Discrete Event and Process Oriented Simulation (1)

- **Discrete event simulation**
  - Discrete event simulation concerns the modeling of a system by a representation in which the state of the system changes at discrete time points known as *events*.
  - An event is a time point where the states of the system changes or other major things happen (such as the start and the end of the simulation).
  - For example, when simulating a *GI/GI/1* queue, the two major events are customer arrival and customer departure.
  
  ![Discrete Event Simulation Diagram](image)

  - An arrival event increases the number of customer in queue by 1 (if the server is busy) or changes the state of the server to busy (if the server was idle).
  - A departure event decreases the number of customer in queue by 1 or changes the server state to idle.
  - Simulated time is measured through the *simulation clock*.
The simulation clock is initialized at 0 and the times of occurrence of future events are determined.

The simulation clock is then advanced sequentially to the time of the “next” event.

The system is observed at event times and statistics of performance measures are collected.

"Drama is life with the dull bits cut out" - A. Hitchcock

- **General scheme of a discrete event simulation**
• Discrete Event simulation of GI/GI/1 to estimate mean delay
  ➢ The “initialize” event sets all statistical counters to zero.
  ➢ Assuming system starts empty. The first event is arrival of customer 1. After that, a sequence of arrival and departure events are simulated and the simulation clock is advanced.
  ➢ The simulation continues until a stopping rule is reached.
  ➢ The following is done every time an arrival event occurs.

![Flowchart](image-url)
➢ The following is done every time a departure event occurs.

➢ To see this procedure in action, check out the power point example. (See also the C code in Law’s book if you like.)
• **Simplified Discrete Event “hand” simulation of GI/GI/1**

- If the objective of the simulation is only to estimate the mean delay in the system and in the queue, then a simplified procedure can be used to simulate GI/GI/1.
- In this case, the stopping rule of the simulation is to simulate \( n \) service completions.
- This is done by generating inter-arrival and service times for each of the \( n \) customers and estimating the waiting times in a tabular recursive form.
- Specifically, Let \( AT_i, S_i, SS_i, SC_i, W_i \) and \( W_q^i \) be the arrival, service, start of service, service completion (departure), waiting in the system and in queue times for customer \( i \).
- Then, assuming the system starts idle

\[
SS_1 = AT_1, \quad SC_1 = SS_1 + S_1, \quad W_q^1 = 0, \quad W_1 = S_1.
\]
\[
SS_i = \max(AT_i, SC_{i-1}), \quad SC_i = SS_i + S_i,
\]
\[
W_q^i = SS_i - AT_i, \quad W_i = W_q^i + S_i, \quad i = 2, \ldots, n.
\]
- For example consider the queue simulated in the power point detailed simulation.
- Suppose the objective now is to simulate 6 service completions.
Then, the simplified hand simulation looks as follows.

<table>
<thead>
<tr>
<th>Customer, $i$</th>
<th>$AT_i$</th>
<th>$S_i$</th>
<th>$SS_i$</th>
<th>$SC_i$</th>
<th>$W^i_q$</th>
<th>$W_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2.9</td>
<td>0</td>
<td>2.9</td>
<td>0</td>
<td>2.9</td>
</tr>
<tr>
<td>2</td>
<td>1.73</td>
<td>1.76</td>
<td>2.9</td>
<td>4.66</td>
<td>1.17</td>
<td>2.93</td>
</tr>
<tr>
<td>3</td>
<td>3.08</td>
<td>3.39</td>
<td>4.66</td>
<td>8.05</td>
<td>1.58</td>
<td>4.97</td>
</tr>
<tr>
<td>4</td>
<td>3.79</td>
<td>4.52</td>
<td>8.05</td>
<td>12.57</td>
<td>4.26</td>
<td>8.78</td>
</tr>
<tr>
<td>5</td>
<td>4.41</td>
<td>4.46</td>
<td>12.57</td>
<td>17.03</td>
<td>8.16</td>
<td>12.62</td>
</tr>
<tr>
<td>6</td>
<td>18.69</td>
<td>4.36</td>
<td>18.69</td>
<td>23.05</td>
<td>0</td>
<td>4.36</td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>15.17</td>
<td>36.56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Estimates of the mean waiting time in the system and in the queue are then computed as

$$\hat{W}_q = \frac{15.17}{6} = 2.53, \quad \hat{W} = \frac{36.56}{6} = 6.09 \text{ minutes}.$$

Such simulation schemes, which are not truly discrete-event, commonly arise in logistics and finance among other areas.

This is sometimes referred to as Monte Carlo simulation (remember the Crabs simulation).

They can be conveniently done in a spreadsheet, preferably with the help of a specialized “Add-in” software such as @Risk or Crystal Ball.

VBA macro Excel can be used to code repetitive tasks and collect statistics in the absence of specialized Add-ins.

**Process-oriented simulation**

Process oriented simulation (applied on “high level” software such as Arena) allows the user to skip the tedious event-driven details of discrete event simulation.
- It is based on viewing the simulation in terms of the experience of entities that flow through the system.
- The logic behind process oriented simulation is similar to constructing a flowchart for the flow of an entity in the system.
- For example, imagine you are a customer in the bank example, then your experience will be as follows:
  - Create yourself (i.e. arrive to the system).
  - Note your arrival time.
  - Wait in queue if the server is busy.
  - Once the server is idle, move out of the queue and “seize” the server.
  - Note your start of service time and estimate your waiting time in queue.
  - Stay in service (i.e. “delay” your self) for an amount of time equal to your service time.
  - “Release” the server.
  - “Dispose” of your bank visit (leave the system).
- Process oriented simulation is a way to think about the system and simulate it in a software like Arena.
- The “real” simulation is done in a discrete event fashion.
- Writing your own discrete event simulation gives you flexibility (at the expense of lots of coding effort).