**A DEVS Model and Cadmium Implementation of a Forestry Probe Power System**

**Assignment 1**

**SYSC 5104**

**Author: Sebastien Cook**

**101022643**

**Date Submitted: October 25 2020**

**Objective:**

The objective of this assignment is to generate and simulate a DEVS model of a forestry probe’s power system. The DEVS model will output the forestry probe’s battery capacitance based on a select solar panel’s charging capabilities and the forestry probe’s energy needs.

**Background:**

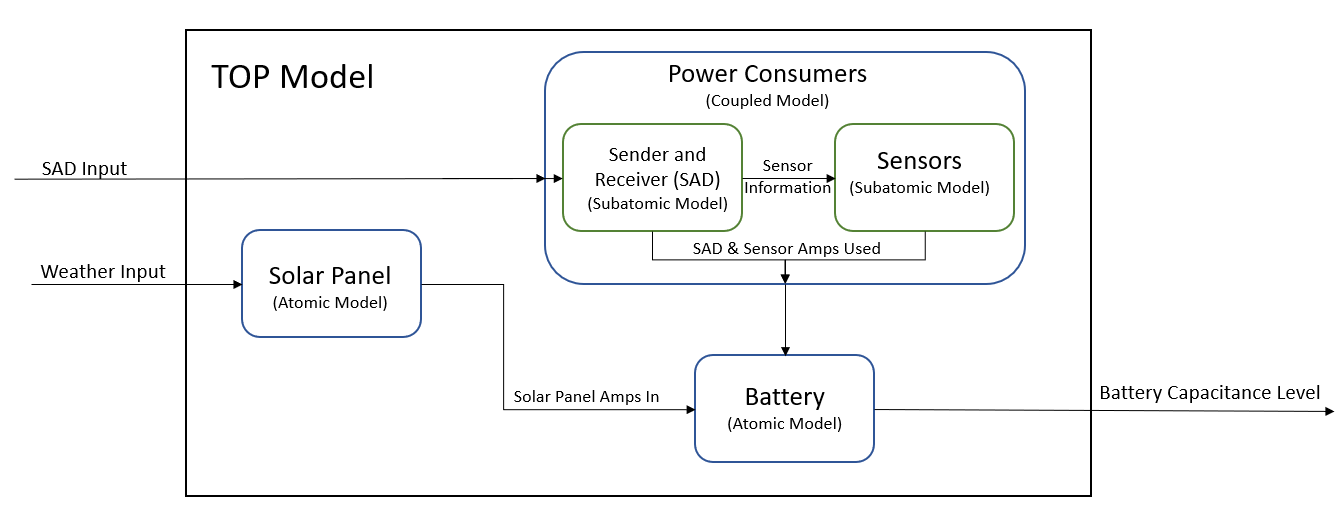
Forestry environmental science research often involves long hours of grueling work for front line researchers who must manually accumulate the data for their experiments. More specifically, people must be sent to the location of interest, which can be a taxing journey, and once there, they then have to prepare the experimental perimeter before the first set of data can even be taken. However, this is not the end of the experiment; days, weeks or even years later, the researchers must return to the experimental location to acquire a new set of data that can be compared to the previous sample. Thus, a very large amount of manpower and funding is required for a relatively small number of data points.

However, this problem can be partially mitigated by deploying forestry probes across the experimental perimeter that will periodically accumulate data, sending them directly to a selected database. While this may be an elegant solution, the forestry probe must have a long lasting or self-charging power system for it to be sustainable in the long term.

1. **Model Description**

As stated in the objective, the model will output the battery’s capacitance based on a selected solar panel and the forestry probe’s energy needs. This is, however, a simplified image of the real system. There are numerous internal and external factors that will affect the solar panel’s efficiency, the sensor’s energy consumption, and hence the battery’s capacitance. For this reason, the DEVS model presented in Figure 1 separates the forestry probe’s components into separate atomic models that can be affected by external and internal changes. An in-depth description and formal descriptions of each models can be found in section 2.0 of this document.

The general DEVS model in Figure 1 is a two-level model comprised of 4 different atomic models, two of which are subatomic models that constitute a coupled model. The atomic model “Solar Panel” models a typical 100W solar panel that is slave to the weather conditions. The subatomic model “SenderReceiver” (SAD) models the energy consumption of the forestry probe’s transmitter and receiver. The subatomic model “Sensors” models the energy consumption currently sampling sensors based on each individual sensor’s settings. The two subatomic models “SAD” and “Sensors” constitute the coupled model “Power Consumers” which encompasses and models the forestry probe systems that require and consume energy. Finally, the atomic model “Battery” models the capacitance of a typical 12V 100-ampH battery that the forestry probe is currently designed to use. “Battery” receives the amps produced by “Solar Panel” and the amps drawn by “Power Consumers” every hour. “Battery” then outputs its capacitance level when affected.

**Figure 1**

The combination of the two atomic models “Solar Panel” and “Battery” along with the coupled model “Power Consumers” establishes the DEVS top model of the forestry probe’s power system. This model requires three data structures for reliable information passing and input.

**Weather Information**

Weather information is a simple data structure that is comprised of a temperature and a visibility value. The three factors that determine a solar panel’s efficiency is the temperature, visibility, and the time of day. The temperature is important since, as described in the selected solar panel’s specification, if the solar panel is functioning in temperatures above 25oC, it will experience a decrease of 0.55% ampH for every degree above 25 oC. Furthermore, visibility is important since it determines how much effective light can potentially reach the solar panel. Finally, the time of day is important since a solar panel can only produce power during the day when sunlight is present.

The data structure only has the temperature and visibility values since the “Solar Panel” atomic model has a state value that keeps track of the time of day throughout the simulation.

**SAD Information**

SAD information is the data structure used to pack and deliver the information for both the receiver and sensor settings. The “Power Consumers” model was structured with the “SAD” model, receiving input from the outside and passing this information along to the sensors to better model the flow of the real system. The SAD information stat structure is as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Transmitter transmission frequency  (times per week) | Sensor number | Frequency of sampling (Hours) | Amps used when sampling |
| 2 | 2 | 48 | 40mA |
| 0 (ignore 0 setting) | 3 | 1 | 35mA |
| … | … | … | … |

**Sensor Information**

Sensor information is very similar to the SAD information data structure. It is comprised of the same information and structure minus the transmitter transmission frequency. The data structure provides a structured and reliable way of sending the individual sensor settings from the Sender and Receiver model to the Sensors model.

1. **DEVS Model Formalism**

In the following section, each atomic model will be formally described using the DEVS formalism technique. This will offer a deeper understanding of the model’s individual models and innerworkings. First, each atomic model’s formal description will be presented, followed by the coupled models.

* 1. **Solar Panel**

The “Solar Panel” atomic model charges the battery based on its current efficiency. Without input, the model would call its internal function every hour, checking the time of day and calculating the hour’s efficiency based on current temperature and visibility states, and then it would output the amps produced in the last hour. If the “Solar Panel” state variable “Day” is true, then state variable “Efficiency” would be set to 100%, and then reduced based on the states “Temp” and “Visibility”. However, at night, “Efficiency” is set to 0% since a solar panel doesn’t produce power at night. After calculating “Efficiency” in the internal function, 8.33 \* Efficiency is outputted. When a weather input is received, the temperature and visibility states of “Solar Panel” are modified, changing the efficiency of the solar panel until a new weather input arrives.

Solar Panel = <X, Y, S, δint , δext, λ, ta>

X = {Weather information(Temp, Visibility)};

Y = { ampsIn };

S = {Efficiency E {0 <= ef <= 100},

Day E {true, false},

Temp E {-infinity < Temp < infinity},

Visibility E {0 <= vis <= 16},

Time E {0 <= Time < 24}}; //Keeps track of the hour of day

δext(S, e, X){ //Weather input changes the solar panel’s temp and visibility states

Temp = X.Temp;

Visibility = X.Visibility;

}

λ(S){

8.33 \* efficiency -> sent onto the output channel; //8.33 is the ampH at peak efficiency

}

ta(S){

next internal = TIME( 1 hour); //Sends the amps to the battery every hour

}

δint(S){

%Efficiency = 1; //100% efficiency

Time++; //Internal function happens every hour

if(Time == 24){ //If it is 12pm, restart clock at 0

Time = 0;

}

if(8 <= Time <= 16) //On average, the sun in Banff is up at 8am and sets at 5pm

day = true;

else

day = false;

if(day){ //If day, calculate the efficiency of that hour

%Efficiency = visibility/16 //16km is max visibility in the trace

if(Temp <= 25) //Efficiency will not be effected by temp when under 25

Efficiency = %Efficiency;

else if(25 < Temp <= 85) //Every degree above 25 reduces efficiency by 0.55%

Efficiency = %Efficiency - (Temp - 25)\*0.55%;

else

Efficiency = 0; //Too hot for the solar panel to function

if(Efficiency < 0) //Make sure there is no negative percentage

Efficiency = 0;

}

else //It is nighttime; solar panel cannot produce power

day = false;

}

* 1. **Battery**

“Battery” is a passive model that is affected by all of the other atomic models. It will remain passive until it receives an input from the “Solar Panel” or “Power Consumers” models. When an input is received, “Battery” will measure its new capacitance charge based on the amps in and the amps out. After calculating its new charge state, it will output its current capacitance since the battery’s capacitance is what is of interest in this simulation. Because “Solar Panel” and “Sensors” are scheduled to send their usage to “Battery” every hour, the battery’s capacitance is then measured and outputted.

Battery = <X, Y, S, δinternal , δexternal, λ, ta>

X = { ampsIn, ampsOut};

Y = { capacitance };

S = { charge E {0 < charge < 100}};

δinternal(S){ //there is not internal statefor this passive model }

δint(S){ //there is no internal state for this passive model }

δext(S, e, X){

charge += ampsIn - ampsOut;

if(charge > 100) //Cannot over charge

charge = 100;

if(charge < 0) //Cannot go below 0 charge

charge = 0;

}

λ(S){

send the current charge onto the output link

}

ta(S){

next internal = infinity;

}

* 1. **Sender and Receiver**

SAD models the forestry probe’s transmitter and receiver power consumption. SAD is one of the two sub models of the Power Consumer model. It receives, as an input settings for the transmitter transmission, frequency and sensor settings to be passed on to the “Sensors” atomic model. The SAD model without input, would transmit and receive by default once a week on Sunday. However, when an input is received, the transmitter’s sending frequency is modified, and the new sensor settings are sent to the “Sensors” model. SAD will ignore any input if it’s not in the receiving state. If the model is in the transmitting or receiving state, it outputs its usage every hour. Otherwise, the usage will be outputted daily.

Sender & Receiver = <X, Y, S, δint , δext, λ, ta>

X = { SAD information(transmitter frequency, sensor number, sensor sample frequency, sensor amps)};

Y = { amps Out, Sensor information(sensor number, sensor sample frequency, sensor amps)};

S = { Receiving E {true, false},

Transmitting E {true. false},

new Data E {true, false}, //If there are new “Sensor” settings to transmit

Transmitter Frequency E {0 < tF <= 7}, //Must transmit at least once a week

Transmitting Day E {0 < tD <= 7}, //Next day scheduled to transmit data

day E {0 < day <= 7}, //Day of the week

hour E {0 <= hour < 24}, //Keeps track of the hour of day when transmitting or receiving

Sensor Information E {(sensor number, sensor sample frequency, sensor amps)}};

}

δinternal(S){ //there is not internal statefor this passive model }

δext(S, e, X){

if(X.transmitting frequency != 0){ //Must transmit at least once a week. 0 means ignore this new setting

transmitting frequency = X.transmitting frequency;

}

Sensor Information = X.sensor information;

new Data = true;

}

λ(S){

int ampsOut = 0; //Local variable

if(transmitting)

ampsOut += transmitting hourly consumption;

if(receiving)

ampsOut += receiving hourly consumption;

send ampsOut on output link;

if(new Data)

send new data on output link connected to sensor;

}

ta(S){

if(!state.receiving && !state.sending)

next\_internal = TIME("00:00:24:000");

else

next\_internal = TIME("00:00:01:000"); //every hour while sending or receiving

}

δint(S){

if(!receiving && !transmitting) //Register that a day has passed and skip the rest

day++;

else{ //If transmitting or receiving, the model sends hourly amps used

hour++;

if(hour == 24) //the day has passed

hour = 0; day++;

transmitting = false; receiving = false; new Data = false;

}

if(day == 7) //Receiver receives once per week by default

receiving = true;

if(day == transmitting day)

transmitting = true;

transmitting day = next transmitting day based on frequency;

}

}

* 1. **Sensors**

“Sensors” is the second subatomic model in the “Power Consumers” coupled model. “Sensors” models the power consumption of up to 10 different sensors on the forestry probe. Each sensor has their own frequency and ampH, allowing for the simulation of multiple different sensors kits that the forestry probe could be deployed with. Furthermore, the individual sensor settings can be changed mid-simulation if, for example, a sensor is only meant to sample data during the summer. Every hour, the “Sensors” internal function will reduce each sensor’s time until next sampling by one hour, and output the amps used by all of the sensors that sample during that hour. Any input will simply change the selected sensor’s operation setting.

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

X = { Sensor information(sensor number, sensor sample frequency, sensor amps)};

Y = { amps Out };

S = { Frequency[max sensors] E {0 < F < infinity}, //Frequency of sampling in hours

amps[max sensors] E {0 < amps < infinity},

Twait[max sensors] E {0 < Tw < infinity}}; //Time left before next sampling

δint (S){

for(int i=0; i < maxSensors; i++){ //Reduce the time of next sample by one hour

Twait[i] -= 1;

if(Twait[i] == -1)

Twait[i] = state.frequency[i];

}

}

δext (S, e, X){ //Receive new settings for a select sensor

int numSensor = X.sensor information -> sensor number

Frequency[numSensor] = X.sensor information -> frequency;

amps[numSensor] = X.sensor information -> amps

}

λ(S){

int ampsOut = 0;

for(int i=0; i < maxSensors; i++){ //Take the amps out of all sensors that are sampling at that time

if(Twait[i] == 0)

ampsOut += state.amps[i];

}

}

ta(S){

next\_internal = TIME("00:00:01:000"); //Every hour

}

* 1. **Power Consumers**

“Power Consumers” is a coupled model comprised of the “SAD”, and “Sensors” subatomic models. “Power Consumers” models the elements of the forestry probe that require and consume power. As an input, “Power Consumers” receives SAD information; it receives the transmitter’s sending frequency and sensor settings. It then, based on the subatomic models’ behaviors, outputs the total amps pulled every hour. Its output link is meant to connect to and update the “Battery” model.

Power Consumer = <X, Y, D, {Mi}, IC, EIC, EOC, select>

X = { SAD information };

Y = { ampsOut, SAD information};

D = { Sender and Receiver, Sensors };

Mi = { Msad, Msensor };

IC = { SAD sensor information output -> Sensor information input; };

EIC = {

Power consumer input -> SAD input;

};

EOC = {

SAD ampsOut -> power consumer ampsOut;

Sensor ampsOut -> power consumer amosOut;

};

* 1. **TOP model**

The TOP model is the combination of all of the previously mentioned models. It is comprised of the “Solar Panel”, “Battery” and “Power Consumers” models. The “Solar Panel” output is connected to the “Battery” ampsIn input, sending “Battery” its hourly charging rate. The “Power Consumers” output is connected to the “Battery” ampsOut input, sending the hourly amps pulled. Finally, “Battery” is connected to the TOP model’s output, sending its current capacitance every hour. The TOP model receives both weather information and SAD information as input. The input of weather allows for the simulation of many different environments and their weather patterns. The SAD information allows for varying simulation outcomes simply by changing the “Power Consumers” settings.

TOP Model = <X, Y, D, {Mi}, IC, EIC, EOC, select>

X = { weather information , SAD information };

Y = { battery capacitance };

D = {

Solar Panel,S

Battery,

Power Consumers

};

Mi = { Msp, Mb, Mpc };

IC = {

Solar Panel ampsIn output -> Battery ampsIn input;

Power Consumers ampsOut output -> Battery ampsOut input;

};

EIC = {

TOP SAD information -> Power consumer;

TOP weather -> SolarPanel input;

};

EOC = {

Battery capacitance -> TOP battery capacitance;

};

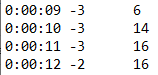
1. **Experimental Framework and Testing Results**

The overall objective is to model a forestry probe’s power system and to simulate its battery capacitance given a solar panel for charging and a set of power consumers. The specific objective within this experimental framework however, is to first validate each atomic and coupled model, and then to test whether the probe ever reaches a battery charge of 0 if it operates in Prince George, British Colombia. This section is structured as follows: (1) each individual atomic validation test and results, (2) “Power Consumers” coupled model validation test and results, and (3) the TOP model’s validation and experimental test and results.

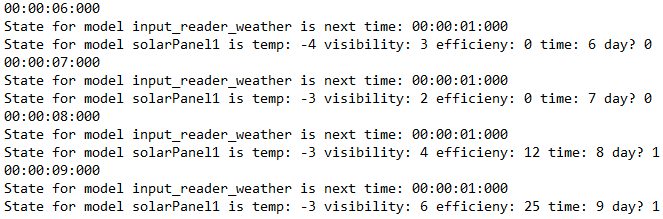
To ensure consistent and comparable results across each atomic and coupled model, each test was done over the course of a year. Because the internal clock’s maximum in Cadmium is 99 hours, and the simulation iterates through 8,760 hours, seconds are used as the time intervals to represent 1 hour within a year. This means that the simulation’s elapsed time is 02:25:59:000, which is equivalent to 8,760 seconds.

* 1. **Solar Panel Test**

“Solar Panel” was tested using the weather history from the town of Prince George, British Colombia in 2019 as reported on the Government of Canada’s climate website[1]. The hourly values for temperature and visibility were put into a trace file. The trace file contains 8760 lines, amounting to a year of operation.

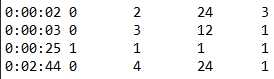


In a separate text files, the states and output of “Solar Panel” were recorded. The states are important for the validation of the model. The first state of interest was the state variable “Day”. It is vital that “Solar Panel” correctly maintains the time of the day since it directly effects the efficiency. It can be seen in the state file below, that at time 6am and 7am, day is false, but at 8am and 9am it is true. Furthermore, the state of efficiency changes from 0 at 7am to a non-0 value at 8am, confirming that during the night, efficiency is automatically set to 0. The next state of interest is efficiency. Between the times of 8am and 9am, the visibility increases by 2km and, as such, the efficiency is raised. This analysis of the state variables confirms that “Solar Panel” functions correctly.



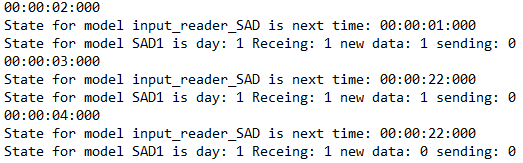
* 1. **Sender and Receiver Test**

“SAD” was tested by inputting a SAD information text file trace, which scheduled inputs at both receiving and non-receiving times to confirm that the SAD model correctly received settings only when in receiving mode.



The initial “SAD” condition has it receiving the first day it is deployed to receive initial settings, and then returns to its standard weekly cycle of receiving. In the trace above, initial settings are sent to “SAD” followed by new settings sent at time 00:00:25. This setting instruction is meant to be ignored since it sends just after “SAD” exits the receiving state.

As seen in the state trace below, at time 00:00:02 and 00:00:03, “SAD” receiving state is true, and “new data” is true, which signals that the new settings have been successfully received. Furthermore, at time 00:00:25, “SAD” is no longer receiving, and thus, despite being sent new settings, “new data” is false, meaning the input was correctly ignored.

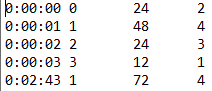


**…**



* 1. **Sensors Test**

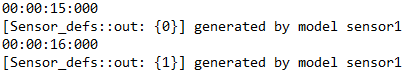
“Sensors” was tested using the same sensor settings given in the “SAD” input trace. This was done since the isolated “Sensors” and “SAD” test results can then be combined and compared with the “Power Consumers” test outputs to validate the coupled model’s functioning. As it can be seen in the “Sensors” input trace file, the instruction at time 00:00:25 was not, assuming correct functioning of the “SAD” atomic model, recorded.



Since the “Sensors” state variables are arrays to allow multiple sensors to be modeled within the same atomic model, the output text file was analyzed to confirm correct functioning. At time 00:00:03, “Sensors” receives a set of settings for sensor 4, which is scheduled to sample every 12 hours and consume 1 ampH while sampling. Since the setting is received at time 00:00:03, the countdown until sampling will begin at time 00:00:04, and exactly 12 hours later at time 00:0016, 1 ampH is drawn as shown below. This confirms that “Sensors” time tracker functions properly, and that each sensor’s settings are correctly saved.



**…**



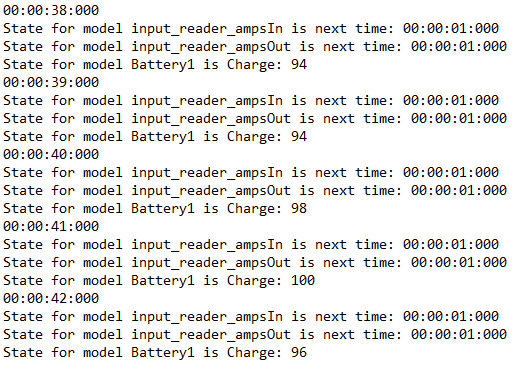
* 1. **Battery Test**

“Battery” was tested using two input files: ampsIn and ampsOut. These files were used to replace “Solar Panel” and “Power Consumers” for an isolated test. Random values for ampsIn and ampsOut were generated for every hour. Actual values from “Solar Panel” and “Power Consumers” were not required for this test since validating the output simply required subtracting the input files, ampsIn and ampsOut, at every hour and comparing it with the “Battery” charge state.

ampsIn ampsOut

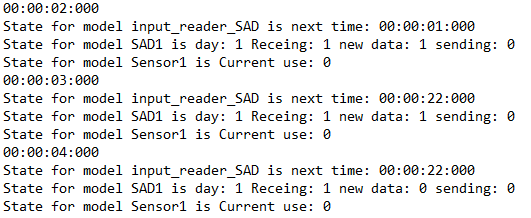
 

The state trace below demonstrates two important aspects. First, at time 00:00:39, the same amount of ampH are supplied and taken from “Battery”. Thus, between time 00:00:38 and 00:00:39,the charge state remains the same. This confirms that the inputs are correctly processed. Secondly, the charge supplied at time 00:00:41 would set the “Battery” charge above 100 which is not possible. Looking at the charge state at time 00:00:41 as shown below, the charge is at 100 rather than 102, confirming that the boundary condition of the battery charge state functions properly.

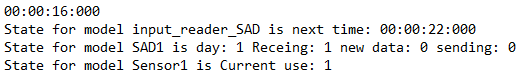


* 1. **Power Consumers Test**

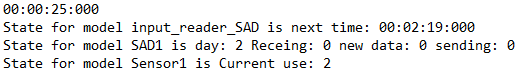
“Power Consumers” was tested using the same inputs as the “SAD” test. As mentioned previously, this permits the comparison between the “SAD” and “Sensors” test results against the “Power Consumers” results to confirm functioning. Given the same inputs, the “Power Consumers” state trace was as follows:



**…**



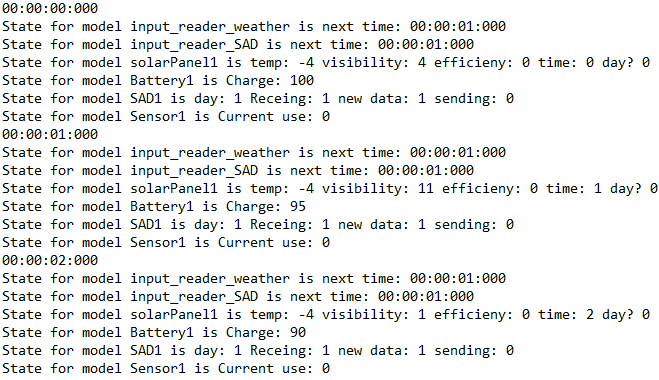
**…**



By analyzing the above trace, three key aspects can be identified that confirm correct functioning. The first is that the “Power Consumers” states are comprised of both “SAD” and “Sensors” states. The second key aspect is revealed by comparing the “SAD” state variables to the isolated “SAD” test. At both time 00:00:03 and 00:00:25, the states are identical. Finally, the third is shown by comparing the “Sensors” states to the isolated “Sensors” test states. Once again, the states are identical, sampling at time 00:00:16 after receiving the new setting at time 00:00:03.

* 1. **TOP Model Test**

The TOP model was tested using the same weather and SAD input files that were employed in the individual tests. Again, this was intentional, as it fulfilled two tasks. Since the weather trace in the individual test was that of the year’s weather history in Prince George, British Colombia, the test served as both a verification and as an experimental test – for verification, the TOP model’s state trace was compared to each of the individual tests to confirm correct functioning, and for the experimental test, the purpose was to check if the “Battery” charge ever reached 0.



Right away, it can be seen that the TOP model’s states are composed of all the states of its atomic models. This confirms that the TOP model is correctly built. Secondly, each of the states can be compared with all of the individual state tests to once again confirm that they are identical and that no discrepancies between the links exist.

The next step is the experimental analysis which is the purpose of the simulation. Looking at the TOP model’s state trace file, “Battery” reaches a charge of 0 for the first time at time 00:02:31, 152 hours after being deployed. “Battery” only regains charge at time 00:02:58, 27 hours later. This demonstrates that the current power consumption is too high to maintain the consistent functioning of the forestry probe over time.

1. **Conclusion**

This simulation of a forestry probe’s power system allows for the analysis of the probe’s system requirements without needing to build the real system. DEVS formalism is a very powerful modeling tool, allowing for the production of robust and reusable designs. An example would be as follows. Currently, the above model of the forestry probe’s system is very non-specific, modeling only the general concepts within the system as a proof of concept with the generation of a general charge trace. However, this model can be easily expanded. Rather than using the previously described “Sensors” model which only models the general behavior and power consumption of up to 10 sensors, individual sensor models, each modeling a specific type of sensor, could be added by simply including a controller model within “Power Consumers” (refer to Appendix 1). This new model would offer a more realistic and in-depth simulation of the real system. That being said, the current model provides a general analysis of the limitations of the power system such that targeted improvements can be made in the future.

The use of DEVS formalism allowed for the modeling of the forestry probe’s power system which was further implemented in Cadmium. Furthermore, it generated a model that is both reusable and expandable.

References:

[1] E. and C. C. Canada, “Government of Canada / Gouvernement du Canada,” *Climate*, 17-Sep-2020. [Online]. Available: https://climate.weather.gc.ca/climate\_data/hourly\_data\_e.html?StationID=48370. [Accessed: 24-Oct-2020].

Appendix 1:

Alternate “Power Consumers” model architecture. The controller atomic model would model the algorithm that runs the sensors. A number of sensors equivalent to the number of ports on the controller can be added. Each sensor is a different atomic model, this allows for more complex and accurate representation of the real system.

