**Depth Finder Device  
Assignment 1**

By : Ronnie Farrell

Student # : 100717103

To : Prof. Wainer

Course : SYSC 5104

Date : October 25, 2011

**Part I**

This system represents a device that sinks to the bottom of a body of water and finds the depth of the water. After it has recorded the depth, it inflates a bladder with air to raise the device back to the surface. The device is activated with a button press. The device uses a coupled model Con\_DS comprised of atomic models controller (Con) and depth sensor (DS), connected to another atomic model inflator. The device uses a controller to poll the current depth sensor every 3 seconds. The controller compares the current depth with the previous depth recorded, and when the current depth equals the previous depth it knows that the device has hit the ground and sends a signal to inflate with the overall depth.

**Model Structure:**

**Components**

* Con\_DS (Coupled Model)
  + Controller (Atomic Model)
    - Takes input from a button press to activate it and sends output to the depth sensor to determine the current depth
    - Once the depth sensor sends input back to the controller, it either sets the current depth or compares the current depth to the previous depth and waits 3 seconds
    - If the previous depth equals the current depth, the device has hit the ground, so activate the inflator, and send the current depth to output
    - If the previous depth is not equal to the current depth, device is still sinking, so re-poll the depth sensor
  + Depth Sensor (Atomic Model)
    - Receives input from the controller telling it to determine the depth, after determining the depth, it sends an output with the depth
* Inflator (Atomic Model)
  + Receives input from the controller and inflates a bladder with air so the device can become positively buoyant and rise to the surface
  + After the device is finished inflating the bladder, it sends an output stating it is finished

**Part II**

The models were organized as atomic / coupled models, where the structure and coupling schemes were defined in Part I. There was no change to the conceptual model. To eliminate redundancy refer to Part I for the structure and coupling schemes of the conceptual model.

**DEVS Formal Specification for the Coupled Models:**

Device= < X, Y, {Con\_DS, Inflator}, EIC, EOC, IC, SELECT >

* X = {in1, in2}
* Y = {out1, out2}
* EIC = { (Device.in1, Con\_DS.in1), (Device.in2, Con\_DS.in2)}
* EOC = { (Inflator.out , Device.out2), (Con\_DS.out1, Device.out1) }
* IC = { (Con\_DS.out2, Inflator.in) }
* SELECT : ({Con\_DS, Inflator}) = Con\_DS

Con\_DS = < X, Y, {Con, DS}, EIC, EOC, IC, SELECT >

* X = {in1, in2}
* Y = {out1, out2}
* EIC = { (Con\_DS.in1, Con.in1), (Con\_DS.in2, DS.in2)}
* EOC = { (Con.out3, Con\_DS.out1), (Con.out2, Con\_DS.out2) }
* IC = {Con.out1, DS.in1), (DS.out, Con.in2)}
* SELECT : ({Con, DS}) = Con

**DEVS Formal Specification for the Atomic Models:**

Inflator = < S, X, Y, δint, δext, λ, ta >

* X = {in }
* Y = {out}
* S = {A, B}
* δint (B) = A
* δext (in==1, A) = B
* ta (A) = INFINITY
* ta (B) = 2 sec
* λ (B) = out

DS = < S, X, Y, δint, δext, λ, ta >

* X = {in1, in2}
* Y = {out}
* S = {A, B, C, D}
* δint (B) = C
* δint (D) = A
* δext (in1==1, A) = B
* δext (in2==depth, C) = D
* ta (A) = INFINITY
* ta (B) = 10 ms
* ta (C) = INFINITY
* ta (D) = 10 ms
* λ (D) = (out=depth)

Con = < S, X, Y, δint, δext, λ, ta >

* X = {in1, in2}
* Y = {out1, out2}
* S = {A, B, C, D}
* δint (B) = C
* δint (D) = if (prev\_depth != cur\_depth) C else A
* δext (in1, A) = B
* δext (in2, C) = D
* ta (A) = INFINITY
* ta (B) = 10 ms
* ta (C) = INFINITY
* ta (D) = 3 sec
* λ (B) = out1
* λ (D) = if (prev\_depth != cur\_depth) (out1=1) else {out2=1, out3=depth}

**Testing Strategies**

All of the models will be coupled into a top layer so that testing can be performed. Testing will be completed in an incremental expansion of the tests, meaning that the tests will be performed on the individual atomic models first, then expanding the hierarchy and testing those models until the overall model is to be tested. The order of my tests will be performed in the following order with the following tests:

* Inflator (Atomic)
  + Create a coupled model file Inflator.ma with proper connections
  + Create an Inflator.ev file with the following inputs:
    - 00:00:03:00 in 0
    - 00:00:05:00 in 5
    - 00:00:08:00 in 1
  + If the output file displays out = 1 at 10 sec, the model works
* DS (Atomic)
  + Create a coupled model file DepthSensor.ma with proper connections
  + Create a DepthSensor.ev file with the following inputs:
    - 00:00:02:00 in1 0 // incorrect input
    - 00:00:02:00 in2 30
    - 00:00:05:00 in1 1 // poll depth sensor
    - 00:00:06:00 in2 50 // input current depth (ft)
    - 00:00:08:00 in1 1 // poll depth sensor
    - 00:00:08:11 in2 63
    - 00:00:10:00 in11
    - 00:00:11:00 in2 70
    - 00:00:14:00 in11
    - 00:00:15:00 in2 99.6
  + If the output file displays depths at 50 ft, 63ft, 70 ft, and 99ft the model works
* Con (Atomic)
  + Create a coupled model file Controller.ma with proper connections
  + Create a Controller.ev file with the following inputs:
    - 00:00:02:00 in1 0 // incorrect input
    - 00:00:05:00 in1 1 // activate controller
    - 00:00:06:00 in2 50 // receives depth of 50 ft from DS
    - 00:00:10:00 in2 60
    - 00:00:14:00 in2 70
    - 00:00:18:00 in2 80.53
    - 00:00:22:00 in2 88
    - 00:00:26:00 in2 88 // device has hit the ground
  + If the output file displays out2 = 1 and out3 = 88ft, the model is correct
* Con\_DS (Coupled)
  + Create a coupled model file Con\_DS.ma with proper connections
  + Create a Con\_DS.ev file with the following inputs:
    - 00:00:02:00 in1 0 // incorrect input
    - 00:00:05:00 in1 1 // activate device
    - 00:00:06:00 in2 50 // receives depth of 50 ft from DS
    - 00:00:10:00 in2 60
    - 00:00:14:00 in2 70
    - 00:00:18:00 in2 76.8
    - 00:00:22:00 in2 76.8 // device has hit the ground
  + If the output file displays out2 = 1 and out1 = 76ft, the model is correct
* Device (Complete model)
  + Create a coupled model file Device.ma with proper connections
  + Create an Device.ev file with the following inputs:
    - 00:00:03:00 in1 4 // incorrect input
    - 00:00:05:00 in1 1 // activate device
    - 00:00:06:00 in2 50 // receives depth of 50 ft from DS
    - 00:00:10:00 in2 60.25
    - 00:00:14:00 in2 70.8
    - 00:00:18:00 in2 71
    - 00:00:22:00 in2 71 // device has hit the ground
    - 00:00:26:00 in2 71
  + If the output file displays out1 = 71ft and out2 = 1, the model is correct

**Part III**

From the specifications defined in Part II, all of the atomic and coupled models were built using CD++. In order to verify the correctness of each of the models, test cases defined in Part II were created in order to test these models.

Note: For all of the models, a .bat (ie. Inflator.bat) file was created to make running the simulator easier. This file runs the simulator with the event file (name.ev), and creates an output file (nameOUT.out) with the outputs and a log file (nameLOG.log) with a log of all the activity.

**Tests and Analysis**

**Inflator Test**

**Expected Behaviour:**  
In order for the inflator model to perform correctly, it should output a value equal to 1 two seconds after it received an input value of 1 signifying that it is finished inflating the bladder.

**Actual Behaviour:**In order to test this functionality, an event file named Inflator.ev was created. This event file contains the times, input ports and values sent to the model. Refer to Part II for the actual values used. These inputs test incorrect values sent to the port as well as the correct values. The only valid input is sent to the model at 8 seconds. The following is what is received in the output file (InflatorOUT.out):

00:00:10:000 out 1

Since there were three inputs sent to the model, but only one being correct, there was only one output with the appropriate value. This is what is expected, as the first two inputs were disregarded. It can be concluded that this model functions correctly.

**Depth Sensor Test**

**Expected Behaviour:**In order for the depth sensor model to perform correctly, it should receive a value of 1 to port in1 telling the sensor to get the depth from port in2. After it receives the activation signal, it waits to receive the depth in feet from port in2. Once the depth is received, it sends an output with a value of the depth to the port out, and transitions to wait for another activation signal.

**Actual Behaviour:**In order to test this functionality, an event file named DepthSensor.ev was created. This event file contains the time, input ports, and values sent to the model. Refer to Part II for the actual values used. These inputs test incorrect and correct values sent to the port. The following is what is received in the output file (DepthSensorOUT.out):

00:00:06:010 out 50

00:00:08:021 out 63

00:00:11:010 out 70

00:00:15:010 out 99

The first activation input sent to the model is incorrect, and that is why the output file does not display an output of 30 at 3.01sec. Since there were three correct activation signals sent, with three depths, and three output values with the correct depths, the model is as expected. The last input sent a depth of 99.6, but since this sensor rounds down to the nearest foot, it displays an output of 99ft, as expected. It can be concluded from these tests that this model functions properly.

**Controller Test**

**Expected Behaviour:**In order for the controller model to perform correctly, it should receive a value of 1 to port in1 activating it resulting in an output of value 1 to port out1 (activating the depth sensor). Once activated it waits for inputs of depths on port in2. When it receives the depths it waits 3 seconds before asking for another depth. It compares the current depth with the previous depth. If the previous depth is equal to the current depth it knows that the device is at the ground, so it sends an output with value of 1 to port out2 (activating inflator), and an output with the depth to port out3. If the previous and current depths are different, it sends an output with value 1 to port out1 (re-polling the depth sensor).

**Actual Behaviour:**  
In order to test this functionality, an event file named Controller.ev was created. This event file contains the time, input ports, and values sent to the model. Refer to Part II for the actual values used. These inputs test incorrect and correct values sent to the port. As mentioned, the device is supposed to re-poll every 3 seconds, which it does, but it also has other small delays, this is why depths are sent to port in2 every 4 seconds and not 3 seconds. The following is what is received in the output file (ControllerOUT.out):

00:00:05:010 out1 1

00:00:09:000 out1 1

00:00:13:000 out1 1

00:00:17:000 out1 1

00:00:21:000 out1 1

00:00:25:000 out1 1

00:00:29:000 out2 1

00:00:29:000 out3 88

The first activation input sent to the model is incorrect, and that is why the output file does not display an output of 1 at 2.01sec. The rest of the outputs are as expected. The depths keep increasing, and this is why we see outputs with value of 1 to port out1 (activating depth sensor). Finally, once it receives two depths that are they same (the device is on the ground), it sends an output with value 1 to port out2 (activating inflator), and an output to port out3 with the depth. This is as expected, and can be concluded that the model functions correctly.

**Con\_DS Test**

**Expected Behaviour:**In order for the Con\_DS model to perform correctly, it should receive a value of 1 to port in1 activating the controller resulting in an output of value 1 to port out1 of the controller activating the depth sensor. Once the depth sensor is activated it waits for inputs of depths on port in2. When it receives the depth it sends the depth back to the controller. The controller waits 3 seconds before asking for another depth. It compares the current depth with the previous depth. If the previous depth is equal to the current depth it knows that the device is at the ground, so the controller sends an output with value of 1 to port out2 (activating inflator), and an output with the depth to port out1. If the previous and current depths are different, the controller re-polls the depth sensor.

**Actual Behaviour:**In order to test this functionality, an event file named Con\_DS.ev was created. This event file contains the time, input ports, and values sent to the model. Refer to Part II for the actual values used. These inputs test incorrect and correct values sent to the port. As mentioned, the device is supposed to re-poll every 3 seconds, which it does, but it also has other small delays, this is why depths are sent to port in2 every 4 seconds and not 3 seconds. The following is what is received in the output file (Con\_DS\_OUT.out):

00:00:25:010 out2 1

00:00:25:010 out1 76

The outputs are as expected. The input file simulated that the device hit the ground at 76.8ft, but since the depth sensor rounds down to the nearest foot, it displays an output of 76 ft on port out1. It also sent an output to port out2 with a value of 1 telling the inflator to turn on. This is as expected, and can be concluded that the model functions correctly.

**Device Test**

**Expected Behaviour:**This is the highest level of abstraction, and represents the whole device. In order for the device model to perform correctly, it should receive a value of 1 to port in1 activating the Con\_DS component. Once the Con\_DS component has determined that it has hit the ground it sends an output with the depth to port out1, and activates the inflator. After the inflator has finished inflating, it should send an output with value 1 to port out2 signalling that it has finished inflating.

**Actual Behaviour:**In order to test this functionality, an event file named Device.ev was created. This event file contains the time, input ports, and values sent to the model. Refer to Part II for the actual values used. These inputs test incorrect and correct values sent to the port. As mentioned, the device is supposed to re-poll every 3 seconds, which it does, but it also has other small delays, this is why depths are sent to port in2 every 4 seconds and not 3 seconds. The following is what is received in the output file (DeviceOUT.out):

00:00:25:010 out1 71

00:00:27:010 out2 1

The outputs are as expected. The input file simulated that the device hit the ground at 71ft which it shows in the output file on port out1. The output file also shows that the inflator finished inflating the bladder 2 seconds after the depth was sent to the output. This is as expected, and can be concluded that the overall model functions correctly.

**Conclusion**

This model was tested using an incremental expansion approach. Each component was simulated with several different tests in order to verify the correctness of the model. As shown throughout this report the overall device and all of the components function as they are expected, and it can be concluded that this simulation was a success.