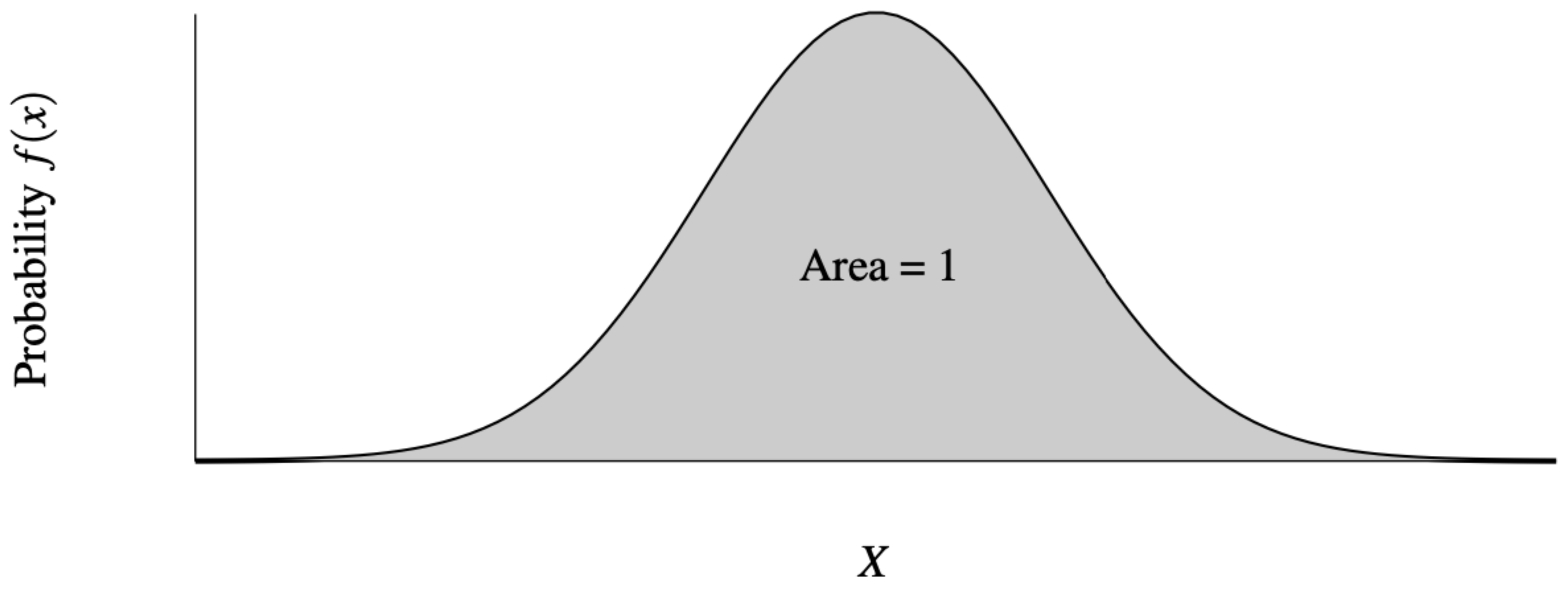
CENG4513
MODELLING & SILMULATION

LECTURE #5

Statistical Probability Distributions

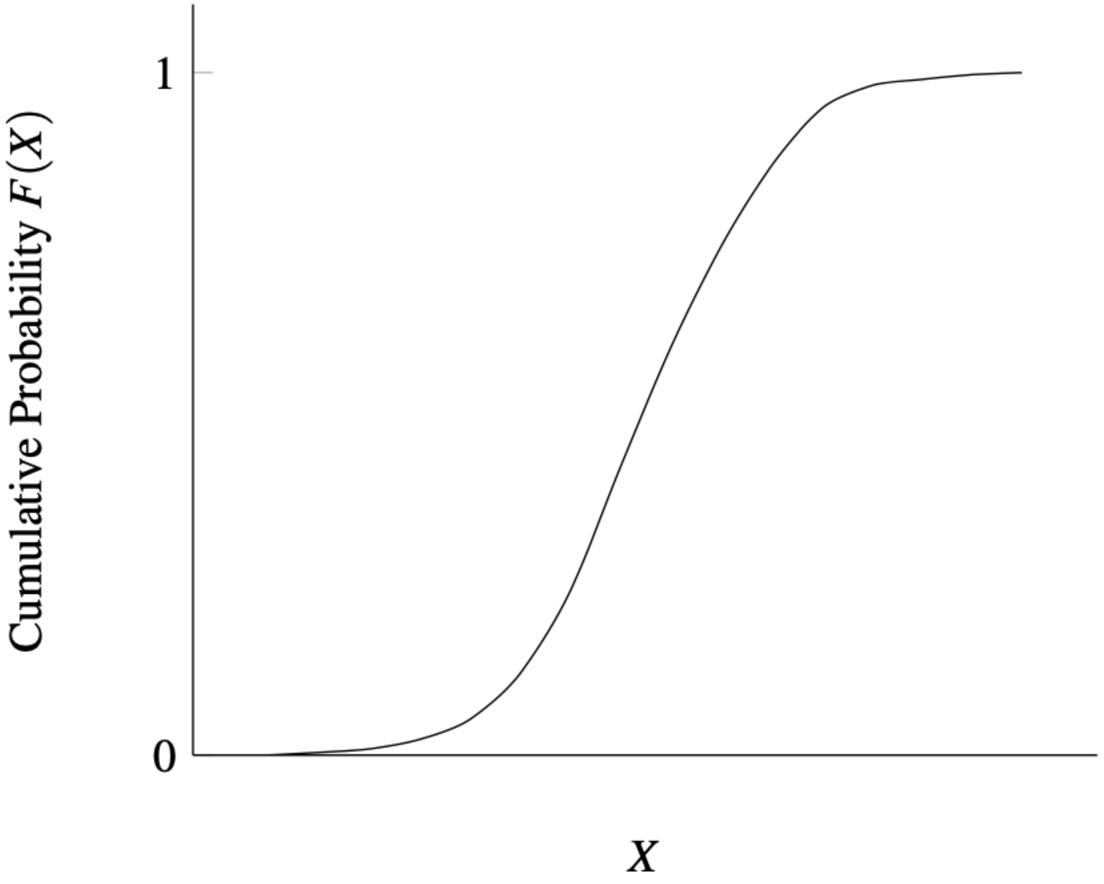
Summary

- Cont. Probability Distribution Function PDF



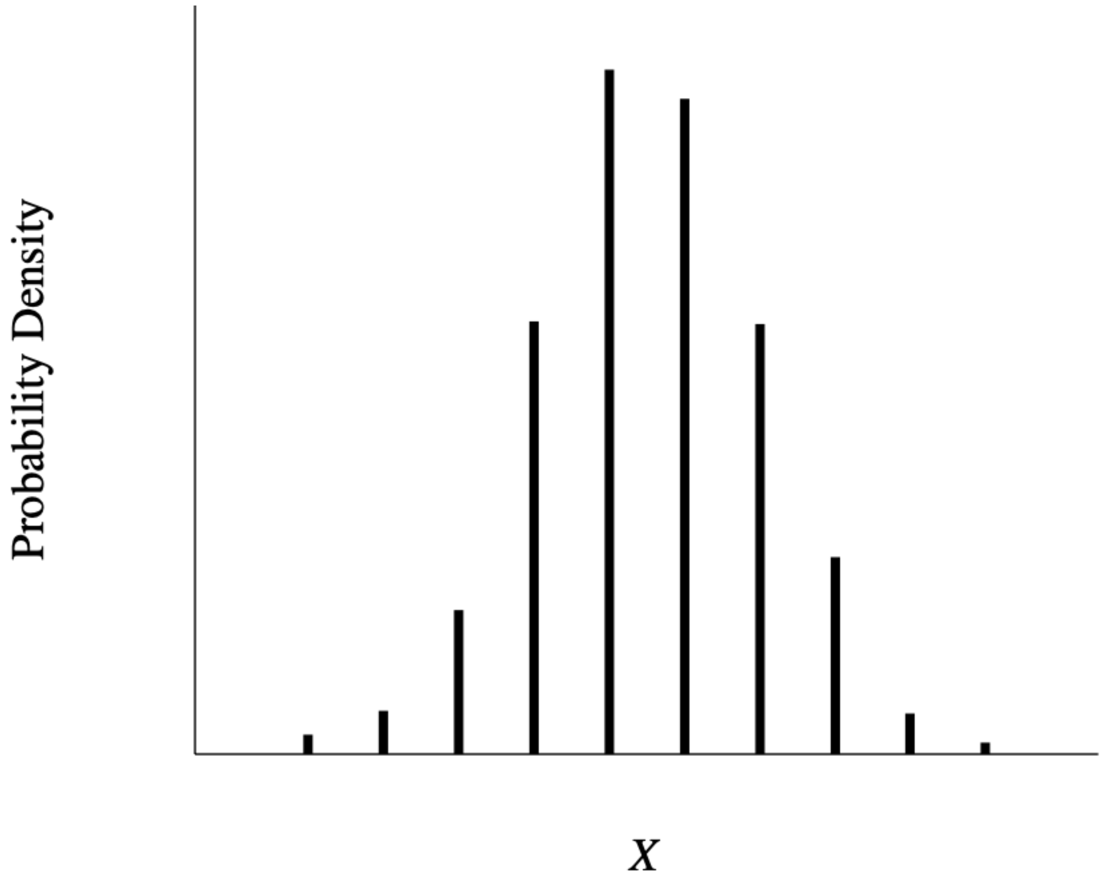
Summary

- Cont. Cumulative Distribution Function CDF



Summary

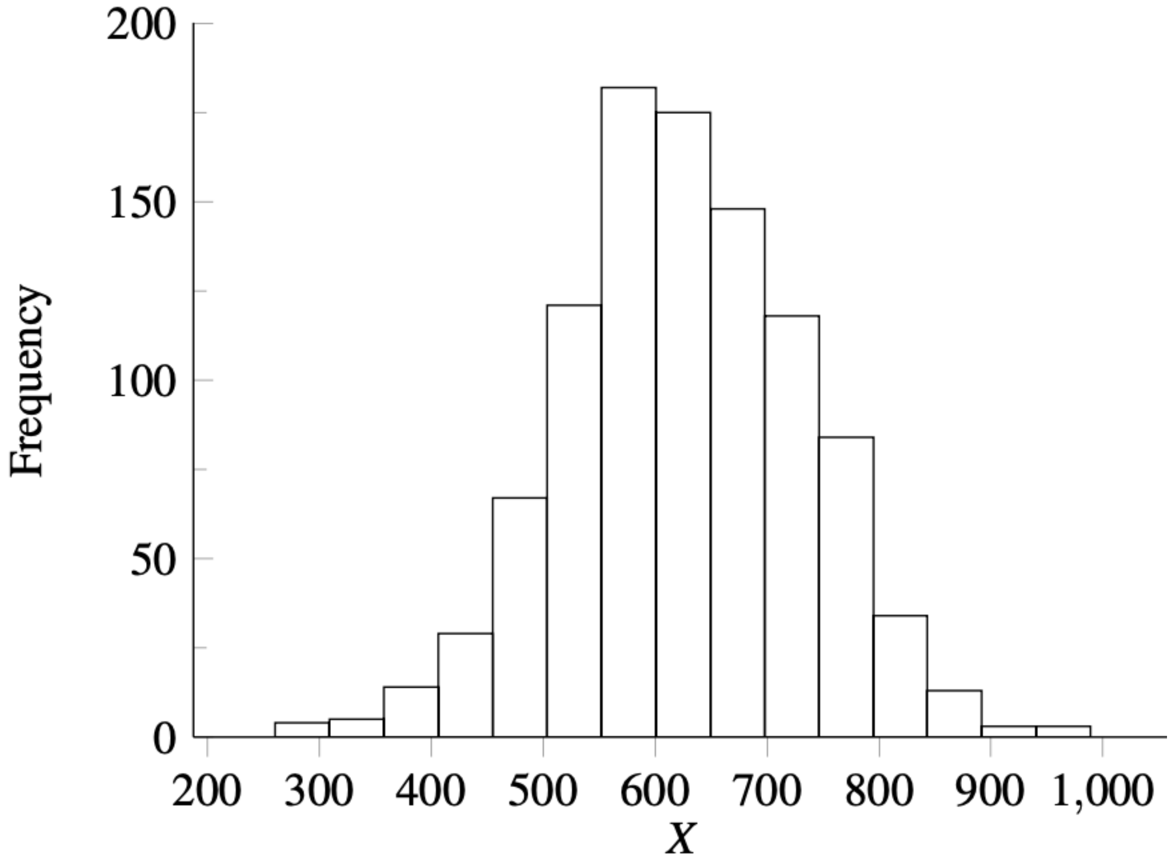
- Discrete Prob. Distribution Function PDF



Summary

- Discrete Prob. Distribution Function PDF

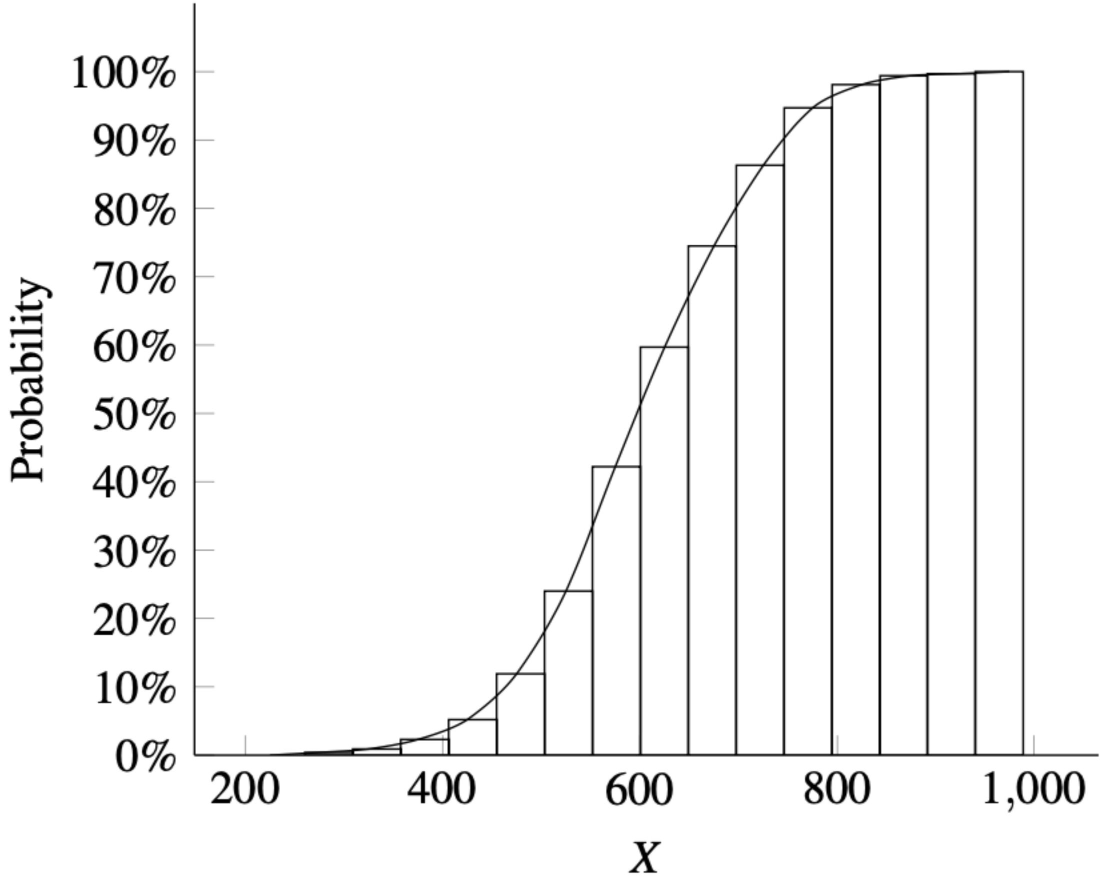
Frequency Histogram



Summary

- ## Cumulative Distribution Function CDF

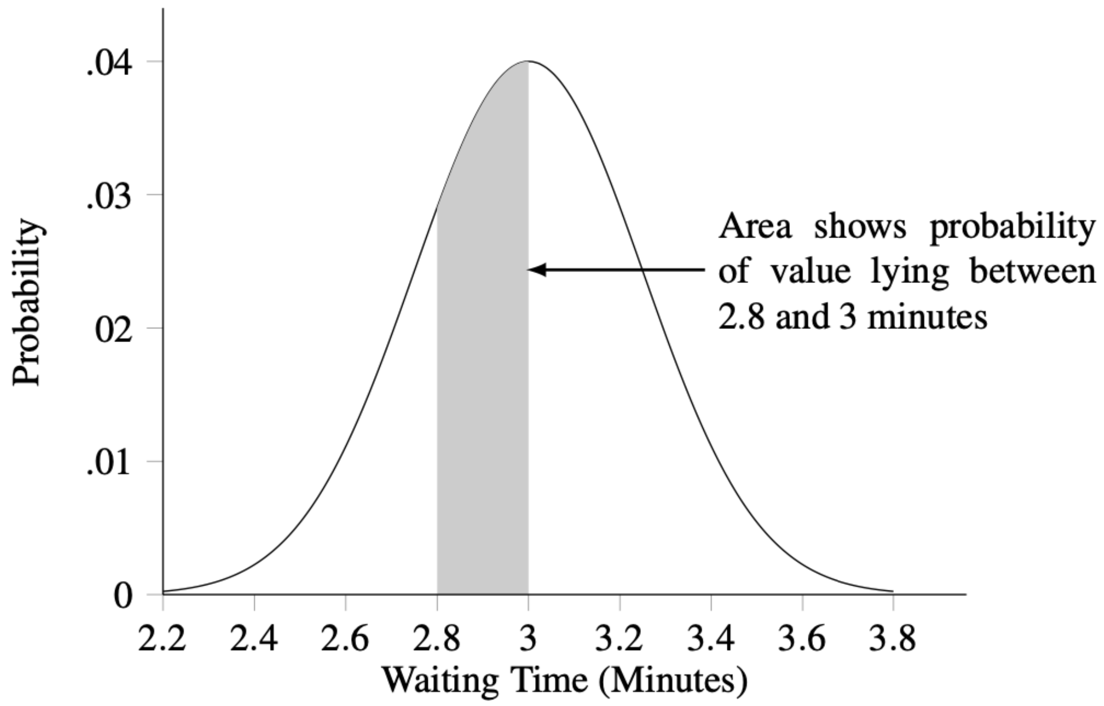
Cumulative Distribution Function from Histogram



Summary

- Cumulative Distribution Function CDF

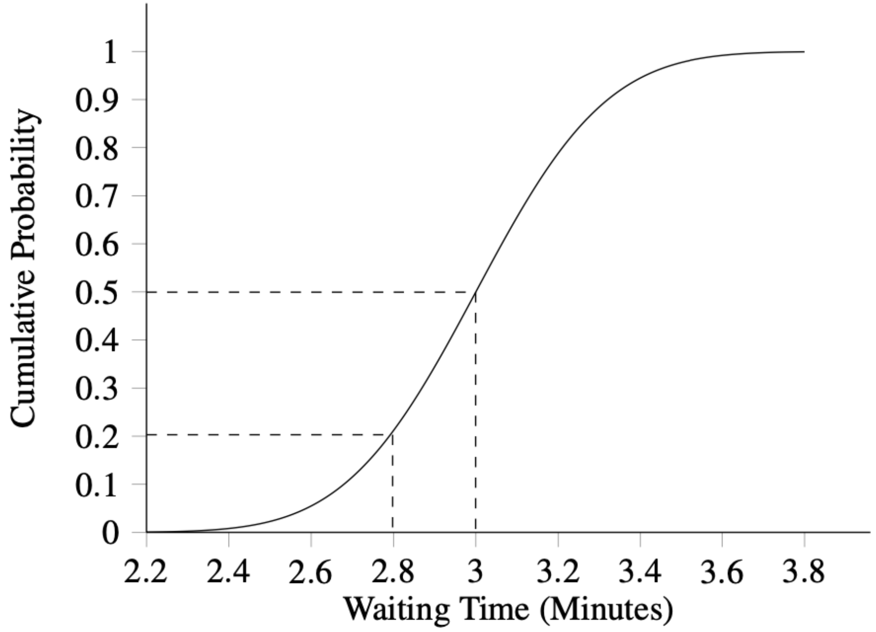
Example:
What is the probability of waiting time being between 2.6 and 2.8?



Summary

▪ PDF & CDF

Example:
What is the probability of waiting time being between 2.6 and 2.8?



The difference of cumulative probabilities (CDF) evaluated at 2.8 and 3.0:

$F(3.0) - F(2.8) = 0.5 - 0.2 = 0.3$

Thus:
There is a 30% chance of waiting time lying between 2.8 and 3.0 mins.

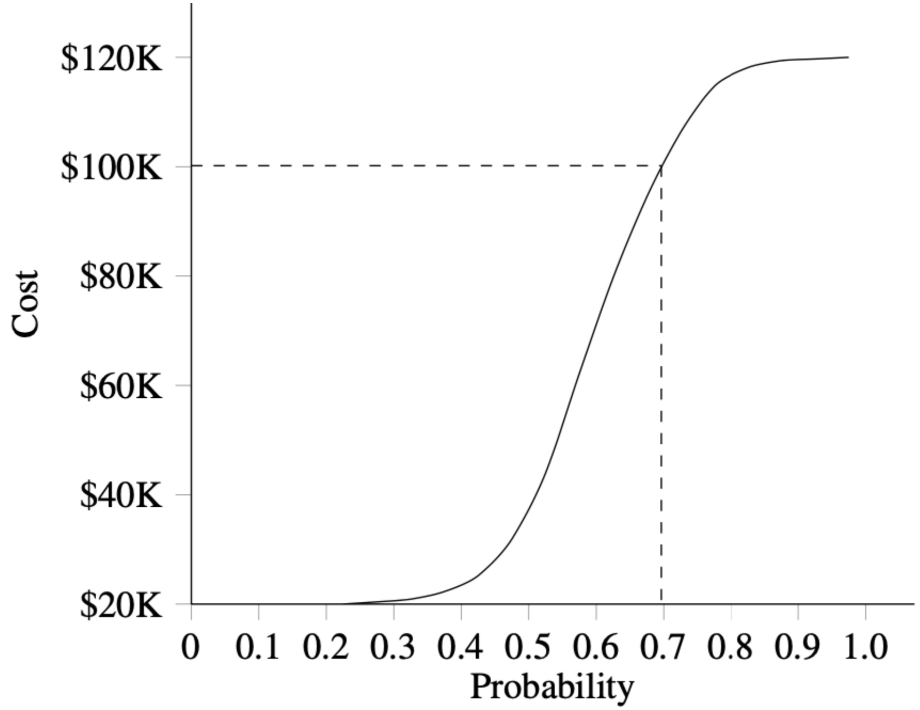
Summary

- PDF & CDF
- The cumulative form of a resulting output distribution can also be used as a confidence level chart. Confidence level refers to the probability of not exceeding a given value.
- For example, the CDF in can be used to determine the confidence level as the cumulative probability for a given value.
- For example, there is a 50% chance that the waiting time will be less than or equal to 3 minutes

Summary

- PDF & CDF

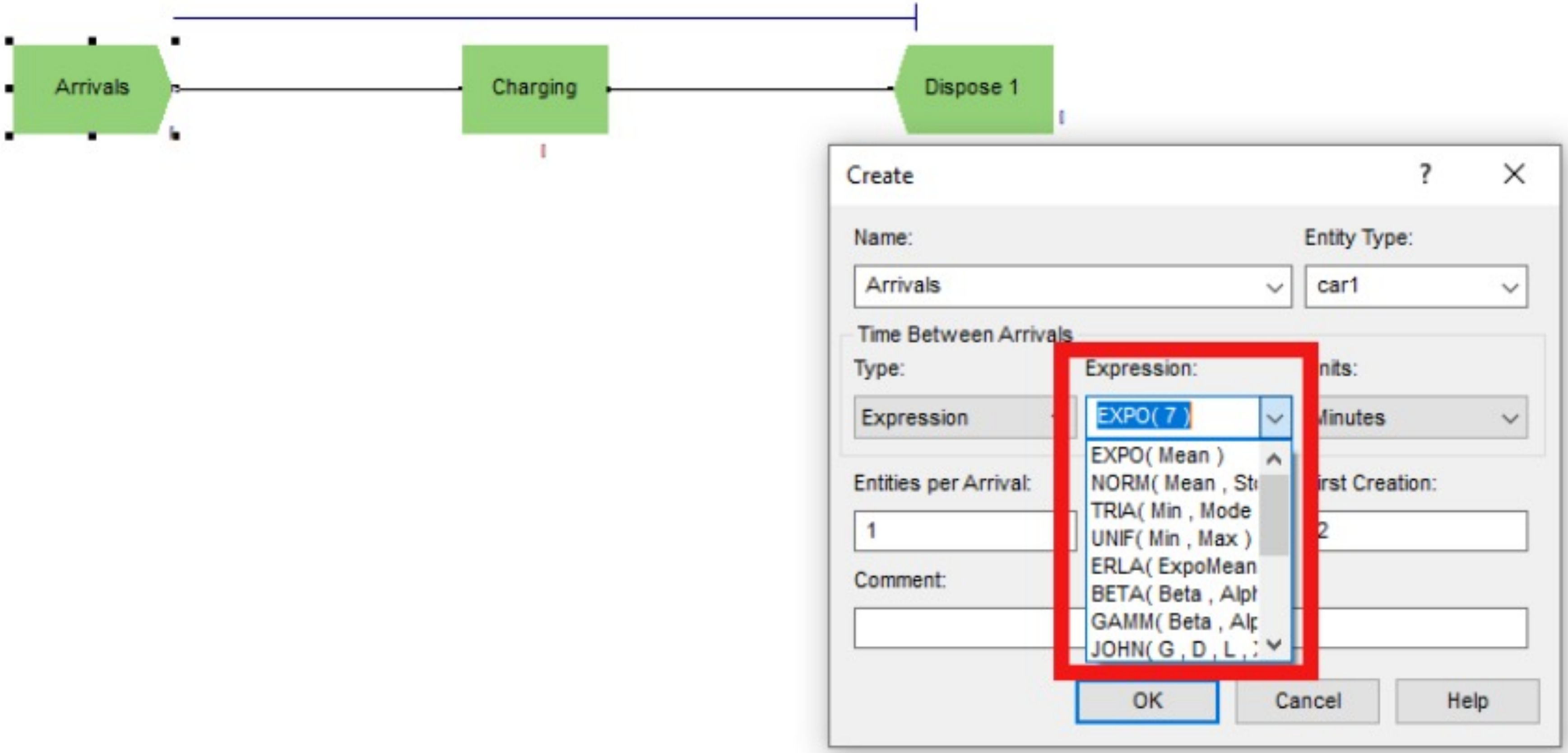
Example:



The figure represents the results of running many simulation runs, and can be used to assess cost risk.

For example, there is a 70% chance that the simulated project will cost $100K or less.

Summary



Arrivals
Charging
Dispose 1
Create
Name:
Entity Type:
Arrivals
car1
Time Between Arrivals
Type:
Expression:
nits:
Expression
EXPO(7)
Minutes
EXPO(Mean)
NORM(Mean , St
TRIA(Min , Mode
UNIF(Min , Max)
ERLA(ExpoMean
BETA(Beta , Alph
GAMM(Beta , Alp
JOHN(G , D , L ,
Entities per Arrival:
irst Creation:
1
2
Comment:
OK
Cancel
Help

Monte Carlo Analysis

Monte Carlo Analysis

- For a model with a number of input parameters each having probability distributions, the distributions are independently and randomly sampled producing various values for each output parameter.

- Using these for a large number of simulation runs, the output parameter values will be characterized as random distributions

- So the term "Monte Carlo" is used broadly to refer to any random sampling method.

Monte Carlo Analysis

- Steps performed for *n* iterations in a Monte Carlo analysis, where an iteration refers to a single simulation run:

1. For each random variable, take a sample from its probability distribution function and calculate its value.
2. Run a simulation using the random input samples and compute the corresponding simulation outputs.
3. Repeat the above steps until *n* simulation runs are performed.
4. Determine the output probability distributions of selected dependent variables using the *n* values from the runs.

Monte Carlo Analysis

Inverse Transform

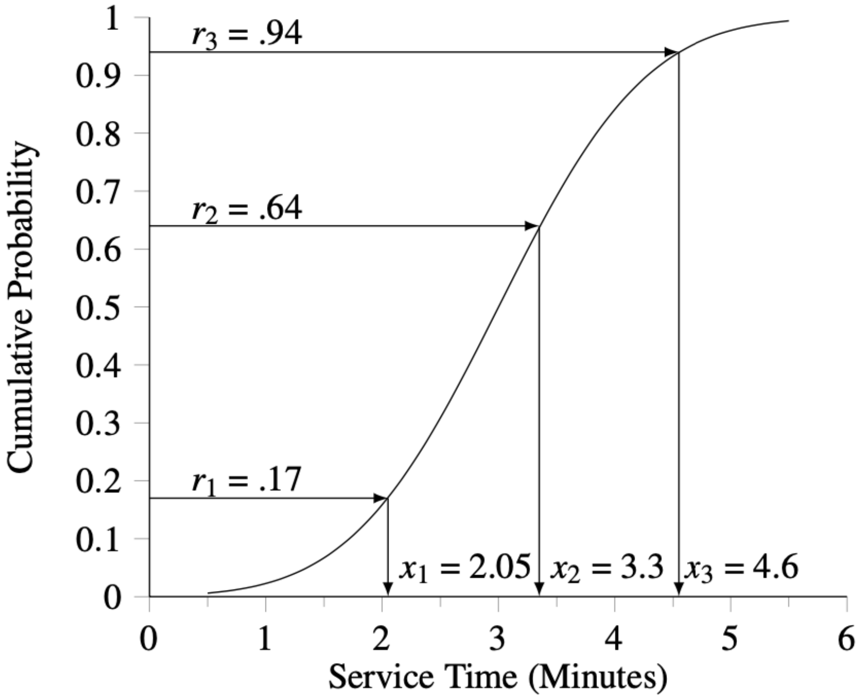
- Monte Carlo uses random numbers to sample from known probability distributions to determine specific outcomes.
- First a random number *r* is chosen that is uniformly distributed between 0 and 1, denoted as *U*(0,1). It is set equal to the cumulative distribution, *F*(*x*) = *r*, and *x* is solved for.
- A particular value *ri* gives a value *xi*, which is a particular sample value of *X*. It can be expressed as:

$$x_i = F^{-1}(r_i)$$

Monte Carlo Analysis

Inverse Transform

Example:



First a U (0, 1) is generated, then the cumulative distribution is used to find the corresponding random variate.

In the example from a normal distribution,

- the first random number of r = 0.17 generates a value of x = 2.05,
- the second random draw of r = 0.64 produces a value of x = 3.3,
- and the third draw of r = .94 generates x = 4.6

Goodness-of-fit

- Statistical Tests enables to compare between two distributions, also known as **Goodness-of-Fit**.
- The **goodness-of-fit** of a statistical model describes how well it fits a set of observations.
- Measures of goodness of fit typically summarize the discrepancy between observed values and the values expected under the model in question
- Goodness-of-fit means how well a statistical model fits a set of observations

Goodness-of-fit

- Chi-Square (χ^2) Test
- Kolmogorov-Smirnov (K-S) Test

Distributions in Excel

Distributions in Excel

- Standard Uniform Distribution *U(0,1)*

  $=RAND()$

- Uniform Distribution *U(a,b)*

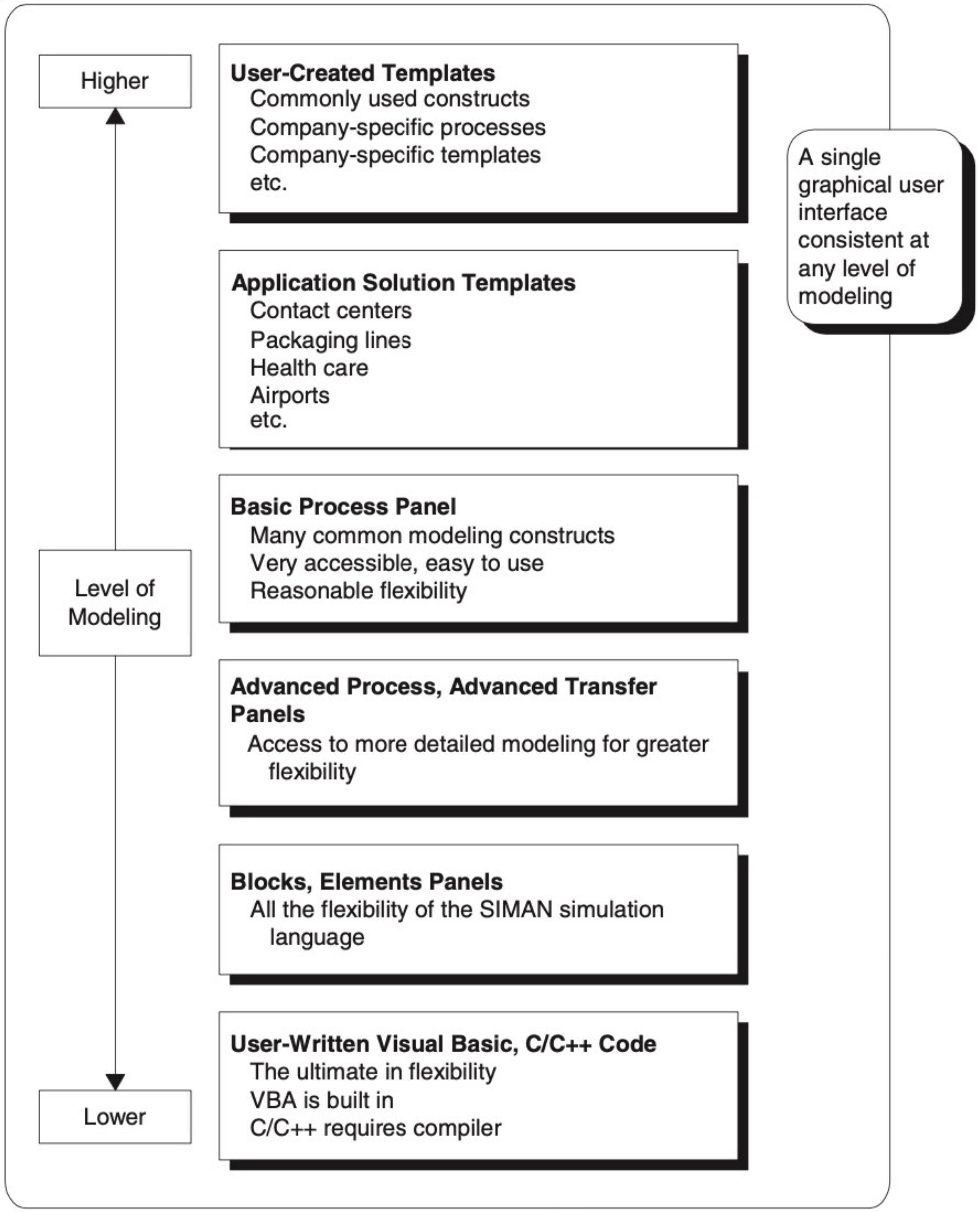
  $= a+(b-a) * RAND()$

- Bernoulli Trial *(p)*

  $= IF(RAND() <= p, 1, 0)$

Where p is determined by the user.

Simulation
&
Arena


Higher
User-Created Templates
Commonly used constructs
Company-specific processes
Company-specific templates
etc.
A single graphical user interface consistent at any level of modeling
Application Solution Templates
Contact centers
Packaging lines
Health care
Airports
etc.
Basic Process Panel
Many common modeling constructs
Very accessible, easy to use
Reasonable flexibility
Level of Modeling
Advanced Process, Advanced Transfer Panels
Access to more detailed modeling for greater flexibility
Blocks, Elements Panels
All the flexibility of the SIMAN simulation language
User-Written Visual Basic, C/C++ Code
The ultimate in flexibility
VBA is built in
C/C++ requires compiler
Lower

Simulation in Arena

Pieces of a Simulation Model

Entity:

- Most simulations involve "players" called *entities* that
  - move around,
  - change status,
  - affect and are affected by other entities and the state of the system,
  - and affect the output performance measures.
- Entities are the *dynamic* objects in the simulation—they usually are created, move around for a while, and then are disposed of as they leave.

Simulation in Arena

Pieces of a Simulation Model

Entity:

- It's possible, though, to have entities that never leave but just keep circulating in the system.
- All entities have to be created, either by you or automatically by the software
- *They're created when they arrive, move through the queue (if necessary), are served by the drill press, and are then disposed of as they leave*

Simulation in Arena

Pieces of a Simulation Model

Attribute:

- To individualize entities, you attach *attributes* to them.
- An attribute is a common characteristic of all entities, but with a specific value that can differ from one entity to another.
- They indicate the characteristics for each individual entity.
- It's up to you to figure out what attributes your entities need, name them, assign values to them, change them as appropriate, and then use them when it's time (that's all part of modeling)

Simulation in Arena

Pieces of a Simulation Model

<u>Attribute:</u>

- The most important thing to remember about attributes is that their values are tied to specific entities.

- The same attribute will generally have different values for different entities, just as different parts have different due dates, priorities, and color codes

Simulation in Arena

Pieces of a Simulation Model

(Global) Variable:

- A *variable* (or a *global* variable) is a piece of information that reflects some characteristic of your system, regardless of how many or what kinds of entities might be around.
- You can have many different variables in a model, but each one is unique.
- There are two types of variables:
  - Arena built-in variables (number in queue, number of busy servers, current simulation clock time, and so on) and
  - User-defined variables (mean service time, travel time, current shift, and so on)

Simulation in Arena

Pieces of a Simulation Model

(Global) Variable:

- In contrast to attributes, variables are not tied to any specific entity, but rather pertain to the system at large. They're accessible by all entities, and many can be changed by any entity.
- Variables are used for lots of different purposes. For instance, the time to move between any two stations in a model might be the same throughout the model
- Variables can also represent something that changes *during* the simulation, like the number of parts in a certain subassembly area of the larger model

Example: Some built-in Arena variables for a model include the status of the server (busy or idle), the time (simulation clock), and the current length of the queue.

Simulation in Arena

Pieces of a Simulation Model

Resource:

- Entities often compete for service from *resources* that represent things like personnel, equipment, or space in a storage area of limited size.
- An entity *seizes* (units of) a resource when available and *releases* it (or them) when finished
- An entity (like a part) could need simultaneous service from multiple resources (such as a machine and a person).

Simulation in Arena

Pieces of a Simulation Model

Resource:

- A resource can represent a group of several individual servers, each of which is called a *unit* of that resource (i.e. several identical “parallel” agents at an airline ticketing counter.)
- The number of available units of a resource can be changed during the simulation run to represent agents going on break or opening up their stations if things get busy.

Simulation in Arena

Pieces of a Simulation Model

<u>Queue:</u>

- When an entity can't move on, it needs a place to wait, which is the purpose of a *queue*
- In Arena, queues have names and can also have capacities to represent, for instance, limited floor space for a buffer.
- You'd have to decide as part of your modeling how to handle an entity arriving at a queue that's already full.

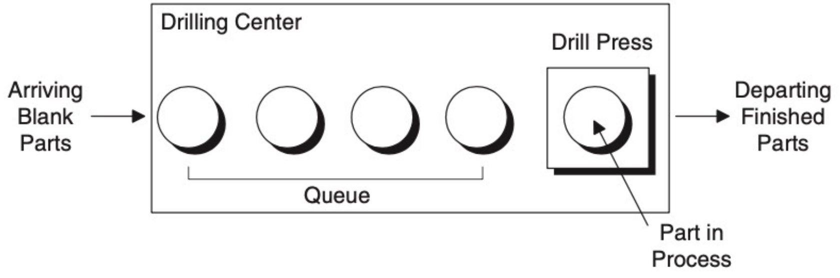
Simulation in Arena

Pieces of a Simulation Model

Statistical Accumulators:

- To get your output performance measures, you have to keep track of various intermediate *statistical-accumulator variables* as the simulation progresses.
- All of these accumulators should be initialized to 0. When something happens in the simulation, you have to update the affected accumulators in the appropriate way.

•The number of parts produced so far
•The total of the waiting times in queue so far
•The number of parts that have passed through the queue so far (for the average waiting-time output measure)
•The longest time spent in queue we've seen so far
•The total of the time spent in the system by all parts that have departed so far
•The longest time in system we've seen so far
•The area so far under the queue-length curve $Q(t)$
•The highest level that $Q(t)$ has so far attained
•The area so far under the server-busy function $B(t)$



Simulation in Arena

Pieces of a Simulation Model

Event:

- An *event* is something that happens at an instant of (simulated) time that might change attributes, variables, or statistical accumulators.

Example:

There are three kinds of events in drill example:

1. **Arrival:** A new part enters the system.
2. **Departure:** A part finishes its service at the drill press and leaves the system.
3. **The End:** The simulation is stopped at time 20 minutes.

Simulation in Arena

Pieces of a Simulation Model

Simulation Clock:

- The current value of time in the simulation is simply held in a variable called the *simulation clock*.
- Unlike real time, the simulation clock does not take on all values and flow continuously.
- It jumps from the time of one event to the time of the next event scheduled to happen. Since nothing changes between events, there is no need to waste (real) time looking at (simulated) times that don't matter.

Simulation in Arena

Pieces of a Simulation Model

Simulation Clock:

- How the event is executed clearly depends on what kind of event it is as well as on the model state at that time, but in general could include updating variables and statistical accumulators, altering entity attributes, and placing new event records onto the calendar.

Simulation in Arena

Pieces of a Simulation Model

Starting & Stopping:

- You have to determine the appropriate starting conditions, how long a run should last, and whether it should stop at a particular time (i.e. 20 minutes) or whether it should stop when something specific happens (like as soon as 100 finished parts are produced).
- It's important to think about this and make assumptions consistent with what you're modeling; these decisions can have just as great an effect on your results as can more obvious things like values of input parameters (such as interarrival-time means, service-time variances, and the number of machines).

Simulation in Arena

Outline of the Action:

- **Arrival:** A new part shows up.
  - Schedule the next new part to arrive later, at the next arrival time, by placing a new event record for it onto the event calendar.
  - Update the time-persistent statistics (between the last event and now).
  - Store the arriving part's time of arrival (the current value of the clock) in an attribute, which will be needed later to compute its total time in system and possibly the time it spends waiting in the queue.
  - If the drill press is idle, the arriving part goes right into service (experiencing a time in queue of zero), so the drill press is made busy and the end of this part's service is scheduled.
  - On the other hand, if the drill press is already busy with another part, the arriving part is put at the end of the queue and the queue-length variable is incremented.

Simulation in Arena

Outline of the Action:

- **Departure:** The part being served by the drill press is done and ready to leave.
    - Increment the number-produced statistical accumulator.
    - Compute and tally the total time in system of the departing part by taking the current value of the clock minus the part's arrival time (stored in an attribute during the Arrival event).
    - Update the time-persistent statistics.
    - If there are any parts in queue, take the first one out, compute and tally its time in queue (which is now ending), and begin its service at the drill press by scheduling its departure event (that is, place it on the event calendar).
    - On the other hand, if the queue is empty, then make the drill press idle. Note that in this case, there's no departure event scheduled on the event calendar.

Simulation in Arena

Outline of the Action:

- **The End:** The simulation is over.
  - Update the time-persistent statistics to the end of the simulation.
  - Compute and report the final summary output performance measures.

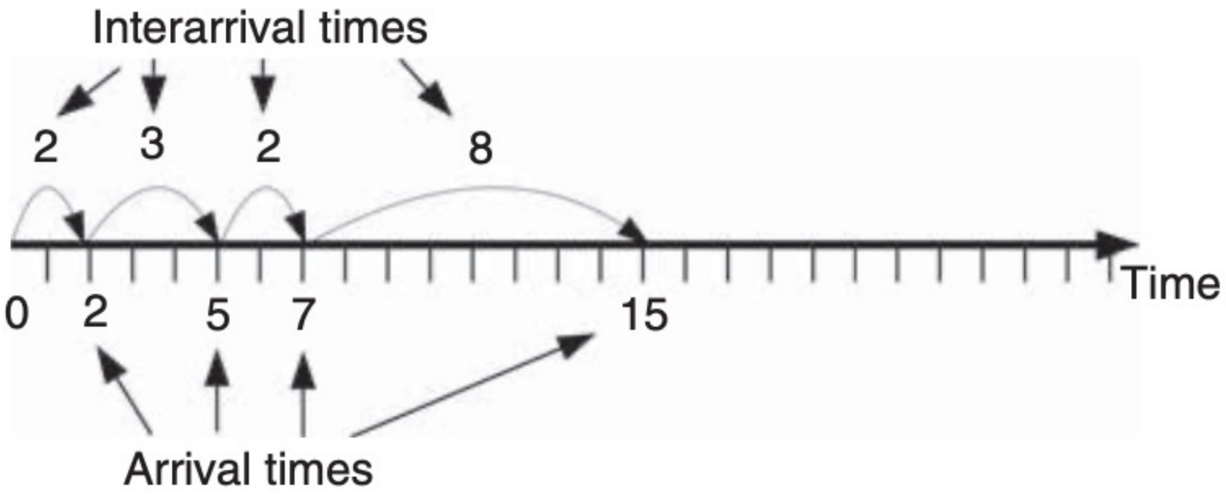
Simulation in Arena

Example: Bank

- Let us consider a simple bank which has two tellers that serve customers from a single waiting line
- The bank opens at 9 am, which can be used to designate time zero for the simulation
- When a customer arrives, the number of customers in the bank will increase by one. An arrival of a customer will change the state of the bank. Thus, the arrival of a customer can be considered an event.

- Since an event causes actions that result in a change of system state, ask: What are the actions that occur when a customer arrives to the bank?
  1. The customer enters the waiting line if there is an available teller, the customer will immediately exit the line and the available teller will begin to provide service.
  2. If there are no tellers available, the customer will remain waiting in the line until a teller becomes available.

Simulation in Arena

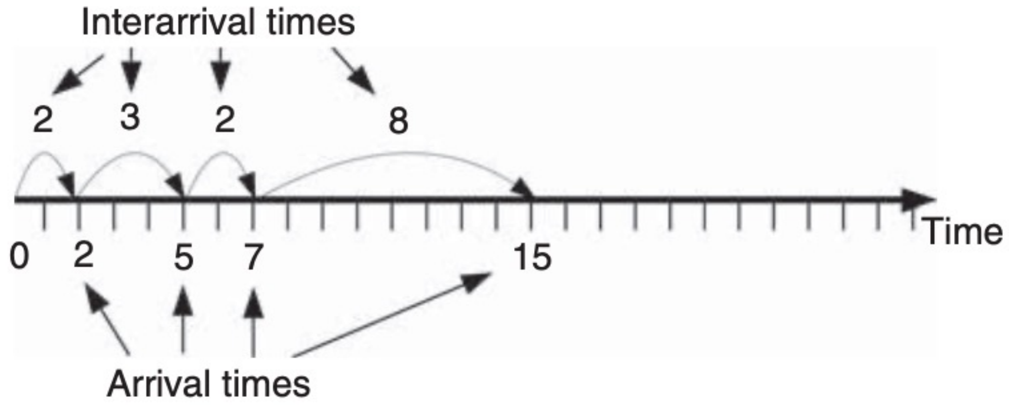


Customer	Time of Arrival	Time between Arrival
1	2	2
2	5	3
3	7	2
4	15	8

Simulation in Arena

Example: Bank

- Consider the arrival of the first customer.
- Since the bank opens at 9 am with no customer and all the tellers idle, the first customer will enter and immediately exit the queue at time 9:02 am (or time 2) and then begin service.
- After the customer completes service, the customer will exit the bank.
- When a customer completes service (and departs the bank) the number of customers in the bank will decrease by 1.



Customer	Time of Arrival	Time between Arrival
1	2	2
2	5	3
3	7	2
4	15	8

Simulation in Arena

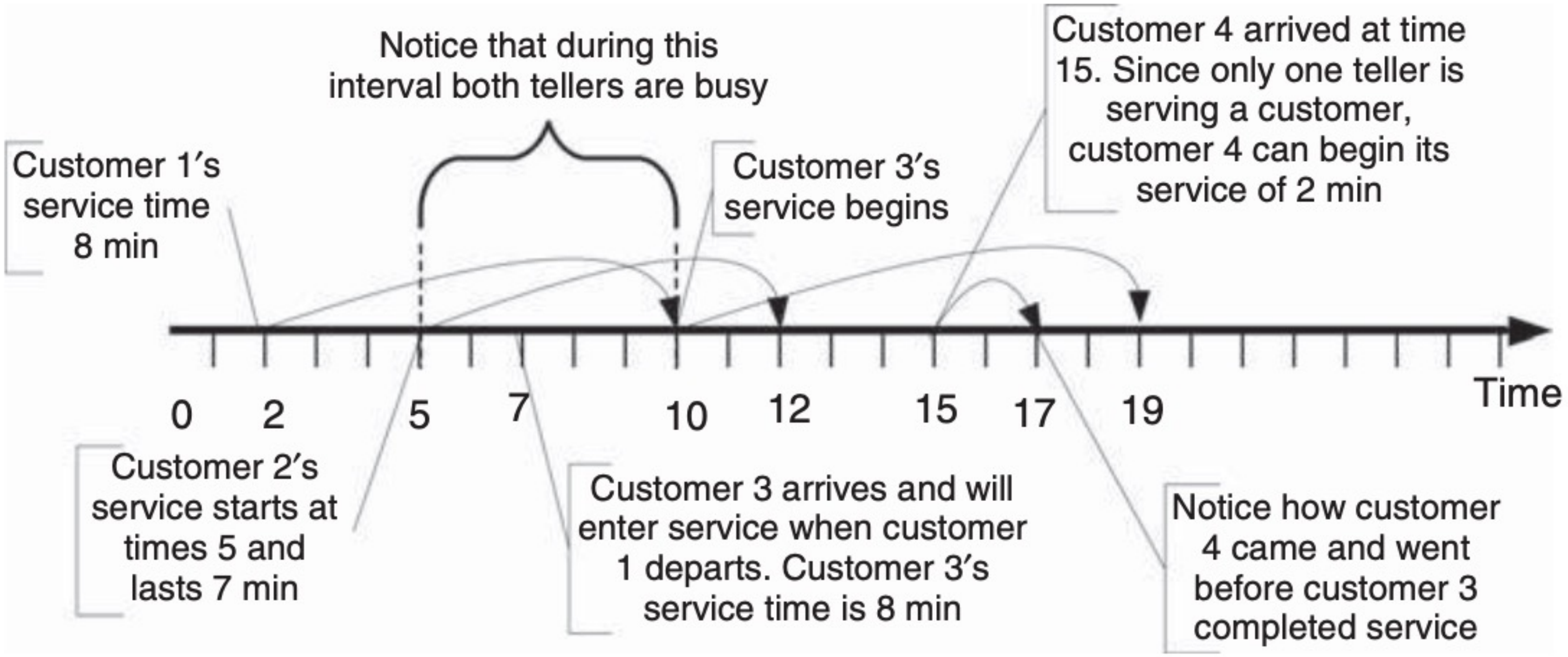
Example: Bank

- Some actions occur when a customer completes service.
- These actions correspond to the second event associated with this system: the customer service completion event.

- What are the actions that occur at this event?
  - The customer departs the bank.
  - If there are waiting customers, the teller indicates to the next customer that he/she will serve the customer. The customer will exit the waiting line and will begin service with the teller.
  - If there are no waiting customers, then the teller will become idle.

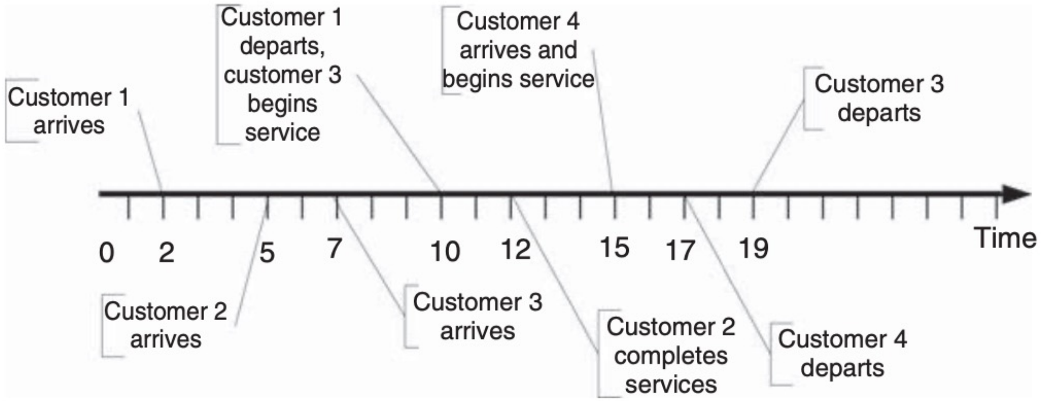
Simulation in Arena

Example: Bank



Simulation in Arena

Example: Bank



Time	Event	Comment
0	Bank opens	
2	Arrival	Customer 1 arrives, enters service for 8 min, one teller becomes busy
5	Arrival	Customer 2 arrives, enters service for 7 min, the second teller becomes busy
7	Arrival	Customer 3 arrives, waits in queue
10	Service completion	Customer 1 completes service, customer 3 exits the queue and enters service for 9 min
12	Service completion	Customer 2 completes service, no customers are in the queue so a teller becomes idle
15	Arrival	Customer 4 arrives, enters service for 2 min, one teller becomes busy
17	Service completion	Customer 4 completes service, a teller becomes idle
19	Service completion	Customer 3 completes service