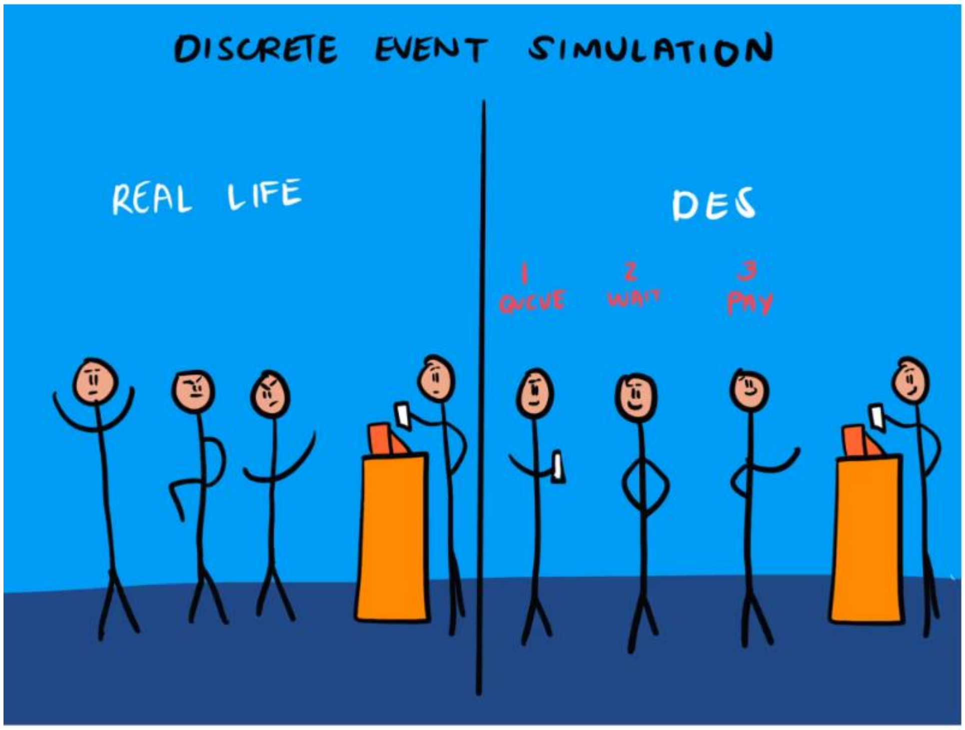
Introduction

A discrete-event simulation (DES) models the operations of a system as a (discrete) sequence of events in time. Each event occurs at a particular instance in time and marks a change of state in the system.[2]

Event simulation, particularly discrete-event simulation, is a fundamental technique in modeling and analyzing complex systems by representing system behavior as sequences of discrete events occurring at specific points in virtual time.



Core Concepts and Terminology

Discrete-event simulation uses a discrete-state model pf the system for simulation (is like a system in certain “boxes” or “states”, and it jumps from one box to another. It doesn’t change smoothly – it changes in steps.[2]

Entity: A distinct object or component within the system that interacts with other components or evolves over time. It can be physical objects (e.g., machines, people) or conceptual items (e.g., orders, tasks)[2]

State: A collection of attributes representing the system’s entities at a specific point in time. The state of the system changes. [2]

Event: An occurrence at a specific point in time that may alter the system's state. For example, this could be an equipment failure, a new request for service, or a traffic light change. Events are the primary drivers of state changes in discrete-choice simulations.[2]

Queue: A waiting line where entities temporarily reside until specific conditions are met or a resource becomes available. This could be a literal line that patients await their next medication injection while at a hospital. Queues are often used to model delays or bottlenecks in the system.[2]

How does discrete event simulation work?

If you're wondering how to build a discrete event simulation model, the process typically starts with identifying entities, events, and system logic.

1. Define the entities (e.g., people, products, patients).
2. Specify the process flow using logic and rules.
3. Assign resources and constraints.
4. Run the simulation to mimic the flow of events over time.
5. Analyze the output to find bottlenecks and optimize performance.

Event-Based Conceptual Model of the Surgical Unit (with Doctor Availability Scheduling)

System overview

The system models patient flow through preparation, surgery, and recovery phases. Limited resources include preparation places, recovery beds, and the operation theatre. A new twist feature doctor availability restricts when surgeries can begin, simulation shift-based scheduling.

Entities and resources

Entity	Description
Patient	The main entity moving through preparation, surgery, and recovery.
Doctor	Represents human resource availability (on shift/ of shift)
PrepPlaces	Limited number of preparation spots.
SurgeryRoom	The operation theatre, used only when both room and doctor are available
RecoveryBeds	Limited number of recovery beds

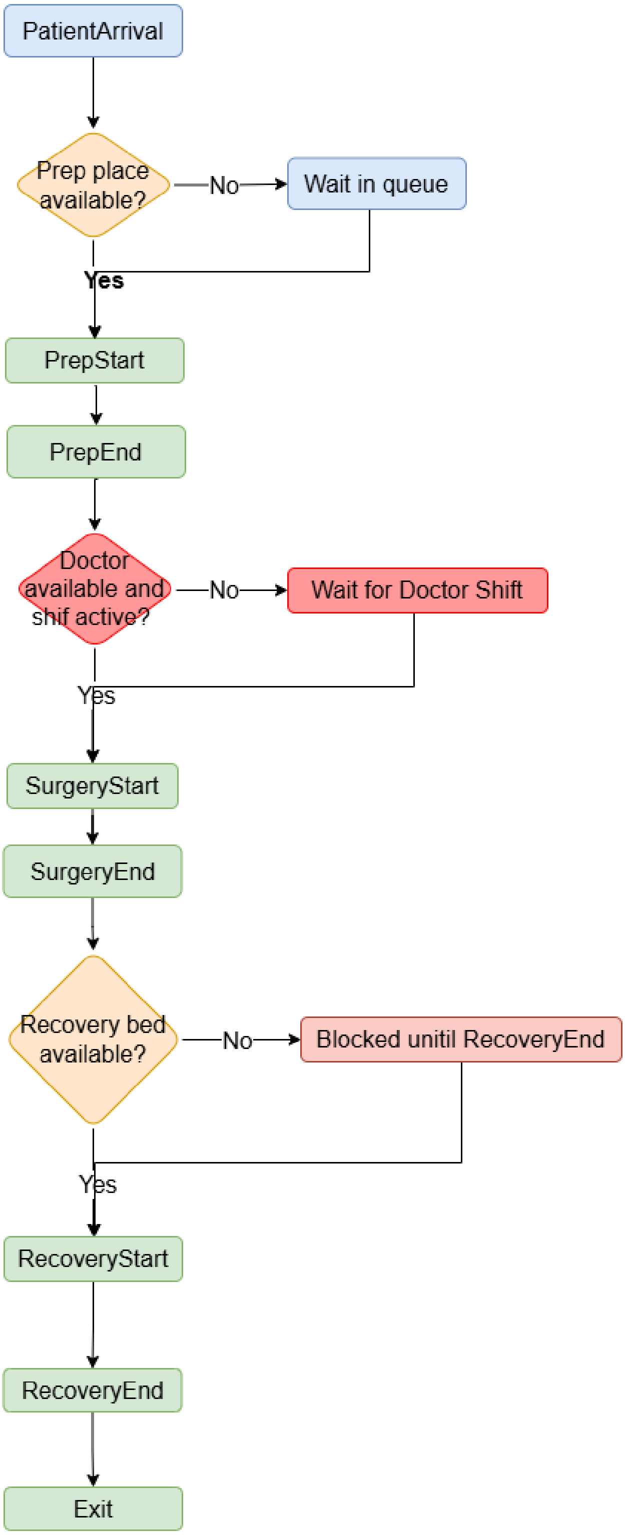
State variables

Variable	Description
n_prep_busy	Number of prep places currently occupied
n_recovery_busy	Number of recovery beds occupied
surgery_state	0 = idle, 1 = in use, 2 = blocked
doctor_state	0 = off shift, 1= available, 3 = busy
queus	Patients waiting for prep or surgery
clock	Simulation time

Discrete Events

Event	Description	State change	Triggers
PatientArrival	New patient enters system	Adds to prep queue or stats prep	Next PatientArrival, possibly PrepStart
PrepStart/ PrepEnd	Start/finnish prep	Occupies or frees prep place	SurgeryStart (if doctor available)
SurgeryStart	Starts surgery when doctor and theatre available	Sets surgery and doctor busy	SurgeryEnd
SurgeryEnd	Finnishes operation	Frees doctor (if still shift), may block if recovery full	RecoveryStart or Blocked state
RecoveryStart/ RecoveryEnd	Occupy or release recovery beds	Changes in recovery occupancy	If blocked, triggers SurgeryResume
DoctorShiftStart	Doctor becomes available for surgeries	Sets dector_state = available	May trigger SurgeryStart for waiting patient
DoctorShiftEnd	Doctor leaves; ongoing surgery finishes, but no new surgry starts	Sets doctor_stat = off shift	Wait for next DoctorShiftStart

Event Flow Diagram (Conceptual)



Stage 1: Patient Arrival and Preparation

Stage 2: Doctor availability check (TWIST FEATURE)

Doctor shift Schedule

- Doctor 1: shift 8am-4pm
- Doctor 1: shift 4pm-12am
- Shifts Overlap: 4pm

Stage 3: Recovery Bed Assignment

Event Flow and State transitions

1. PatientArrival → Resource Check (Prep)
   if prep place available: trigger PrepStart. Otherwise: queue until prep ends elsewhere.
2. PrepEnd → Resource Check (Doctor)
   TWIST: if doctor unavailable/off-shift: enter "Wait for DoctorShift" state. Otherwise trigger SurgeryStart
3. SurgeryEnd → Resource Check (Recovery)Key Outputs
   If recovery bed available: trigger RecoveryStart. Otherwise: surgery room remains blocked until bed frees.
4. RecoveryEnd → Exit
   Patient leaves system. Recovery bed becomes available for next patient

Metrics	**Description**
Throughput Time	Total time per patient from arrival to exit
Surgery utilization	Fraction of time theatre is in use (when doctor available)
Doctor idle time	Fraction of time doctor us on shift but not operation
Blocking probability	Probability surgery room is blocked because recovery full
Queue lengths	Number of patients waiting for surgery when doctor off-shift

Summary

In this event-based model, each event modifies the system's state by allocating or releasing limited resources such as preparation places, the operation theatre, and doctors.

The additional availability of doctor introduces scheduling constraints that influence when surgeries can start, potentially increasing patient waiting times or idle operating-room periods.

1. *Stewart Robinson (2004). Simulation – The practice of model development and use. Wiley.*
2. *SoftwareSim. (2022, March 12). A gentle introduction to discrete event simulation. Retrieved December 3, 2024, from* ***https://softwaresim.com/blog/a-gentle-introduction-to-discrete-event-simulation/***