State transition system

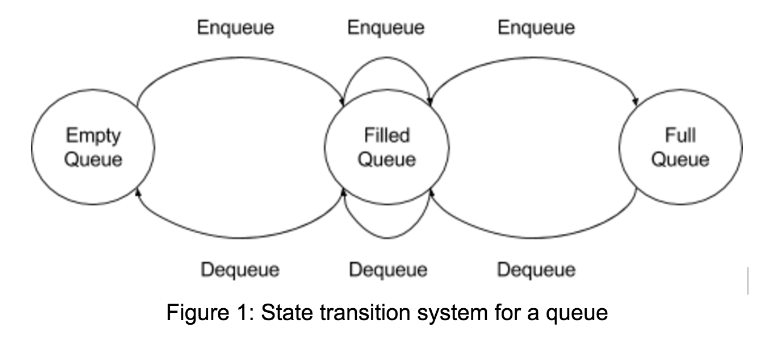
* a **concept** used to describe the **behavior** of a system
  + all potential behaviors, even if low probability
  + still describe how that state can be arrived at if it is an expected behavior of the system

Terminologies

1. State
   * The **information** that a system remembers defines its state
   * For example, a queue system can be in a number of different "states": an empty state, a queued state, or a full state.
2. Transition
   * Used to **describe the change** of a system from one state to another
   * A single system state can have multiple transitions; majority of systems today will have multiple transitions branching from one system state. This makes behaviors non-deterministic, since we cannot predict what the next state of the system will be.
3. Behavior
   * **describes what the system will do** when exposed to a condition, which can vary from timed system events to user input.

State transitions can be:

1. Unlabeled
   * defined as a set of **state (S) and transition (→) pairs** that are used to describe the system's behavior
   * If p and q are two different states in S, then the transition between them is depicted by (p → q)
   * Can be used where the transition between all states is the same
     + For example, your system could respond to a single button press that simply transitions the system sequentially from one state to the next.
2. Labelled
   * simple includes a set of labels, ~, with addition to the state-transition pair
   * Given the same states p and q in S, then the transition between the two states is shown as (p ⭇ q)



3 states:

1. Empty
2. Filled
3. Full

Where can state transition system help us?

* + Help us understand **how parallel processing, multithreading, or distributed computing** can affect the overall state of our system
  + Do **we need to wait for another process to finish** its work before continuing?
  + At which junction of our system will we be **bottlenecked**?
  + Help us identify deadlocks
    - process is waiting indefinitely for another to release a shared resource or complete its work
    - help you easily identify deadlocks since they occur if there is a condition that prevents a transition out of particular state