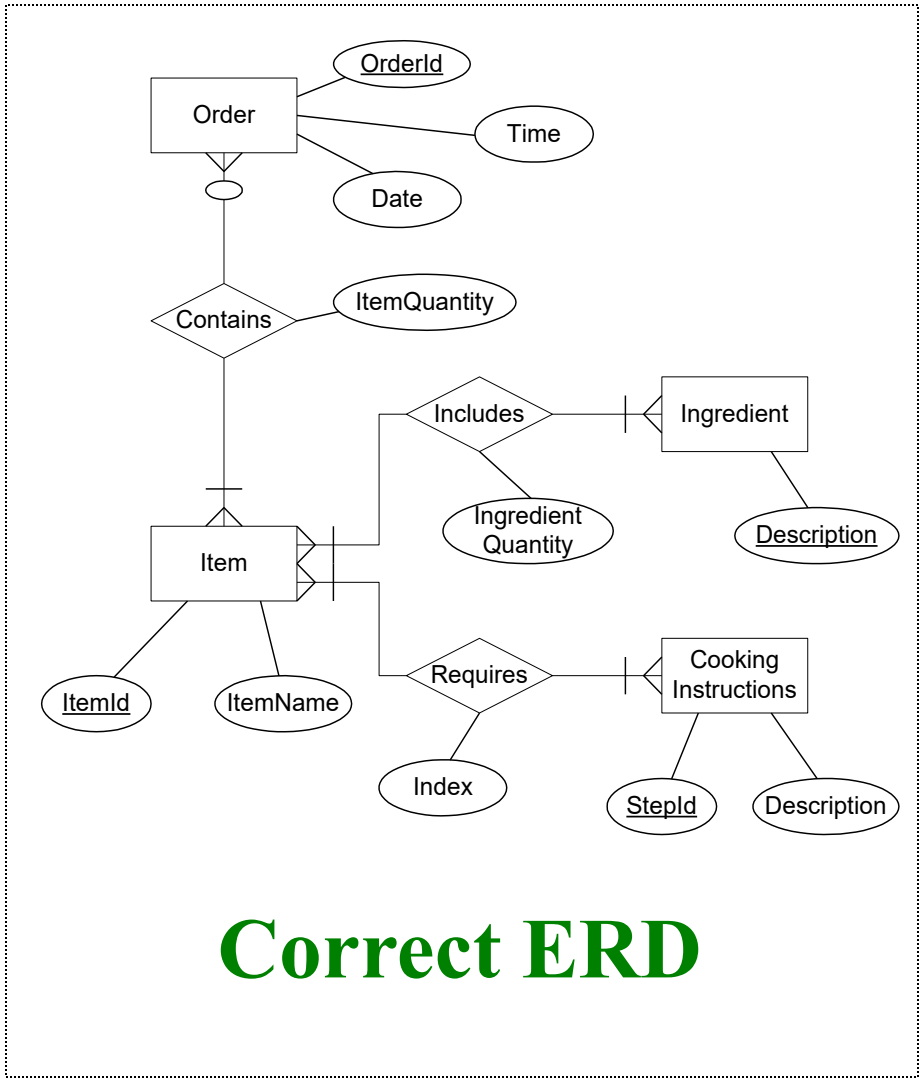
# Example DFD and ERD



# Example DFD and ERD



# State Transition Diagram

# State Transition Diagram

- ***What is State Transition Diagram?***
  - State transition diagrams are used to show how the system respond to the user.
  - It indicates how the system moves from one state to another.
  - It also show what actions are taken as a result of some event

j.khan@kardan.edu.af

# State Transition Diagram

- A **State Transition Diagram (STD)** is a **behavioral diagram** used in software engineering and system design to **show how an object or system transitions from one state to another** in response to events.
- A State Transition Diagram shows:
- **States**: The different situations or conditions the system can be in.
- **Transitions**: How the system moves from one state to another based on events or actions.
- **Events**: Triggers that cause the change in state.

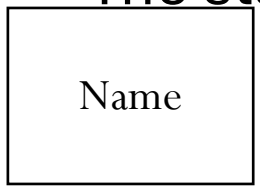
j.khan@kardan.edu.af

# State Transition Diagram Elements

- ***Elements of State Transition Diagram***
  - **State**
    - Is a set of functions and its needed data.
    - At any time a particular STD can only be in one state.
  - **Transition**
    - A transition is a change in state caused by an event.
    - The transition is a directed line between the old state and the new state.
    - The line is labeled with the name of the event.
  - **Start**
    - The initial state.
  - **Stop**
    - The Final state

# State Transition Diagram [State]

- ***What is State?***
  - A state is an observable mode of behavior(Function + Data).
  - The state of the system determines the response to different events.



# State Transition Diagram[ Transition]

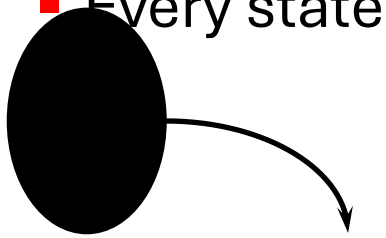
- ***What is Transition?***
  - The transition is an activity causing the system to move from one state to

event/action

j.khan@kardan.edu.af

# State Transition Diagram [ Start]

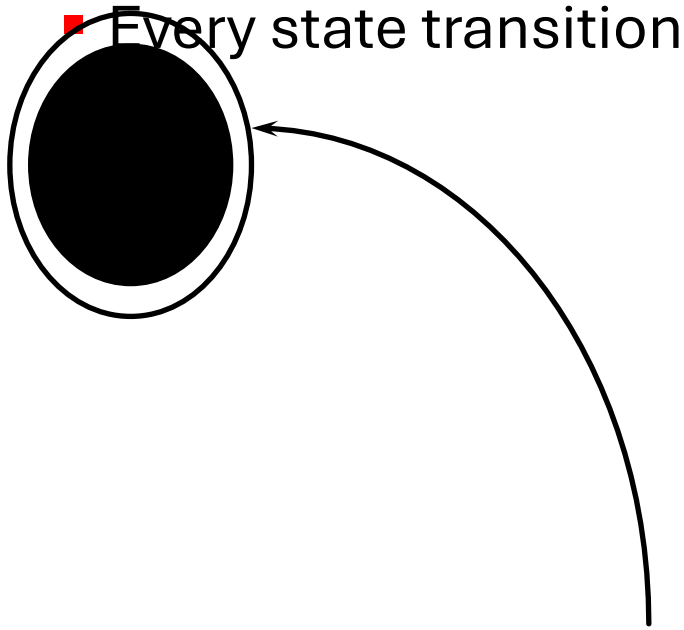
- ***What is Start state?***
  - Every state transition diagram must have a start symbol



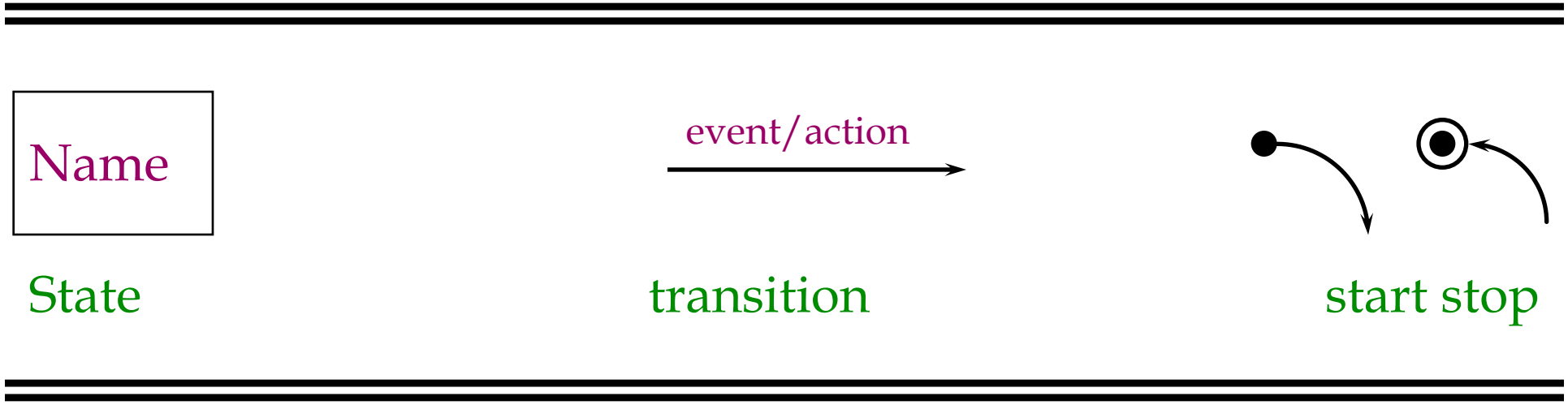
j.khan@kardan.edu.af

# State Transition Diagram [ stop]

- ***What is stop state?***
  - Every state transition diagram may have one or more stop symbols.

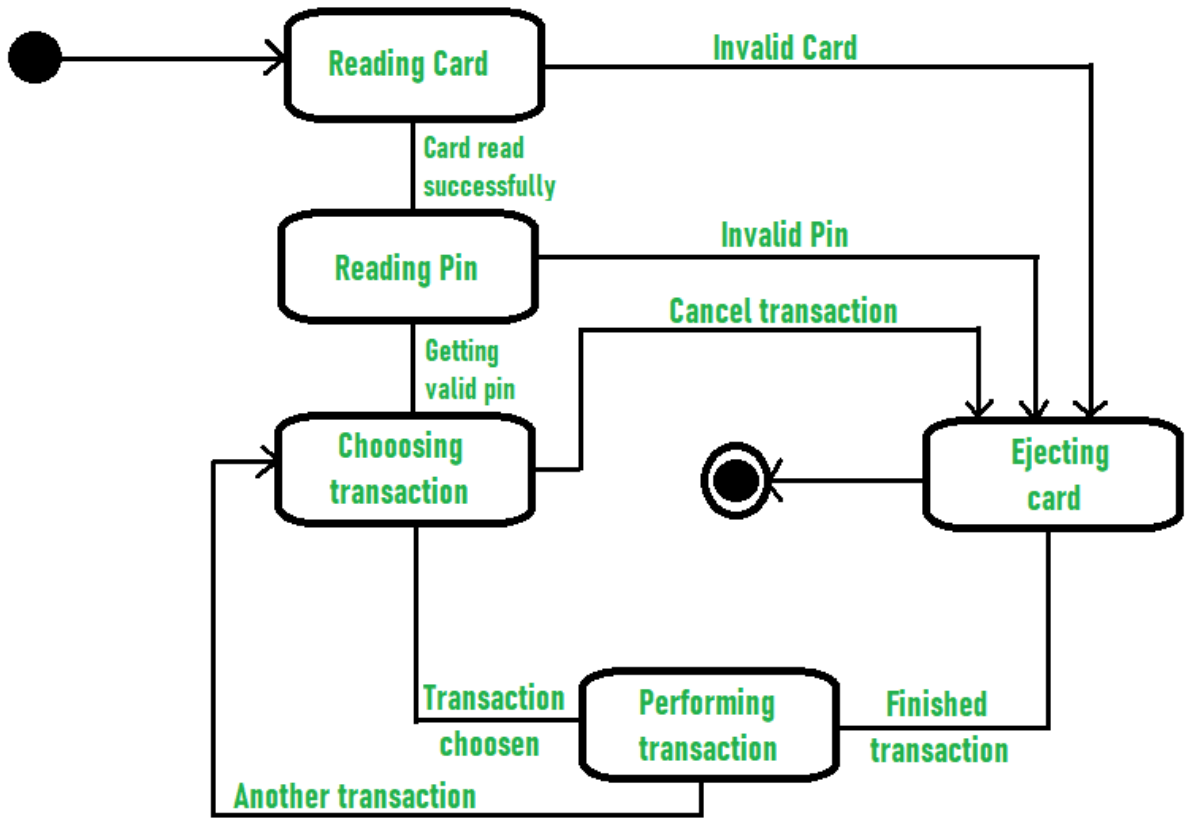


# State Transition Diagram



# State Transition Diagram Example

- ***Example?***
  - ATM state machine example

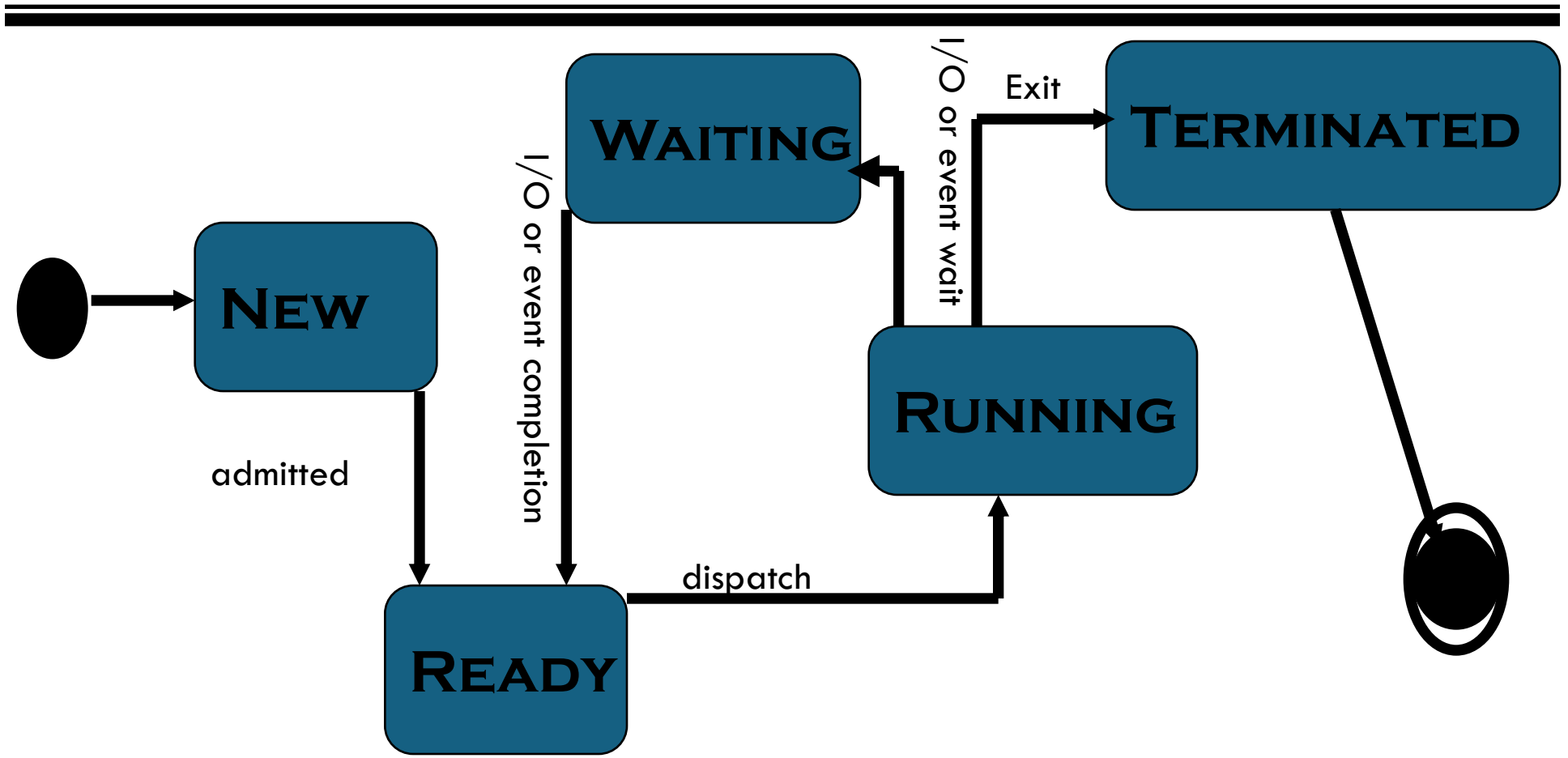



**State Transition Diagram for ATM System**

# [Process execution][windows] Example

- ***Possible states are***
  - New
  - Ready
  - Running
  - Waiting
  - Terminated

# Process execution[windows]



Thank You