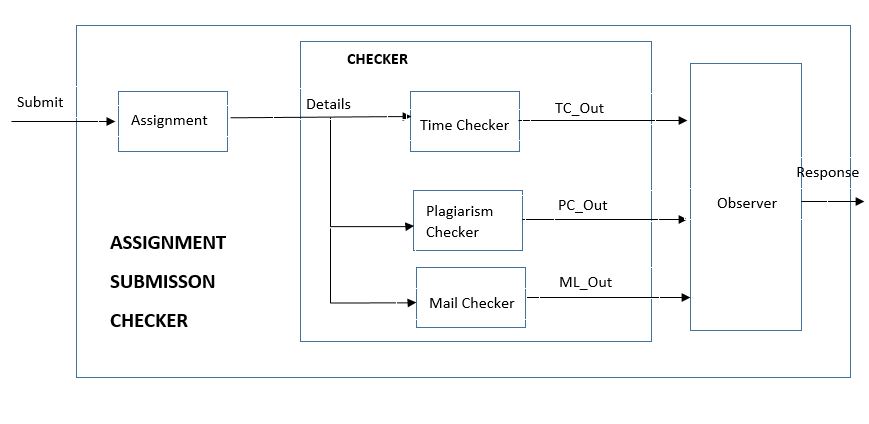
|  |  |
| --- | --- |
| A picture of a winding road and trees  Assignment Submision CHecker  Assignment 1 : SYSC 5104 : METH DISCRET-EVENT MODEL & SIM | Abstract  The document provides a DEVS model that checks the validity of the submitted assignment by checking the submission time, possible plagiarism and Email ID to which the assignment is submitted  User  Shashi Bhushan  UOttawa ID : 8490661 |

# Part I

The document shows a DEVS model that checks the validity of the submitted assignment by checking the submission time, possible plagiarism and Email ID the assignment is submitted to. The figure below shows the block diagram for the assignment submission checker. If the assignment is submitted before time on the right email id, and it is not plagiarized, successful response is provided. Otherwise, unsuccessful response is provided. The building blocks functionally is described as follows:



**Figure 1:** Block diagram for the assignment submission checker

* **Assignment:** The assignment block represents the assignment submitted by the user. Before the submit input is provided, the submit block is in passive state. The submit input triggers the block to be active and the *assignment is passed to the checker block*.
* **Checker:** The checker block is responsible for the validity of the submitted assignment. It consists of three sub blocks
* **Time Checker:** This block checks the time at which the assignment is submitted. If the assignment is submitted on or before time, output 1 is provided, else output 0 is provided. For simplicity, the time will be an integer *value ranging from 1 to 24* representing 24 hours of a day. If the assignment is submitted before time 12, the submission is valid and the output 1 is generated.
* **Plagiarism Checker:** This block checks the assignment for any possible plagiarism. If the assignment is plagiarized, output 0 is provided, else output 1 is provided. For simplicity, the plagiarism checker checks only the serial number of the assignment that can have a *value 1 through 99*. If the serial number is less than 10, the submission is invalid and output 0 is generated. If the serial number is more than 10, the submission is valid and the output 1 is generated.
* **Mail Checker:** This block checks the Email ID to which the assignment is submitted. If the assignment is submitted on the gwainer id (represented by value1), output 1 is provided. *It can have value 1 or 0*. On the other hand, if the assignment is submitted to gchair account (represented by value 0), output 0 is provided. For simplicity, the
* **Observer:** This block checks the output of all three blocks of the checker block and provides the final output. If all 3 outputs are 1, the submission is successful. If any or all of the outputs from the checker are 0, the submission is unsuccessful. The observer gives the successful or unsuccessful response with relevant response code depending on the response from the checker.

# Part II

The AssgnSubCheck has one input (submit) and one output (response). The submit input is a six digit integer number. The first digit should always be 1 and is not to be considered while computing. The second and third digit represents the submission time and can be *1 through 24*. The fourth and fifth digit represents the serial number of the assignment and the value can be *1 through 99*. The last digit can be *0 or 1*. The working of all the blocks of the models is described above. We now give the formal specification of all the components used in the model.

## Formal Specifications

The formal specifications <S, X, Y, δint, δext, λ, ta> for the atomic models are defined as follows:

### Observer:

S = {passive, active}

X = {TC\_Out, PC\_Out, ML\_Out}

Y = {Response}

δint (active) = passive

δext (TC\_Out or PC\_Out or ML\_Out, passive) = active

δext (TC\_Out or PC\_Out or ML\_Out, active) = active

λ (active)

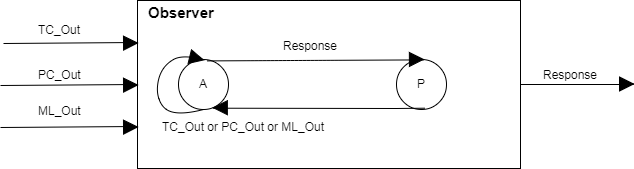
{

Check the inputs tc\_out, pc\_out and ml\_out. If all three inputs are 1, the output is 1. Otherwise, the output is 0.

}

ta (passive) = INFINITY

ta (active) = response\_time //Time taken by Observer block to generate response

****

### TimeChecker:

S = {passive, active}

X = {Details}

Y = {TC\_Out}

δint (active) = passive

δext (Details, passive) = active

δext (Details, active) = active

λ (active)

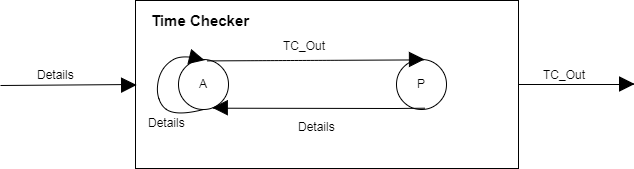
{

Extract the time field (digit 2 and 3) from the input value ’details” and check it. If it is smaller than or equal to 12, the output is 1, otherwise the output is 0.

}

ta (passive) = INFINITY

ta (active) = tc\_time //Time taken by the Time Checker block to generate output



### PlagiarismChecker:

S = {passive, active}

X = {Details}

Y = {PC\_Out}

δint (active) = passive

δext (Details, passive) = active

δext (Details, active) = active

λ (active)

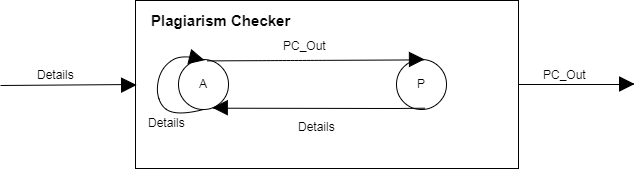
{

Extract the serial number field (digit 4 and 5) from the input value ’details” and check it. If it is smaller than or equal to 10, the output is 0, otherwise the output is 1.

}

ta (passive) = INFINITY

ta (active) = pc\_time //Time taken by the Plagiarism Checker block to generate output



### MailChecker:

S = {passive, active}

X = {Details}

Y = {ML\_Out}

δint (active) = passive

δext (Details, passive) = active

δext (Details, active) = active

λ (active)

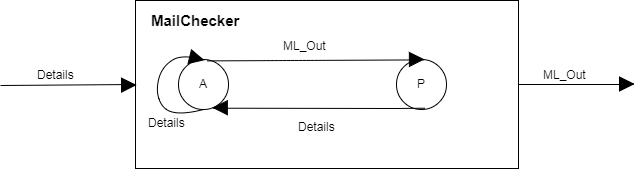
{

Extract the mail id field (digit 6) from the input value ’details” and check it. If it is gchair (0) the output is 0. If it is gwainer (1), the output is 1.

}

ta (passive) = INFINITY

ta (active) = ml\_time //Time taken by the Mail Checker block to generate output



### Assignment:

S = {passive, active}

X = {Submit}

Y = {Details}

δint (active) = passive

δext (Submit, passive) = active

δext (Submit, active) = active

λ (active)

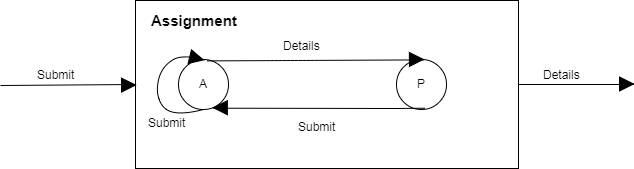
{

Send the “details” input to the checker module when submit input is provided.

}

ta (passive) = INFINITY

ta (active) = assgn\_time //Time taken by the Assignment block to forward details



The formal specifications < *X*, *Y*, D, {Mi}, IC, EIC, EOC, select > for the coupled models Checker and AssgnSubCheck are as follows:

### Checker:

Coupled Checker CM = <I, X, Y, D, {Mi}, {Ii}, {Zij}>

I = interfaces = {details, tc\_out, pc\_out, ml\_out}

X = Input Set = {details}

Y = Output Set = {tc\_out, pc\_out, ml\_out}

D = Index to components Mi

M1 = {TimeChecker}

M2 = {PlagiarismChecker}

M3 = {TimeChecker}

I1 = {Assignment}

I2 = {Observer}

Zij = mapping between components

details details@TimeChecker

details details@PlagiarismChecker

details details@MailChecker

tc\_out@TimeChekcer tc\_out

pc\_out@PlagiarismChecker pc\_out

ml\_out@MailChecker ml\_out

### AssgnSubCheck:

Coupled AssgnSubCheck CM = <I, X, Y, D, {Mi}, {Ii}, {Zij}>

I = interfaces = {submit, response}

X = Input Set = {submit}

Y = Output Set = {response}

D = Index to components Mi

M1 = {Assignment}

M2 = {Checker}

M3 = {Observer}

I1 = {}

Zij = mapping between components

submit submit@Assignmnet

details@Assignmnet details@Checker

tc\_out@Checker tc\_out@Observer

pc\_out@Checker pc\_out@Observer

ml\_out@Checker ml\_out@Observer

response@Observer response

# Part III

## Test Strategies

The atomic models and coupled will be tested using the “black box” testing method. Test cases are created by adding different combinations of input to the event file (.ev), run the simulation (.scp) and check whether the outputs in the output file (.out) are what we expected.

## Test Cases and Execution Analysis

### Atomic Model Observer:

The input of the Observer is the output of the coupled model Checker which in turn the output of the atomic models Time Checker, Plagiarism Checker and Mail Checker. The output of the observer is 1 if and only if all three inputs are 1. Otherwise, the output of the observer (response) is 0. All three 1 inputs means that the assignment is submitted on or before time, it is not plagiarized and it is submitted to the correct email id. The failure to do any three of these will result in unsuccessful submission. The output is generated after a fixed time duration after the inputs are received. The time duration in our case is 10 time units. The observer.ev file is created checking all the possible combinations as follows:

**00:00:10:00 ml\_out 1**

**00:00:10:00 pc\_out 1**

**00:00:10:00 tc\_out 1**

00:00:30:00 ml\_out 0

00:00:30:00 pc\_out 1

00:00:30:00 tc\_out 1

00:00:50:00 ml\_out 1

00:00:50:00 pc\_out 0

00:00:50:00 tc\_out 1

00:01:10:00 ml\_out 1

00:01:10:00 pc\_out 1

00:01:10:00 tc\_out 0

00:01:30:00 ml\_out 0

00:01:30:00 pc\_out 0

00:01:30:00 tc\_out 0

The inputs in the bold are the only case which results in the successful submission of the assignment shown by output value, response value1. Any other input combination will result in response 0 representing the unsuccessful submission. The output file ObserverOUt.out generated is as follows:

**00:00:20:000 response 1**

00:00:40:000 response 0

00:01:00:000 response 0

00:01:20:000 response 0

00:01:40:000 response 0

### Atomic model TimeChecker:

The input of the TimeChecker model is “details” which consists of the time of submission, serial number of assignment and the submission mail id. The model extracts the time field (digit 2 and 3) from the input and checks whether it is smaller than or equal to 12 which represents that the assignment is submitted before time and results in output tc\_out to be 1. Otherwise the output is 0. The output is generated after a fixed time duration after the inputs are received. The time duration in our case is 5 time units. The TimeChecker.ev file is created checking various possible values of the time as follows:

**00:00:10:00 details 110151**

00:00:20:00 details 115201

**00:00:30:00 details 111151**

00:00:40:00 details 116201

**00:00:50:00 details 101151**

00:01:00:00 details 124201

**00:01:10:00 details 105151**

00:01:20:00 details 119201

**00:01:30:00 details 104201**

The inputs in bold are the cases which results in the output tc\_out to be 1 representing that the assignment is submitted on or before time. Other inputs shown will result in response 0 representing that the assignment is submitted after due time. The output file TimeCheckerOUT.out generated is as follows:

**00:00:15:000 tc\_out 1**

00:00:25:000 tc\_out 0

**00:00:35:000 tc\_out 1**

00:00:45:000 tc\_out 0

**00:00:55:000 tc\_out 1**

00:01:05:000 tc\_out 0

**00:01:15:000 tc\_out 1**

00:01:25:000 tc\_out 0

**00:01:35:000 tc\_out 1**

### Atomic model PlagiarismChecker:

The input of the PlagiarismChecker model is “details” which consists of the time of submission, serial number of assignment and the submission mail id. The model extracts the serial number field (digit 4 and 5) from the input and checks whether it is greater than or equal to 10 which represents that the assignment submitted is not plagiarized and results in output pc\_out to be 1. Otherwise the output is 0. The output is generated after a fixed time duration after the inputs are received. The time duration in our case is 5 time units. The PlagiarismChecker.ev file is created checking various possible values of the assignment serial number as follows:

00:00:10:00 details 110081

**00:00:20:00 details 115201**

**00:00:30:00 details 111151**

**00:00:40:00 details 116101**

**00:00:50:00 details 101241**

00:01:00:00 details 124011

**00:01:10:00 details 105991**

00:01:20:00 details 119041

**00:01:30:00 details 104511**

The inputs in bold are the cases which results in output pc\_out to be 1 representing that the assignment submitted is not plagiarized. Other inputs shown will result in output 0 representing that the assignment submitted is plagiarized. The output file PlagiarismCheckerOUT.out generated is as follows:

00:00:15:000 pc\_out 0

**00:00:25:000 pc\_out 1**

**00:00:35:000 pc\_out 1**

**00:00:45:000 pc\_out 1**

**00:00:55:000 pc\_out 1**

00:01:05:000 pc\_out 0

**00:01:15:000 pc\_out 1**

00:01:25:000 pc\_out 0

**00:01:35:000 pc\_out 1**

### Atomic model MailChecker:

The input of the MailChecker model is “details” which consists of the time of submission, serial number of assignment and the submission mail id. The model extracts the mail id field (digit 6) from the input and checks whether it is equal to 1 which represents that the assignment is submitted to the right mail id and results in output ml\_out to be 1. The field 0 represents that the assignment is submitted to the wrong mail id and the output is 0. The output is generated after a fixed time duration after the inputs are received. The time duration in our case is 5 time units. The MailChecker.ev file is created checking various possible values of the assignment serial number as follows:

**00:00:10:00 details 110151**

00:00:20:00 details 115200

**00:00:30:00 details 111151**

00:00:40:00 details 116100

**00:00:50:00 details 101241**

**00:01:00:00 details 124011**

00:01:10:00 details 105990

00:01:20:00 details 119040

**00:01:30:00 details 104511**

The inputs in bold are the cases which results in output ml\_out to be 1 representing that the assignment is submitted to the correct mail id. Other inputs shown will result in output 0 representing that the assignment is submitted to the wrong mail id. The output file MailCheckerOUT.out generated is as follows:

**00:00:15:000 ml\_out 1**

00:00:25:000 ml\_out 0

**00:00:35:000 ml\_out 1**

00:00:45:000 ml\_out 0

**00:00:55:000 ml\_out 1**

**00:01:05:000 ml\_out 1**

00:01:15:000 ml\_out 0

00:01:25:000 ml\_out 0

**00:01:35:000 ml\_out 1**

### Coupled Model AssgnSubCheck:

The coupled model AssgnSubCheck is the top model which consists of atomic models Assignment, Observer and coupled model Checker. The checker consists of atomic models TimeChecker, PlagiarismChecker and MailChecker. The input “details” is send by the Assignment model to the Checker when the input submit is received. The Checker sends the time, serial number and mail id of the assignment and sends the corresponsing output to the Observer model, which generates the final output “response” based on all three inputs. If all three inputs are 1, the output is 1 representing that the assignment is submitted successfully. Otherwise the output is 0 representing that the assignment submission is unsuccessful. The AssgnSubCheck.ev file is created checking various possible values of the assignment serial number as follows:

**00:00:10:00 submit 110151**

00:00:40:00 submit 124151

00:01:10:00 submit 110071

00:01:40:00 submit 110150

00:02:10:00 submit 115080

**00:02:40:00 submit 101501**

**00:03:10:00 submit 105991**

00:03:40:00 submit 110070

00:04:10:00 submit 114090

**00:04:40:00 submit 104111**

The inputs in bold are the cases which results in output response to be 1 representing that the assignment is submitted successfully. Other inputs shown will result in output 0 representing that the assignment submission is unsuccessful. The output file AssgnSubCheckOUT.out generated is as follows:

**00:00:30:000 response 1**

00:01:00:000 response 0

00:01:30:000 response 0

00:02:00:000 response 0

00:02:30:000 response 0

**00:03:00:000 response 1**

**00:03:30:000 response 1**

00:04:00:000 response 0

00:04:30:000 response 0

**00:05:00:000 response 1**

The executed test cases shows that the model Assignment Submission Checker works as expected. The individual atomic models have been tested successfully and the integration testing on the top model also gives the expected results. The specification are met successfully.