SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with ‘A’ Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

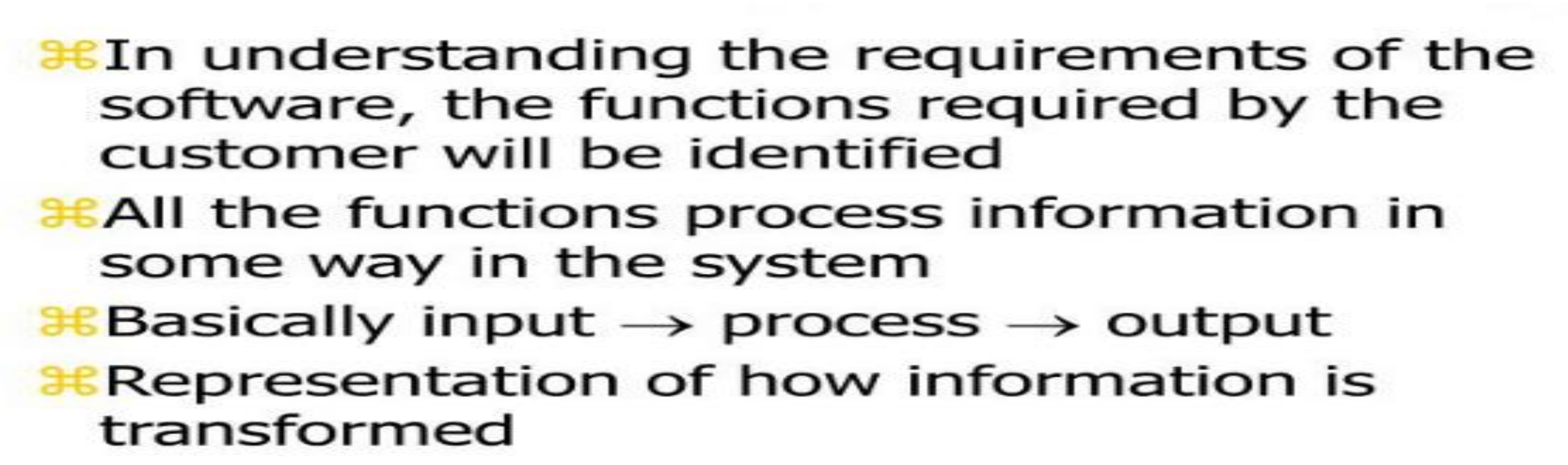
Department of Information Technology

Course Name –Software Engineering

II Year / III Semester

REQUIREMENT ANALYSIS

Functional Model



⌘In understanding the requirements of the software, the functions required by the customer will be identified

⌘All the functions process information in some way in the system

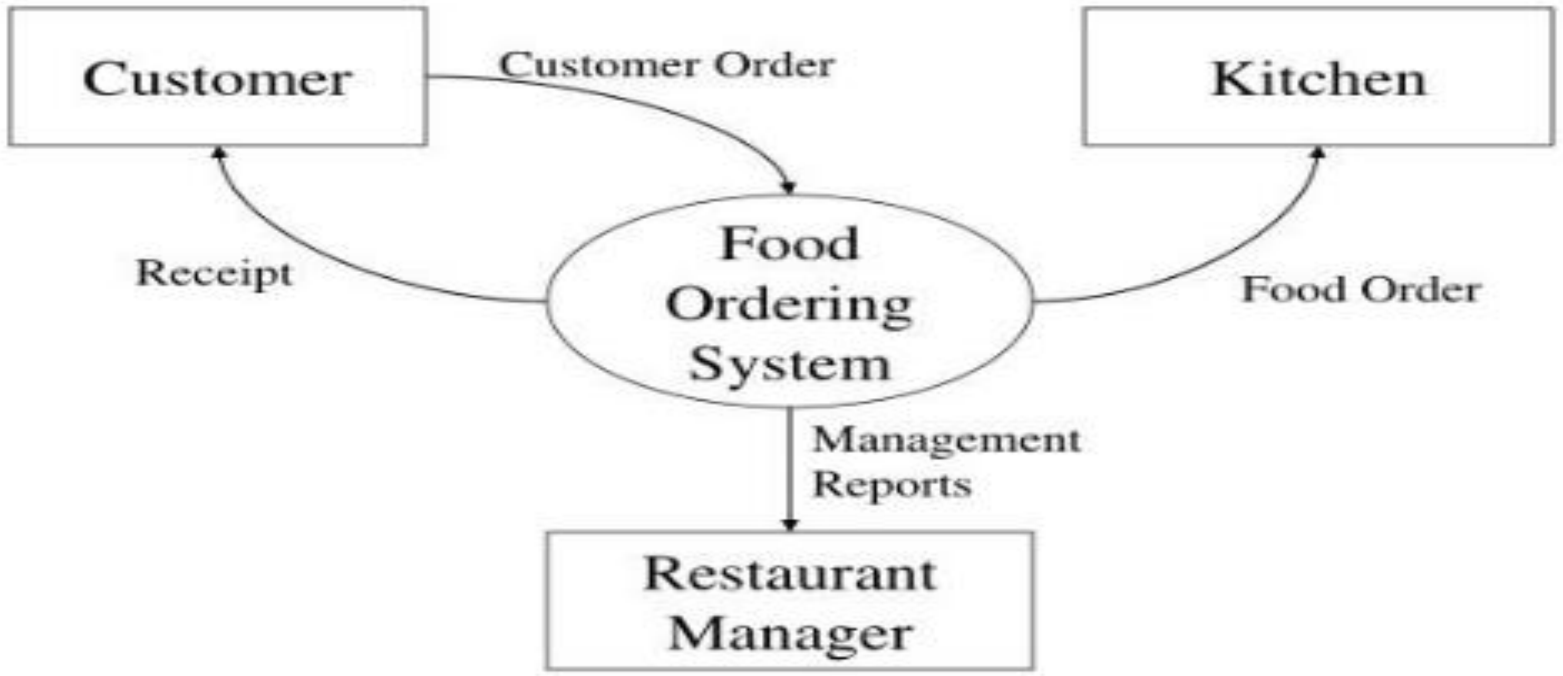
⌘Basically input → process → output

⌘Representation of how information is transformed

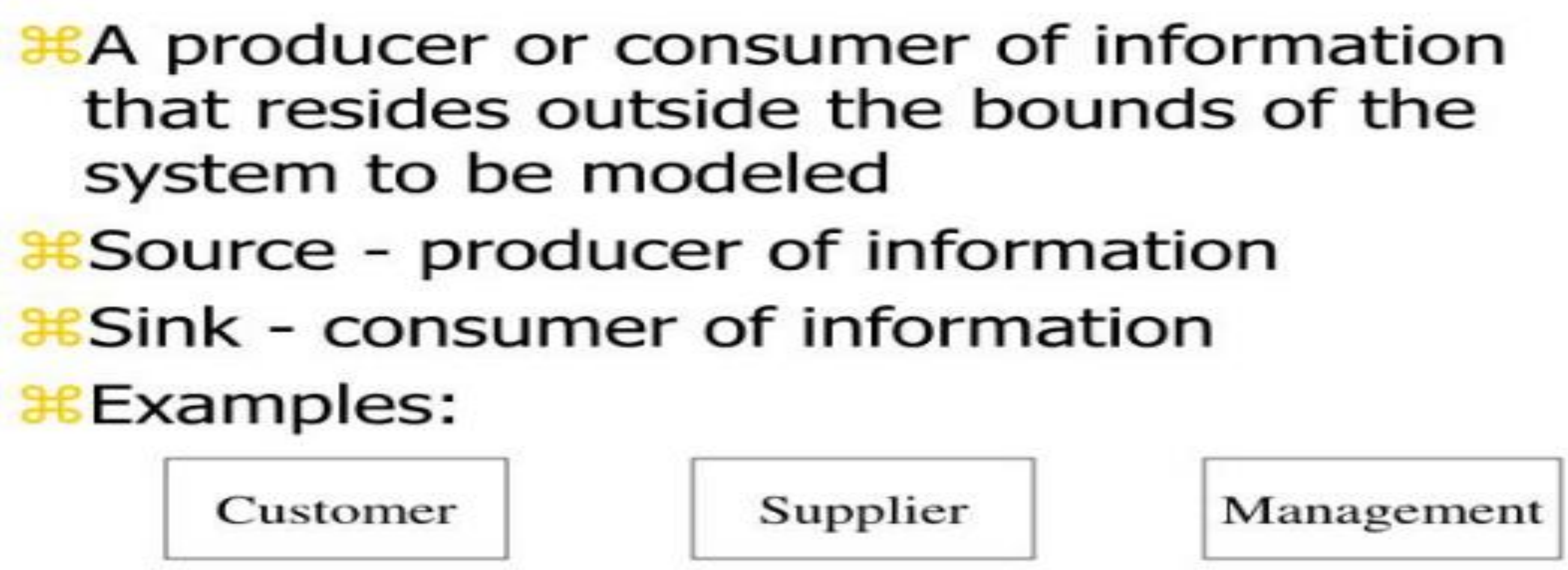
Data Flow Diagram (DFD)

- Graphical representation of functional modeling
- In analysis, provide representation of information flow in existing and required system
- In design, the DFDs can be decomposed into lower level processes (sub-systems) for implementation

Context Diagram



External Entity

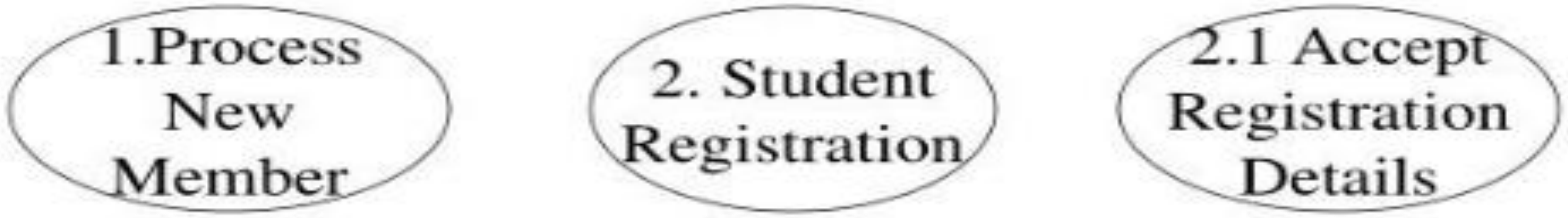


- A producer or consumer of information that resides outside the bounds of the system to be modeled
- Source - producer of information
- Sink - consumer of information
- Examples:

Customer
Supplier
Management

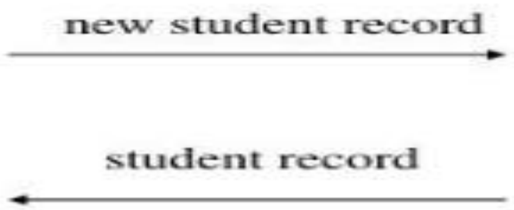
Process

- A transformer of information (a function) that resides within the bounds of the system to be modeled
- Examples:

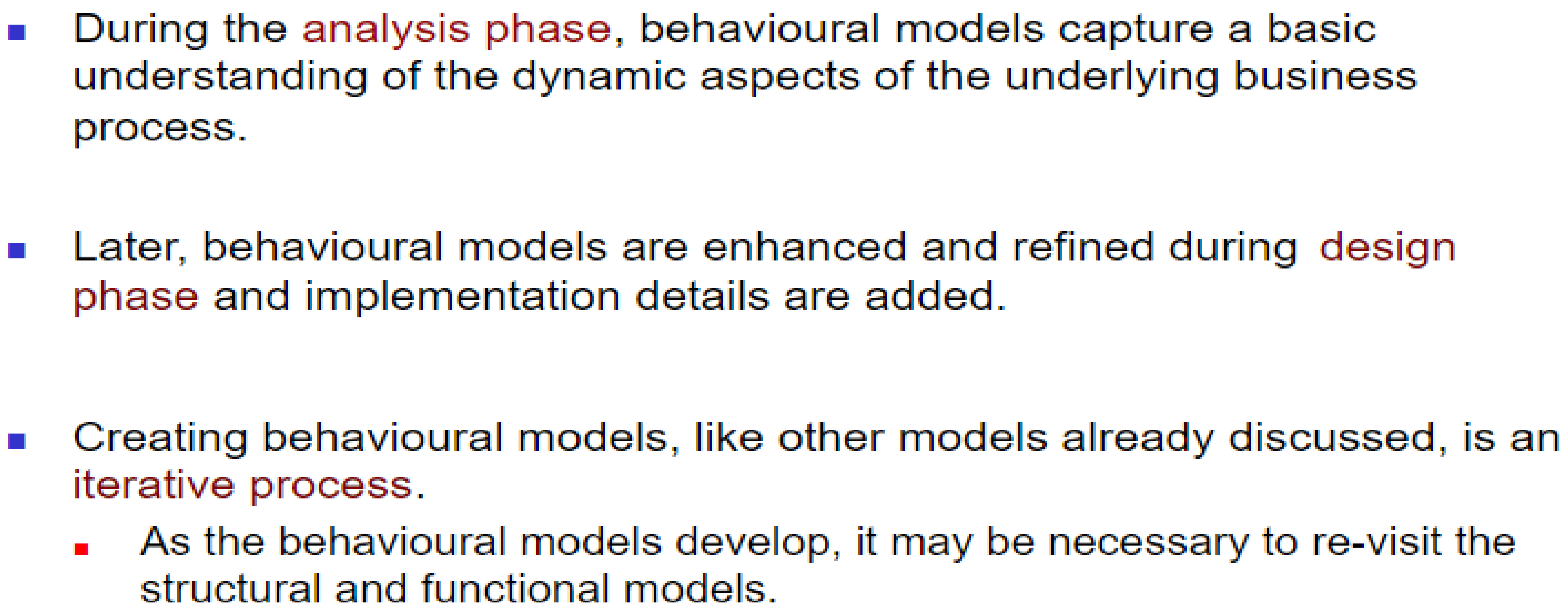


Data Flow

- Data object that flows in the system; the arrowhead indicates the direction of data flow

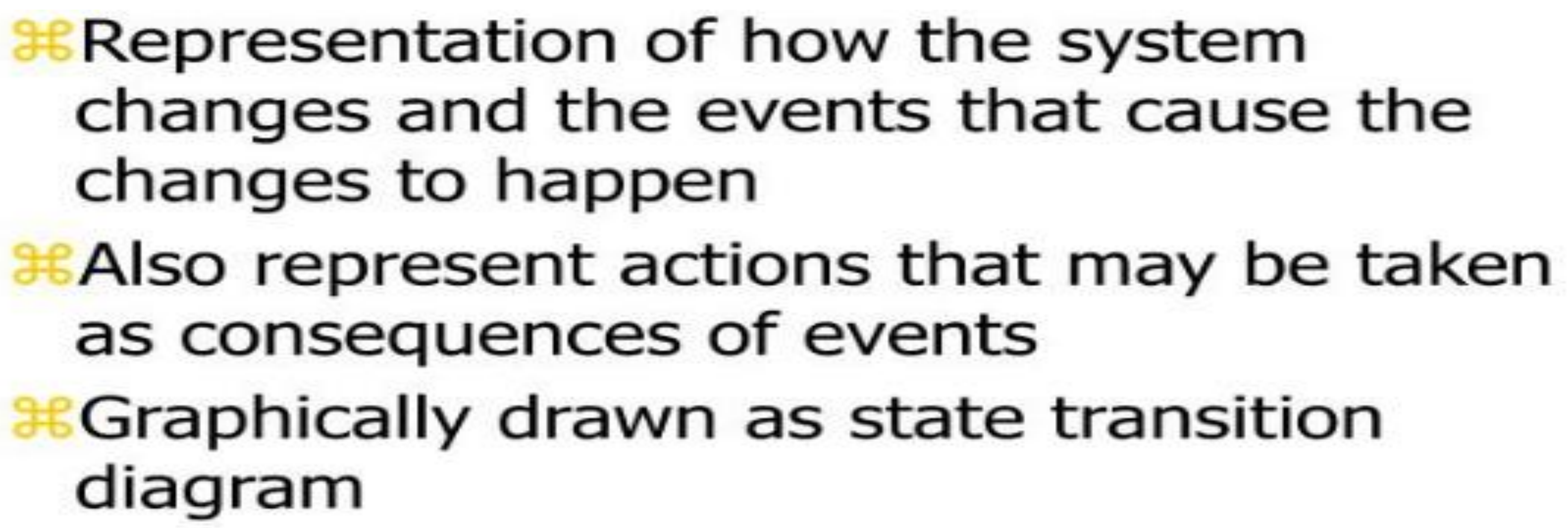


Behavioural Models



- During the analysis phase, behavioural models capture a basic understanding of the dynamic aspects of the underlying business process.
- Later, behavioural models are enhanced and refined during design phase and implementation details are added.
- Creating behavioural models, like other models already discussed, is an iterative process.
  - As the behavioural models develop, it may be necessary to re-visit the structural and functional models.

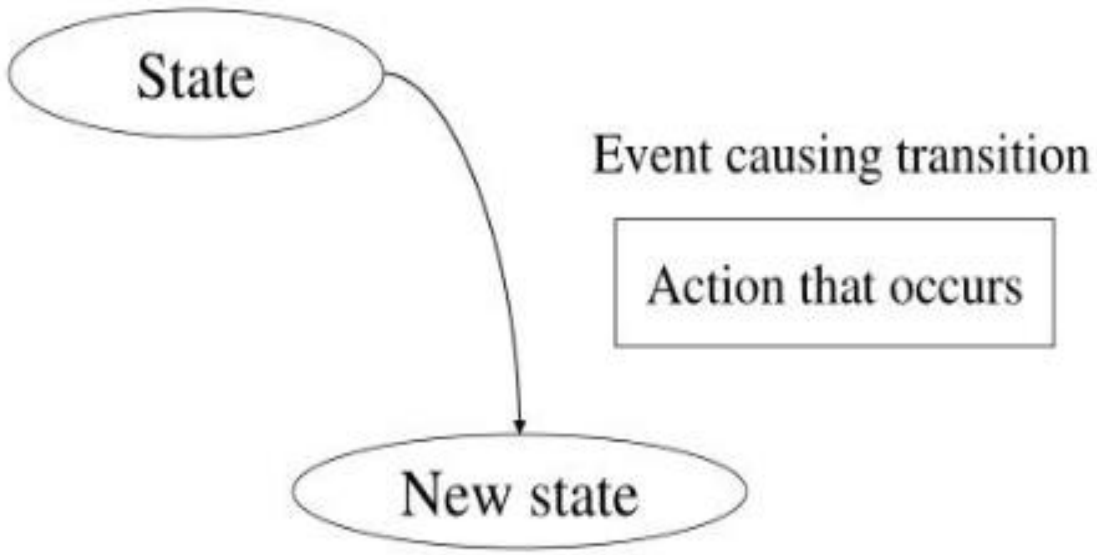
Functional and Behavioural Models

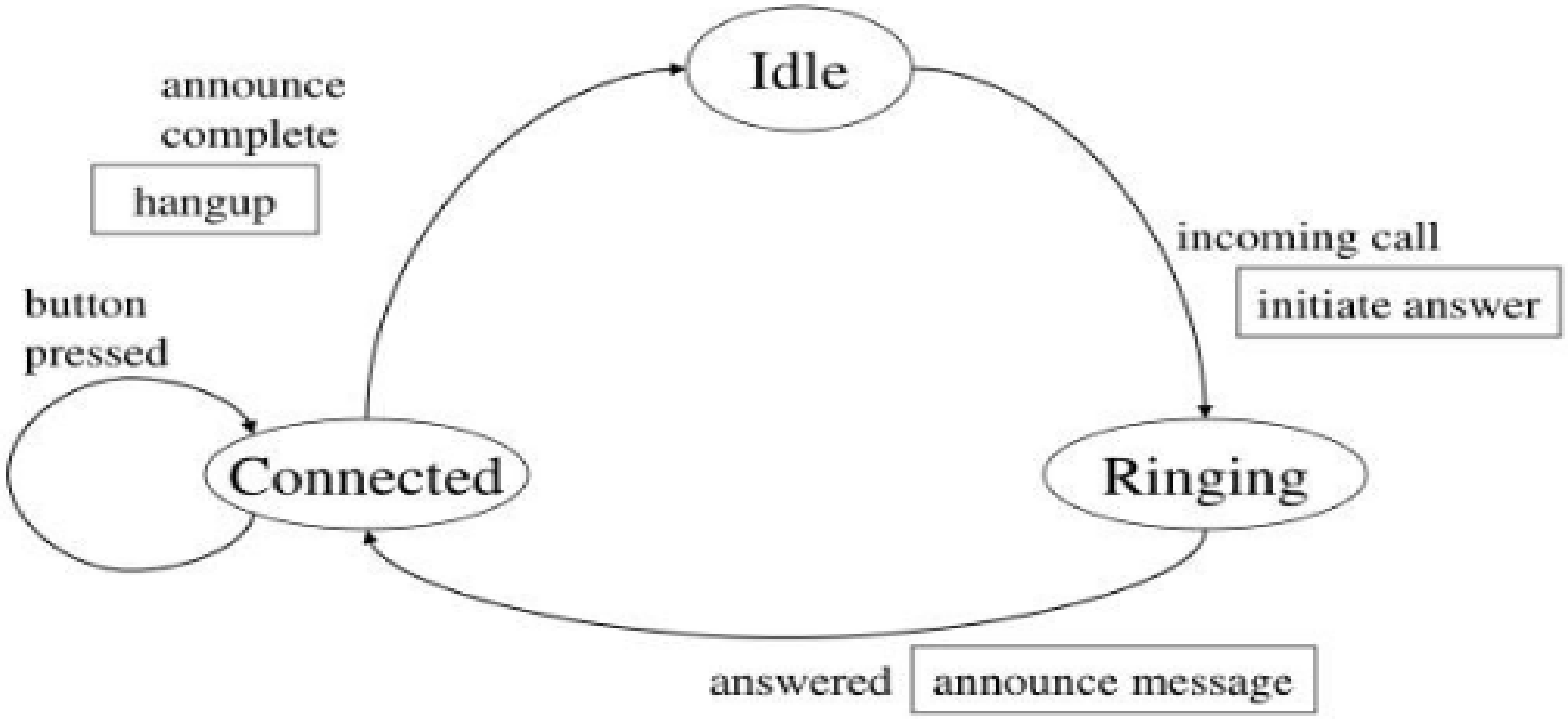


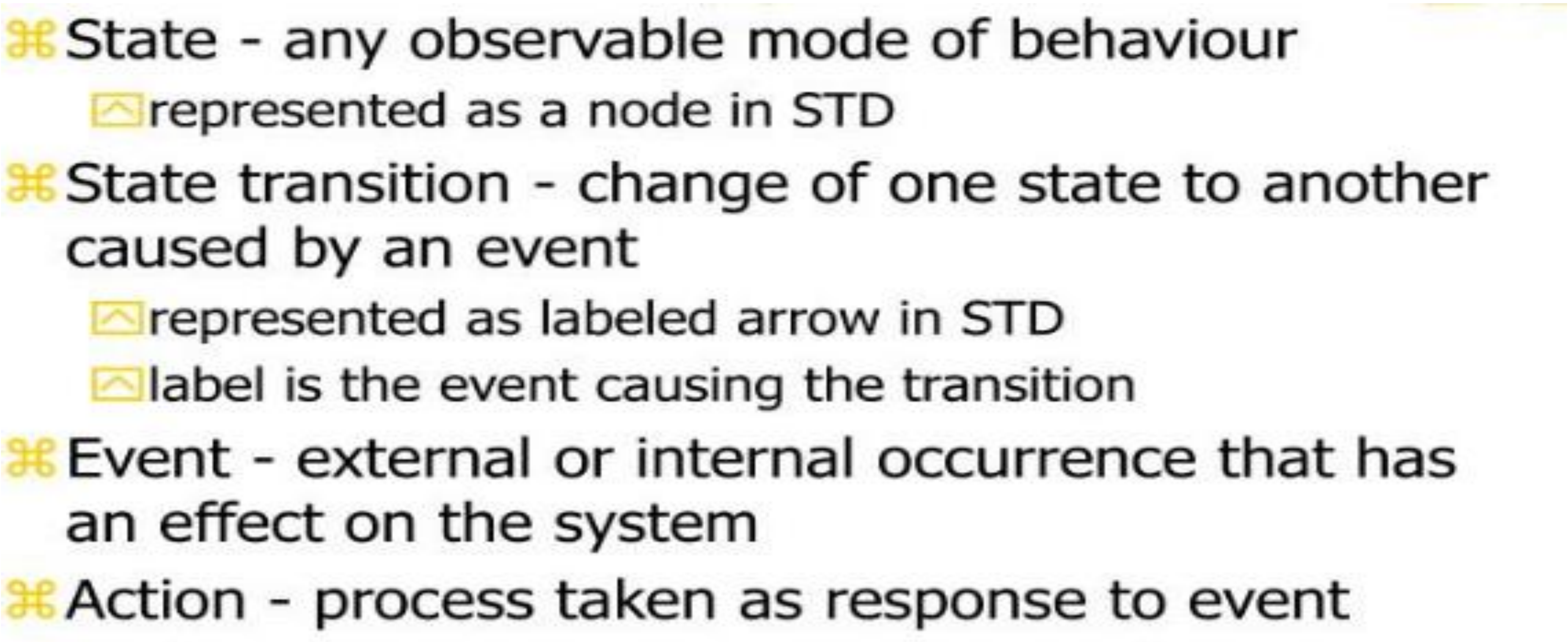
⌘Representation of how the system changes and the events that cause the changes to happen

⌘Also represent actions that may be taken as consequences of events

⌘Graphically drawn as state transition diagram




Idle
announce
complete
hangup
incoming call
initiate answer
button
pressed
Connected
Ringing
answered
announce message



- State - any observable mode of behaviour
  - represented as a node in STD
- State transition - change of one state to another caused by an event
  - represented as labeled arrow in STD
  - label is the event causing the transition
- Event - external or internal occurrence that has an effect on the system
- Action - process taken as response to event

- State – the state of an object is defined by the value of its attributes and its relationships with other objects at a particular point in time.

- Example: possible states of Patient class: checked-in, admitted, under-observation, released

- Note: not all attributes or attribute changes will make a difference (e.g. a change in the patient's address should not determine how the patient is treated.)

- Event – something that takes place at a certain point in time and changes a value(s) that describes an object, which in turn changes the object's state.

- An event can be:
    - a condition becoming true
    - receiving a message from another object to execute an operation (a method call)
    - the passage of a designated period of time

- The state of an object determines exactly what the response to the event will be.

- Example: patient (current state 'under observation') is diagnosed as healthy, status changes to 'released'.

- Transition — the movement of an object from one state to another.

- A guard condition is a True/False expression. Transition only takes place if expression is True.
  - Expression includes attributes values

- Example: a patient transitions from the state ‘under-observation’ to ‘released’ when [Diagnosis = Healthy]