

Construction Operation Simulation

Lecture #6

Coding Simulation Models

Amin Alvanchi, PhD

Construction Engineering and Management



[LinkedIn](#)



[Instagram](#)



[WebPage](#)



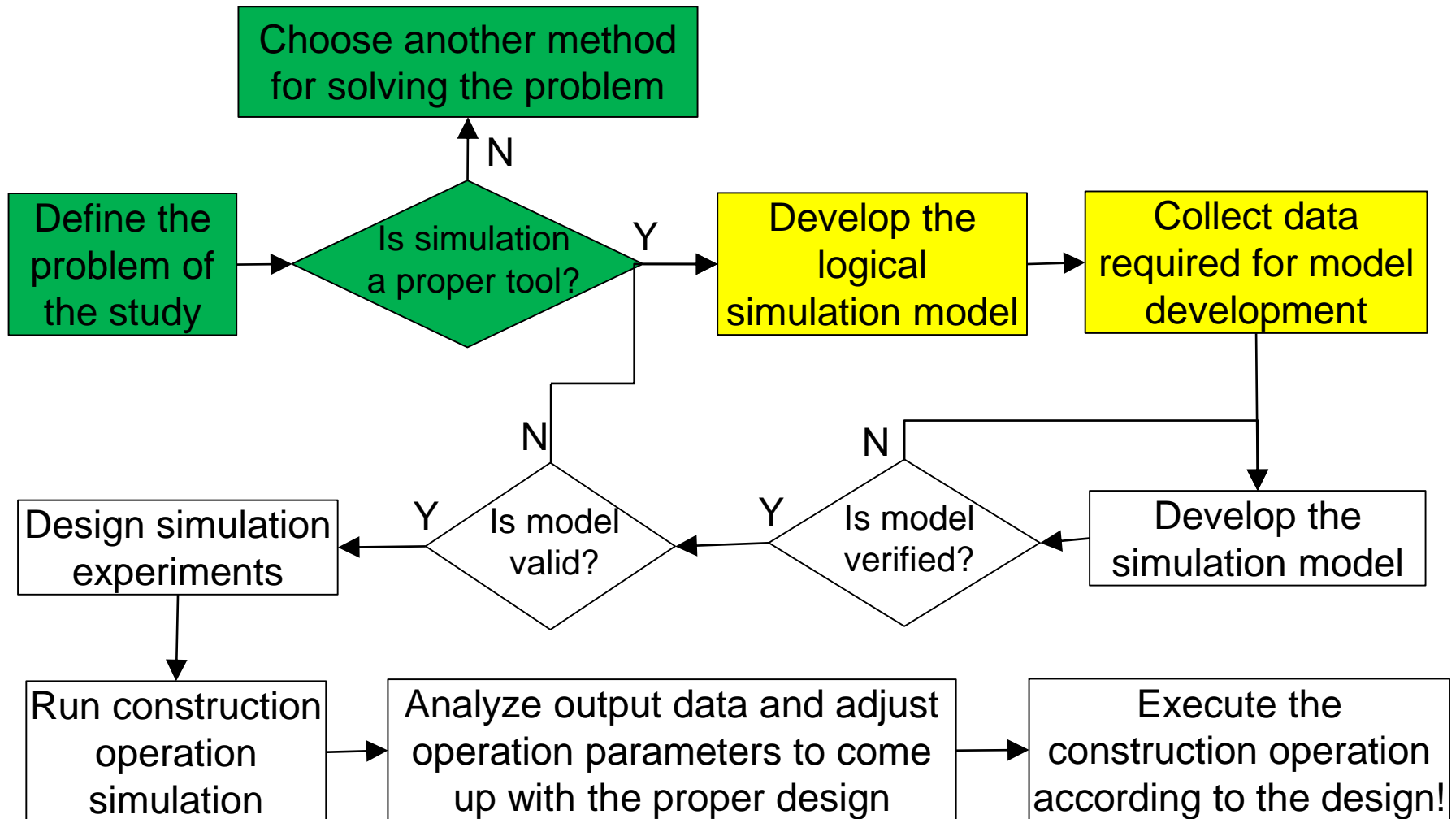
Outline

2

- Introduction
- Coding simulation models

Main steps in simulation studies

3



Coding simulation models

4

- In hand simulation we followed the logic that a computer program pursues to simulate construction operations.
- At this stage we are going to explain computer programming codes needed for running simulation model!
- After preparing computer codes of a simulation model we can run the simulation model many times with the minimum time required!
- Steps required for developing a simulation program of a construction operation is very similar to hand simulation:
 - 1) Recognize model elements
 - 2) Determine event procedures
 - 3) Determine initialization procedure
 - 4) Determine main classes/ objects
 - 5) Start coding:
 - 1) Prepare the interface
 - 2) Define main classes
 - 3) Code event procedures
 - 4) Define initialization procedures
 - 5) Code simulation loop
 - 6) Start running the simulation model

Coding simulation models

5

- **Single queue example:** Suppose an asphalt plant with a single asphalt loading station. Asphalt hauling trucks from different paving project arrive to the asphalt plant randomly during the day (from 7am to 5pm). Time between truck arrival time has an exponential distribution with the average of 15 minutes. If loading station is idle, truck directly goes to the loading station and asphalt loading gets started. Loading duration has a normal distribution with the average of 10 minutes and standard deviation of 2 minutes. If loading station is busy, truck stays in the line and waits until its turn. Code the simulation model of the operation!



Coding simulation models

6

1- Recognize model elements:

Entity: Truck

Resource: Asphalt plant

Activity:

- Time between arrival: $\exp(\lambda=4 \text{ per hour or } 1/15 \text{ per minute})$
- Loading: $N(10 \text{ minutes}, 2 \text{ minutes})$

Event:

Truck arrival; Loading completion

System state:

Num of trucks in asphalt loading Queue; asphalt plant status;

Coding simulation models

7

2- Determine event procedures

Truck arrives to the asphalt plant:

- 1) Schedule next truck arrival event!
- 2) If asphalt plant is busy, number of waiting trucks for loading is increased by one.
- 3) If asphalt plant is idle, make the plant busy; Schedule a new loading complete event!

Loading completion event:

- 1) If there is no truck waiting to be loaded, make asphalt plant idle
- 2) If there are trucks waiting to be loaded, number of trucks waiting to be loaded is decreased by one; Schedule a new loading completion event!

Coding simulation models

8

3- Determine initialization procedure

- Set plant status idle
- Set queue length as 0
- Set simulation time as 10 hours or 600 minutes
- Schedule first truck's arrival event

4- Determine main classes/ objects

- Truck-Entity
- Asphalt Plant
- Engine

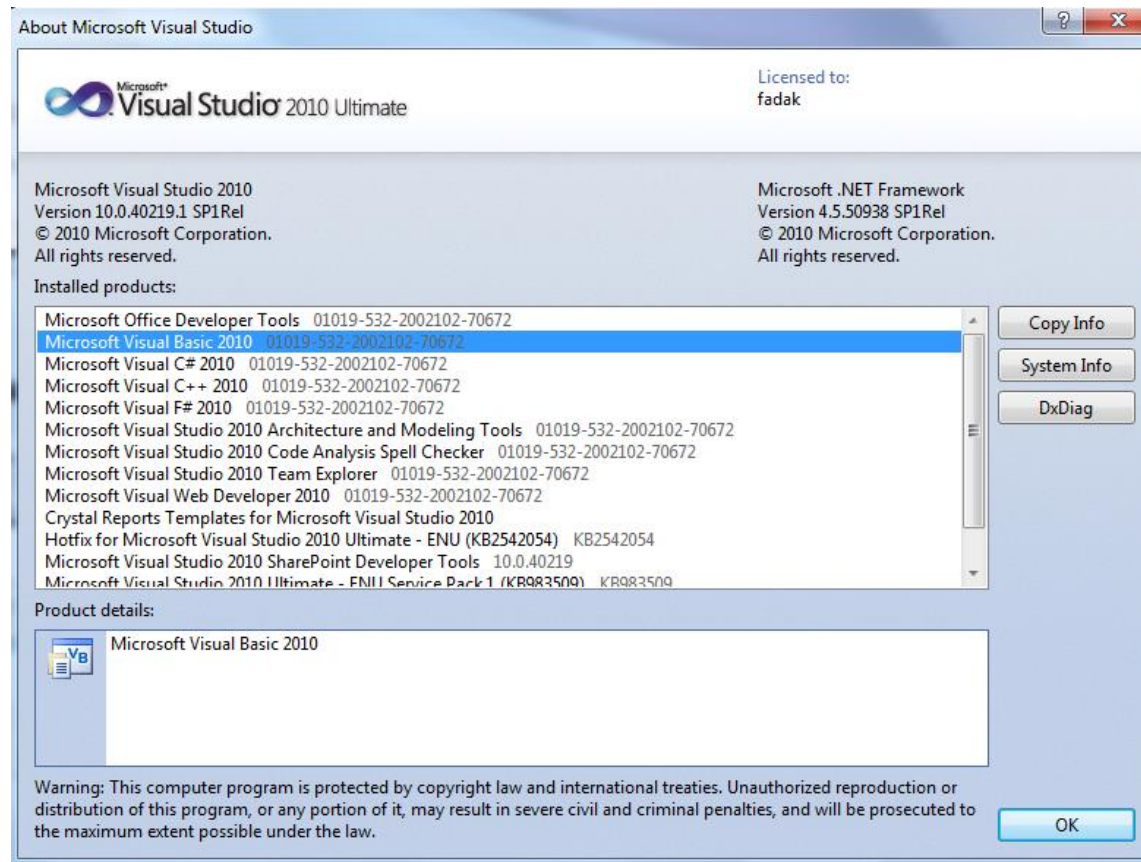
Coding simulation models

9

4- Code event procedures (Codes are in VB.NET)



Hands on coding simulation model

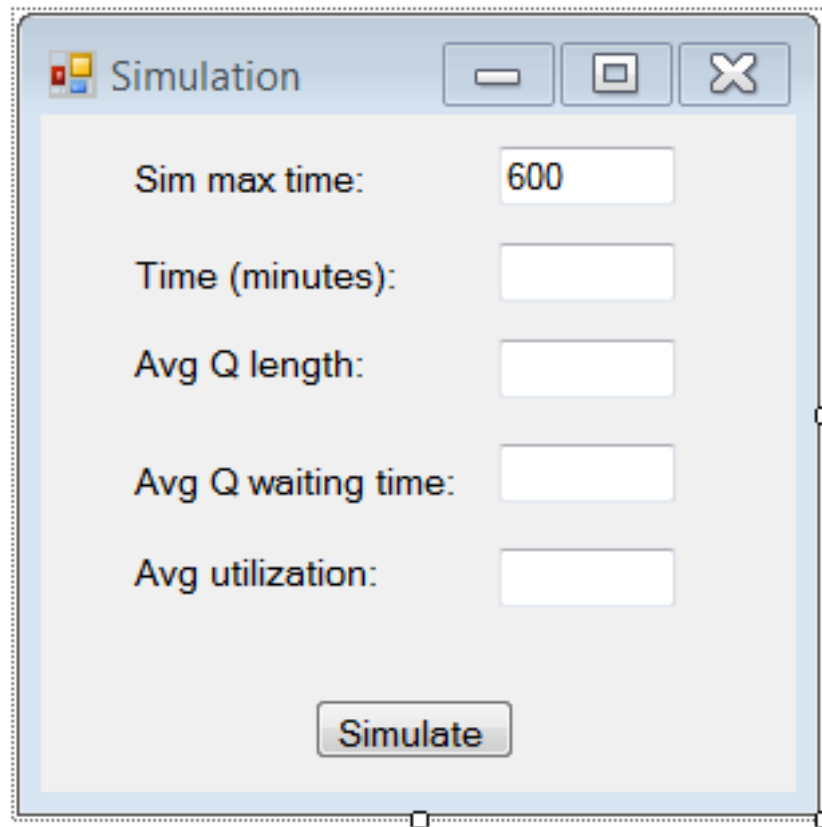


Coding simulation models

10

5- Start Coding

1) Prepare interface



The image shows a software window titled "Simulation". It contains five input fields for simulation parameters, each with a label to its left. The first field, "Sim max time:", contains the value "600". The other four fields are empty. Below these fields is a button labeled "Simulate". The window has a standard title bar with minimize, maximize, and close buttons.

Label	Value
Sim max time:	600
Time (minutes):	
Avg Q length:	
Avg Q waiting time:	
Avg utilization:	

Simulate

Coding simulation models

11

5- Start Coding

2) Classes



What classes do you expect to code?

truck, simulation engine, asphalt plant



What should be the order of the class declarations?

Coding simulation models

12

5- Start Coding

2) Classes-Truck



What are expected attributes (properties), methods and events of a truckEntity?

Truck Id? Truck capacity? Truck age? ...

Truck:

Properties: truckId, enterQTime

Methods:-

Events: -

Coding simulation models

13

5- Start Coding

2) Classes-Truck

```
Public Class TruckEntity
```

```
    Public truckId As Integer
```

```
    Public enterQTime As Double
```

```
End Class
```

Coding simulation models

14

5- Start Coding

2) Classes – Engine

Engine:

Properties: FEL

Method: ScheduleNewEvent, nextEvent

Event: -

Coding simulation models

15

5- Start Coding

2) Classes – Engine

Type of FEL?

FEL is defined as a collection of FELItems which should include
eventType, entity info, and event time.

How do we define an event type?

Coding simulation models

16

5- Start Coding

2) Classes – Engine – Auxiliary Enumeration

‘Enumerations are integer types making the editor handy using a textual entry format

```
Public Enum eventType
```

```
    TruckArrival = 1
```

```
    LoadingCompletion = 2
```

```
End Enum
```

How do we define an FELItem?

Coding simulation models

17

5- Start Coding

2) Classes – Engine – Auxiliary Class

```
Public Class FELItem  
    Public simEvent As eventType  
    Public entity As New TruckEntity  
    Public eventTime As Double  
End Class
```

Coding simulation models

18

5- Start Coding

2) Classes – Engine

```
Public Class Engine
```

```
    Public FEL As New Collection
```

```
    Public curTime As Double = 0
```

```
    Public Sub scheduleNewEvent(ByVal newFELItem As FELItem)
```

```
        FEL.Add(newFELItem, newFELItem.entity.truckId.ToString)
```

```
    End Sub
```

Coding simulation models

19

5- Start Coding

2) Classes – Engine

```
Public Function nextEvent() As FELItem
    nextEvent = New FELItem
    nextEvent.entity.truckId = 0
    nextEvent.eventTime = Double.MaxValue
    For Each myFELItem As FELItem In FEL
        If myFELItem.eventTime < nextEvent.eventTime Then
            nextEvent.entity = myFELItem.entity
            nextEvent.eventTime = myFELItem.eventTime
            nextEvent.simEvent = myFELItem.simEvent
        End If
    Next
    curTime = nextEvent.eventTime
    If nextEvent.entity.truckId > 0 Then
        FEL.Remove(nextEvent.entity.truckId.ToString)
    End If
End Function
End Class
```

Coding simulation models

20

5- Start Coding

2) Classes - Asphalt Plant



What are expected attributes (properties) of the asphalt plant?

plant status, Q

Furthermore, we need to collect some statistics which become attributes of the plant:

avgQLength, avgQWaitingTime, avgUtilization, totalTruckServred



What are expected methods (or activities) of the asphalt plant?

loading, timeBetweenTruckArrivals, findingFirstTruckInQ



What are expected events of the asphalt plant?

truckArrival, loadingCompletion

Coding simulation models

21

5- Start Coding

2) Classes - Asphalt Plant –Property- Status -Auxiliary Enumeration



What can be a resource status?

Idle or Busy

'Using Enumeration

```
Public Enum resourceStatus
```

```
    Idle = 0
```

```
    Busy = 1
```

```
End Enum
```

Coding simulation models

22

5- Start Coding

2) Classes - Asphalt Plant – Property- Q



How is plant Q property defined?

Plant Q is a collection of truckEntity Objects!

Coding simulation models

23

5- Start Coding

2) Classes - Asphalt Plant -Property

Public Class AsphaltPlant

'Properties

Public status As resourceStatus

Public Q As New Collection

Public avgQLength As Double = 0

Public avgQWaitingTime As Double = 0

Public avgUtilization As Double = 0

Public totalTruckServred As Integer = 0

Auxiliary
attributes, can
be defined
either Private
or Public

Public totalTruckPassedQ As Integer = 0

Public lastStatusReportTime As Double = 0

Public totalTruckArrived As Integer = 0

Coding simulation models

24

5- Start Coding

2) Classes - Asphalt Plant – Methods –Auxiliary Functions

```
Public Function loadingTime() As Double
```

```
    Dim z As Double
```

```
    z = Math.Pow(-2 * Math.Log(Rnd()), 0.5) * Math.Cos(2 * Math.PI * Rnd())
```

```
    loadingTime = 10 + z * 2
```

```
End Function
```

```
Public Function nextTruckArrival() As Double
```

```
    nextTruckArrival = -15 * Math.Log(Rnd())
```

```
End Function
```


Coding simulation models

25

5- Start Coding

2) Classes - Asphalt Plant – Methods –Auxiliary Functions

```
Public Function firstTruckInQToLoading() As TruckEntity
    firstTruckInQToLoading = New TruckEntity
    firstTruckInQToLoading.EnterQTime = Double.MaxValue
    For Each myTruckItem As TruckEntity In Q
        If myTruckItem.EnterQTime < firstTruckInQToLoading.EnterQTime Then
            firstTruckInQToLoading.EnterQTime = myTruckItem.EnterQTime
            firstTruckInQToLoading.truckId = myTruckItem.truckId
        End If
    Next
    Q.Remove(firstTruckInQToLoading.truckId.ToString)
End Function
```

Coding simulation models

26

5- Start Coding

2) Classes - Asphalt Plant – Event- newTruckArrived

Public Event newTruckArrived(ByRef myEngine As Engine, ByRef newTruck As TruckEntity)



Where is newTruckArrived event raised?

Only in the main simulation engine procedure!



What does happen when a newTruckArrive?

Event occurrence (or handler) procedure is run!

Truck arrives to the asphalt plant:

- 1) Schedule next truck arrival event!
- 2) If asphalt plant is busy, number of waiting trucks for loading is increased by one.
- 3) If asphalt plant is idle, make the plant busy; Schedule a new loading complete event!

Coding simulation models

27

5- Start Coding

2) Classes - Asphalt Plant – Event- newTruckArrived

```
Public Sub newTruckArrival(ByRef myEngine As Engine, ByRef newTruck As TruckEntity)
    Handles Me. newTruckArrived

    '1) Schedule next truck arrival event!
    Dim newFELItem As New FELItem
    totalTruckArrived = totalTruckArrived + 1
    newFELItem.entity.truckId = totalTruckArrived + 1
    newFELItem.eventTime = myEngine.curTime + nextTruckArrival()
    newFELItem.simEvent = eventType.TruckArrival
    myEngine.scheduleNewEvent(newFELItem)
    '2)Handle Truck arrival in plant if plant is busy
    If status = resourceStatus.Busy Then 'prepare and send truck to the Q
        'calculate utilization
        avgUtilization = (avgUtilization * lastStatusReportTime + (myEngine.curTime -
lastStatusReportTime)) / myEngine.curTime
        'calculate avg Q length
        avgQLength = (avgQLength * lastStatusReportTime + Q.Count * (myEngine.curTime -
lastStatusReportTime)) / myEngine.curTime
        'add truck to the Q
        newTruck.EnterQTime = myEngine.curTime
        Q.Add(newTruck, newTruck.truckId.ToString)
```

Coding simulation models

28

5- Start Coding

2) Classes - Asphalt Plant – Event- newTruckArrived

```
Else '3) if plants status is idle send the truck to the loading
    'calculate avg waiting time considering current truck's waiting time as 0
    avgQWaitingTime = avgQWaitingTime * totalTruckPassedQ / (totalTruckPassedQ + 1)
    'calculate utilization
    avgUtilization = avgUtilization * lastStatusReportTime / myEngine.curTime
    'calculate avg Q length
    avgQLength = avgQLength * lastStatusReportTime / myEngine.curTime
    'Change Status to busy
    status = resourceStatus.Busy
    'Increase number of trucks passed the Q
    totalTruckPassedQ = totalTruckPassedQ + 1
    'Schedule loading completion event
    newFELItem = New FELItem
    newFELItem.entity.truckId = newTruck.truckId
    newFELItem.eventTime = myEngine.curTime + loadingTime()
    newFELItem.simEvent = eventType.LoadingCompletion
    myEngine.scheduleNewEvent(newFELItem)
End If
lastStatusReportTime = myEngine.curTime
End Sub
```

Coding simulation models

29

5- Start Coding

2) Classes - Asphalt Plant – Event– loadingCompleted

Public Event loadingCompleted(ByRef myEngine As Engine)



Don't we need the Entity Object?

No, because the entity exits from the plant (or our system) and we do not need it anymore. We only need to remove the entity from the memory!

Again, this event is raised in the main simulation engine procedure,

The loading completion occurrence procedure (or handler):

- 1) If there is no truck waiting to be loaded, make asphalt plant idle
- 2) If there are trucks waiting to be loaded, number of trucks waiting to be loaded is decreased by one; Schedule a new loading completion event!

Coding simulation models

30

5- Start Coding

2) Classes - Asphalt Plant – Event- loadingCompleted

```
Public Sub loadingCompletion(ByRef myEngine As Engine) Handles
    Me.loadingCompleted

    totalTruckServred = totalTruckServred + 1
    'calculate utilization
    avgUtilization = (avgUtilization * lastStatusReportTime +
(myEngine.curTime - lastStatusReportTime)) / myEngine.curTime
    '1) If there is no truck waiting to be loaded, make asphalt plant idle
    If Q.Count = 0 Then
        'calculate avg Q length
        avgQLength = avgQLength * lastStatusReportTime / myEngine.curTime
        'Change Status to idle
        status = resourceStatus.Idle
```

Coding simulation models

31

5- Start Coding

2) Classes - Asphalt Plant – Event- loadingCompleted

Else '2)If trucks waiting in the Q, number of trucks waiting to be loaded is decreased by one;
Schedule a new loading completion event!

'calculate avg Q length

$$\text{avgQLength} = (\text{avgQLength} * \text{lastStatusReportTime} + \text{Q.Count} * (\text{myEngine.curTime} - \text{lastStatusReportTime})) / \text{myEngine.curTime}$$

'Retrieve first truck from the Q to be loaded

Dim myTruckToLoading As TruckEntity = firstTruckInQToLoading()

'calculate avg waiting time

$$\text{avgQWaitingTime} = (\text{avgQWaitingTime} * \text{totalTruckPassedQ} + \text{myEngine.curTime} - \text{myTruckToLoading.EnterQTime}) / (\text{totalTruckPassedQ} + 1)$$

'Increase number of trucks passed the Q

$$\text{totalTruckPassedQ} = \text{totalTruckPassedQ} + 1$$

Dim newFELItem As New FELItem

$$\text{newFELItem.entity.truckId} = \text{myTruckToLoading.truckId}$$

$$\text{newFELItem.eventTime} = \text{myEngine.curTime} + \text{loadingTime}()$$

$$\text{newFELItem.simEvent} = \text{eventType.LoadingCompletion}$$

$$\text{myEngine.scheduleNewEvent}(\text{newFELItem})$$

End If

$$\text{lastStatusReportTime} = \text{myEngine.curTime}$$

End Sub

Coding simulation models

32

5- Start Coding

2) Classes - Asphalt Plant – Event - RaiseEvent

```
Public Function eventHappened(ByRef myEngine As Engine, ByRef newTruck As
    TruckEntity, ByVal myEvent As eventType) As eventType
    eventHappened = myEvent
    Select Case eventHappened
        Case eventType.TruckArrival
            RaiseEvent newTruckArrived(myEngine, newTruck)
        Case eventType.LoadingCompletion
            RaiseEvent loadingCompleted(myEngine)
    End Select
End Function
```


Coding simulation models

33

5- Start Coding

3) Initialization (it is run in the main body)

```
Public Sub initializeSim(ByRef myEngine As Engine, ByRef myPlant As AsphaltPlant)
    Dim myFirstFELItem As New FELItem
    myFirstFELItem.entity.truckId = 1
    myFirstFELItem.eventTime = myPlant.nextTruckArrival
    myFirstFELItem.simEvent = eventType.TruckArrival
    myEngine.scheduleNewEvent(myFirstFELItem)
End Sub
```

Coding simulation models

34

5- Start Coding

4) Main simulation engine procedure (in the main body)

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button1.Click
```

```
    'Main loop of simulation
```

```
    Dim maxSimTime As Double = Val(TextBox5.Text)
```

```
    Dim myTruck As New TruckEntity
```

```
    Dim myPlant As New AsphaltPlant
```

```
    Dim myEngine As New Engine
```

```
    Dim myFELItem As New FELItem
```

```
    'Initialize simulation by scheduling first truck arrival
```

```
    initializeSim(myEngine, myPlant)
```

Coding simulation models

35

5- Start Coding

4) Main simulation loop (in the main body)

While myEngine.curTime < maxSimTime

 myFELItem = myEngine.nextEvent()

 myPlant.eventHappened(myEngine, myFELItem.entity, myFELItem.simEvent)

 txtCurTime.Text = myEngine.curTime.ToString

 txtAvgQL.Text = myPlant.avgQLength.ToString

 txtAvgWaitT.Text = myPlant.avgQWaitingTime.ToString

 txtUtilization.Text = myPlant.avgUtilization.ToString

 txtTruckServed.Text = myPlant.totalTruckServred.ToString

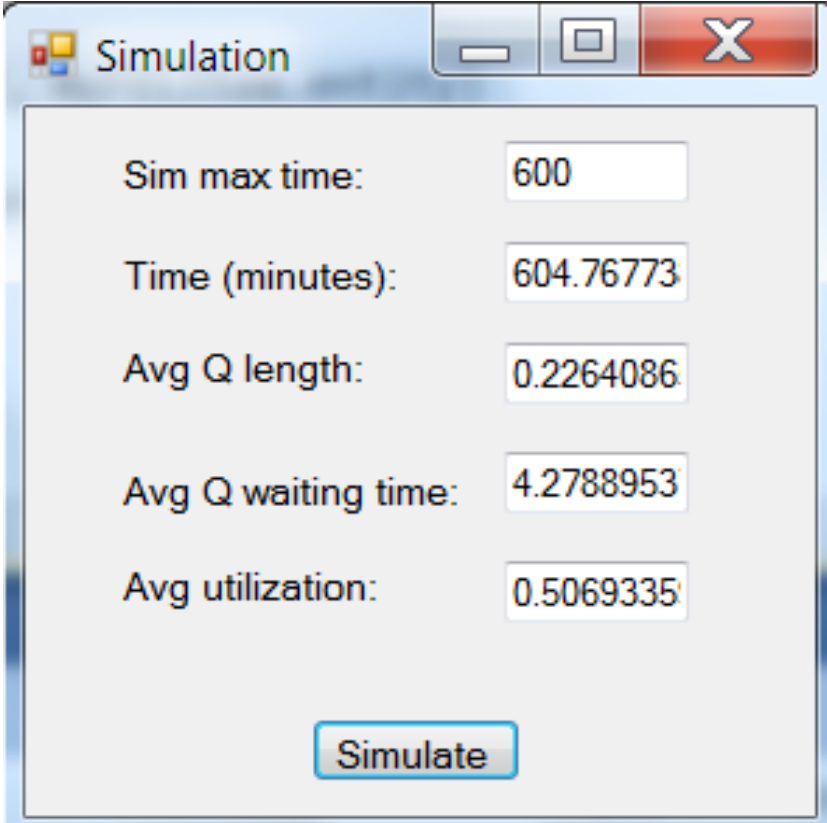
 Me.Refresh()

End While

Coding simulation models

36

5- Simulation run sample



A screenshot of a software window titled "Simulation". The window contains five input/output fields and a "Simulate" button. The fields are arranged in a list, each with a label and a corresponding text box. The "Sim max time" field contains the value "600". The "Time (minutes)" field contains the value "604.76773". The "Avg Q length" field contains the value "0.2264086". The "Avg Q waiting time" field contains the value "4.2788953". The "Avg utilization" field contains the value "0.5069335". The "Simulate" button is located at the bottom center of the window.

Parameter	Value
Sim max time:	600
Time (minutes):	604.76773
Avg Q length:	0.2264086
Avg Q waiting time:	4.2788953
Avg utilization:	0.5069335

Simulate

Home assignment 7

37



Suppose our earthmoving example with 4 trucks and 4 loaders in the system with working hours from 7 am to 7 pm. There is only one dumping site. Different operation activities have following durations:

- Loading: Normal(12 minutes, 1.5 minutes)
- Trip to dumping site: Triangular(3 minutes, 4 minutes, 6 minutes)
- Dumping: Triangular (5 minutes, 6 minutes, 10 minutes)
- Trip from dumping: Uniform (3 minutes, 4 minutes)

Time of each truck arrival at the morning has a Triangular distribution with minimum of 6:55 am, most likely of 7:05 am and maximum of 7:15 am.

Develop codes required for modeling the problem. You are free to use any programming language you are familiar with.

(Due in 2 weeks)



Thank you!