Group 22

Author: Yuxin Miao

Software VALIDATION

Elevator Control System

Table of Contents

[System Architecture 2](#_Toc44612538)

[T1: Unit Test 2](#_Toc44612539)

[T1.1: Elevator\_Core Unit Test 2](#_Toc44612540)

[T1.1.1: Test moveUp() 2](#_Toc44612541)

[T1.1.2: Test moveDown() 3](#_Toc44612542)

[T1.1.3: Test openDoor() 3](#_Toc44612543)

[T1.1.4: Test closeDoor() 5](#_Toc44612544)

[T1.1.5: Test stop() 6](#_Toc44612545)

[T1.1.6: Test travelling() 7](#_Toc44612546)

[T1.2: Controller\_Panel 11](#_Toc44612547)

[T1.2.1:Test acceptUpOrder() 11](#_Toc44612548)

[T1.2.2:Test acceptDownOrder() 11](#_Toc44612549)

[T1.2.3:Test acceptOpenDoorOrder() 11](#_Toc44612550)

[1.2.4:Test acceptCloseDoorOrder() 12](#_Toc44612551)

[1.2.5:Test acceptStopOrder() 12](#_Toc44612552)

[1.2.6:Test acceptFloorOrder() 13](#_Toc44612553)

[T2: Integration Test 13](#_Toc44612554)

[T2.1: Elevator\_Core + Controller\_Core Integration 13](#_Toc44612555)

[T2.1.1:Test processRequest() 13](#_Toc44612556)

[T2.1.2: Test assignElevator() 14](#_Toc44612557)

[T3: Functional Test 15](#_Toc44612558)

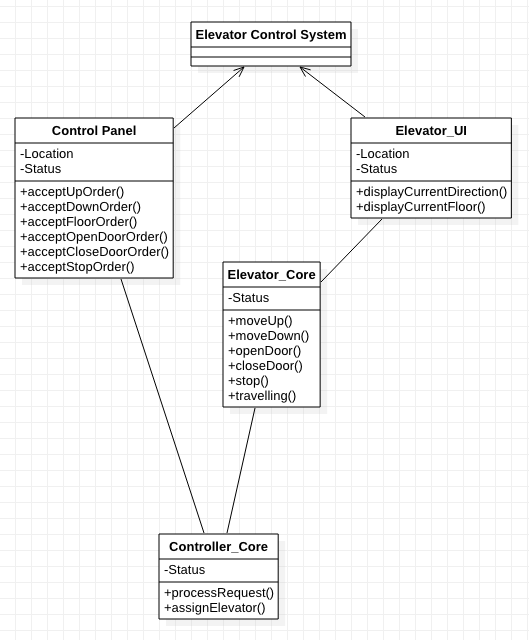
[T3.1: Test Single elevator 15](#_Toc44612559)

[T3.1.1:Test moveup 15](#_Toc44612560)

[T3.1.2:Test movedown 16](#_Toc44612561)

# System Architecture

The system architecture is shown below:



# T1: Unit Test

## T1.1: Elevator\_Core Unit Test

### T1.1.1: Test moveUp()

function up(e)

e.status=1; Tcover1.1.1.1

e.inUI.status.Text="up";

prev = e.floor;

e.floor = e.floor+1;

for v = 0:0.2:1

temp=prev+v;

e.inUI.floor.Value=temp;

pause(0.05);

end

e.updatepannel();

end

* Coverage Criteria: Statement coverage
* Test case

|  |  |
| --- | --- |
|  | Test Case T1.1.1.1 |
| Coverage Item | Tcover1.1.1.1 |
| Input |  |
| State | E1.floor = 1; |
| Expected Output | E1.floor == 2 |

* Test coverage: 1/1=100%
* Test result: 1 passed

### T1.1.2: Test moveDown()

function down(e)

e.status=-1; Tcover1.1.2.1

e.inUI.status.Text="down";

prev = e.floor;

e.floor = e.floor-1;

for v = 0:0.2:1

temp=prev-v;

e.inUI.floor.Value=temp;

pause(0.05);

end

e.updatepannel();

end

* Coverage Criteria: Statement coverage
* Test case

|  |  |
| --- | --- |
|  | Test Case T1.1.2.1 |
| Coverage Item | Tcover1.1.2.1 |
| Input |  |
| State | E1.floor = 2; |
| Expected Output | E1.floor == 1 |

* Test coverage: 1/1=100%
* Test result: 1 passed

### T1.1.3: Test openDoor()

function open(e)

pause(0.5);

e.inUI.lamp.Color=[1,0,0];

e.inUI.open.BackgroundColor=[1,1,1];

e.inUI.Image.ImageSource='open.png';

if e.floor == 1

if e.id == 1 Tcover1.1.3.1

e.F1UI.E1.ImageSource='open.png';

else Tcover1.1.3.2

e.F1UI.E2.ImageSource='open.png';

end

elseif e.floor == 2

if e.id == 1 Tcover1.1.3.3

e.F2UI.E1.ImageSource='open.png';

else Tcover1.1.3.4

e.F2UI.E2.ImageSource='open.png';

end

else

if e.id == 1 Tcover1.1.3.5

e.F3UI.E1.ImageSource='open.png';

else Tcover1.1.3.6

e.F3UI.E2.ImageSource='open.png';

end

end

if e.status == 1

if e.floor == 1 Tcover1.1.3.7

e.F1UI.up.ImageSource='up\_off.png';

elseif e.floor == 2 Tcover1.1.3.8

e.F2UI.up.ImageSource='up\_off.png';

end

elseif e.status == -1

if e.floor == 2 Tcover1.1.3.9

e.F2UI.down.ImageSource='down\_off.png';

elseif e.floor == 3 Tcover1.1.3.10

e.F3UI.down.ImageSource='down\_off.png';

end

else

if e.floor == 1 Tcover1.1.3.11

e.F1UI.up.ImageSource='up\_off.png';

elseif e.floor == 2 Tcover1.1.3.12

e.F2UI.up.ImageSource='up\_off.png';

e.F2UI.down.ImageSource='down\_off.png';

else Tcover1.1.3.13

e.F3UI.down.ImageSource='down\_off.png';

end

end

end

* Coverage Criteria: Condition coverage
* TestCase

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Coverage Item | Input | State | Expected Output |
| Test Case T1.1.3.1 | Tcover1.1.3.1  Tcover1.1.3.7 |  | E1.floor = 1;  E1.F1UI.E1.ImageSource = 'close.png'  E1.status = 1;  E1.F1UI.up.ImageSource='up\_on.png'; | E1.F1UI.E1.ImageSource=='open.png'  E1.F1UI.up.ImageSource=='up\_off.png' |
| Test Case T1.1.3.2 | Tcover1.1.3.2  Tcover1.1.3.11 |  | E1.floor = 2;  E1.F2UI.E1.ImageSource = 'close.png'  E1.status = 1;  E1.F1UI.up.ImageSource='up\_on.png'; | E1.F2UI.E1.ImageSource=='open.png'  E1.F2UI.up.ImageSource=='up\_off.png' |
| Test Case T1.1.3.3 | Tcover1.1.3.3  Tcover1.1.3.8 |  | E1.floor = 3;  E1.F1UI.E1.ImageSource = 'close.png'  E1.status = 1;  E1.F1UI.down.ImageSource='down\_on.png'; | E1.F3UI.E1.ImageSource=='open.png'  E1.F3UI.down.ImageSource=='down\_off.png' |
| Test Case T1.1.3.4 | Tcover1.1.3.4  Tcover1.1.3.12 |  | E2.floor = 1;  E2.F1UI.E2.ImageSource = 'close.png'  E2.status = 0;  E2.F1UI.up.ImageSource='up\_on.png'; | E2.F1UI.E2.ImageSource=='open.png'  E2.F1UI.up.ImageSource=='up\_off.png' |
| Test Case T1.1.3.5 | Tcover1.1.3.5  Tcover1.1.3.10 |  | E2.floor = 2;  E2.F1UI.E2.ImageSource = 'close.png'  E2.status = -1;  E2.F2UI.up.ImageSource='up\_on.png';  E2.F2UI.down.ImageSource='down\_on.png'; | E2.F2UI.E2.ImageSource=='open.png'  E2.F2UI.up.ImageSource=='up\_off.png'  E2.F2UI.down.ImageSource=='down\_off.png' |
| Test Case T1.1.3.6 | Tcover1.1.3.6  Tcover1.1.3.13 |  | E2.floor = 3;  E2.F3UI.E2.ImageSource = 'close.png'  E2.status = 0;  E2.F3UI.down.ImageSource='down\_on.png' | E2.F3UI.E2.ImageSource=='open.png'  E2.F3UI.down.ImageSource=='down\_off.png' |
| Test Case T1.1.3.7 | Tcover1.1.3.9 |  | E2.floor = 2;  E2.F1UI.E2.ImageSource = 'close.png'  E2.status = -1;  E2.F2UI.down.ImageSource='down\_on.png'; | E2.F2UI.E2.ImageSource=='open.png'  E2.F2UI.down.ImageSource=='down\_off.png' |

* Test coverage: 13/13=100%
* Test result: 13 passed

### T1.1.4: Test closeDoor()

function close(e)

e.inUI.lamp.Color=[0,1,0];

e.inUI.close.BackgroundColor=[1,1,1];

e.inUI.Image.ImageSource='close.png';

if e.floor == 1

if e.id == 1 Tcover1.1.4.1

e.F1UI.E1.ImageSource='close.png';

else Tcover1.1.4.2

e.F1UI.E2.ImageSource='close.png';

end

elseif e.floor == 2

if e.id == 1 Tcover1.1.4.3

e.F2UI.E1.ImageSource='close.png';

else Tcover1.1.4.4

e.F2UI.E2.ImageSource='close.png';

end

else

if e.id == 1 Tcover1.1.4.5

e.F3UI.E1.ImageSource='close.png';

else Tcover1.1.4.6

e.F3UI.E2.ImageSource='close.png';

end

end

end

* Coverage Criteria: Condition coverage
* TestCase

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Coverage Item | Input | State | Expected Output |
| Test Case T1.1.4.1 | Tcover1.1.4.1 |  | E1.floor = 1;  E1.F1UI.E1.ImageSource = 'open.png' | E1.F1UI.E1.ImageSource=='close.png' |
| Test Case T1.1.4.2 | Tcover1.1.4.2 |  | E1.floor = 2;  E1.F2UI.E1.ImageSource = 'open.png' | E1.F2UI.E1.ImageSource=='close.png' |
| Test Case T1.1.4.3 | Tcover1.1.4.3 |  | E1.floor = 3;  E1.F1UI.E1.ImageSource = 'open.png' | E1.F3UI.E1.ImageSource=='close.png' |
| Test Case T1.1.4.4 | Tcover1.1.4.4 |  | E2.floor = 1;  E2.F1UI.E2.ImageSource = 'open.png' | E2.F1UI.E2.ImageSource=='close.png' |
| Test Case T1.1.4.5 | Tcover1.1.4.5 |  | E2.floor = 2;  E2.F1UI.E2.ImageSource = 'open.png' | E2.F2UI.E2.ImageSource=='close.png' |
| Test Case T1.1.4.6 | Tcover1.1.4.6 |  | E2.floor = 3;  E2.F3UI.E2.ImageSource = 'open.png' | E2.F3UI.E2.ImageSource=='close.png' |

* Test coverage: 6/6=100%
* Test result: 6 passed

### T1.1.5: Test stop()

function stop(e)

if e.status>=0

e.removefromq(e.floor,1);

if e.floor==1 Tcover1.1.5.1

e.F1UI.up.ImageSource='up\_off.png';

%disp("reset F1 button")

e.inUI.F1.BackgroundColor=[1,1,1];

elseif e.floor==2 Tcover1.1.5.2

e.F2UI.up.ImageSource='up\_off.png';

%disp("reset F2 button")

e.inUI.F2.BackgroundColor=[1,1,1];

else Tcover1.1.5.3

%disp("reset F3 button")

e.inUI.F3.BackgroundColor=[1,1,1];

end

elseif e.status==-1

e.removefromq(e.floor,-1);

if e.floor==3 Tcover1.1.5.4

e.F3UI.down.ImageSource='down\_off.png';

%disp("reset F3 button")

e.inUI.F3.BackgroundColor=[1,1,1];

elseif e.floor==2 Tcover1.1.5.5

e.F2UI.down.ImageSource='down\_off.png';

%disp("reset F2 button")

e.inUI.F2.BackgroundColor=[1,1,1];

else Tcover1.1.5.6

e.inUI.F1.BackgroundColor=[1,1,1];

%disp("reset F1 button")

end

end

end

* Coverage Criteria: Condition coverage
* TestCase

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Coverage Item | Input | State | Expected Output |
| Test Case T1.1.5.1 | Tcover1.1.5.1 |  | E1.upq = [1,2,3];  E1.floor = 1;  E1.status = 0;  E1.F1UI.up.ImageSource = 'up\_on.png' | E1.F1UI.up.ImageSource=='up\_off.png'  E1.inUI.F1.BackgroundColor==[1,1,1] |
| Test Case T1.1.5.2 | Tcover1.1.5.2 |  | E1.upq = [1,2,3];  E1.floor = 2;  E1.status = 0;  E1.F1UI.up.ImageSource = 'up\_on.png' | E1.F2UI.up.ImageSource=='up\_off.png'  E1.inUI.F2.BackgroundColor==[1,1,1] |
| Test Case T1.1.5.3 | Tcover1.1.5.3 |  | E1.upq = [1,2,3];  E1.floor = 3;  E1.status = 0; | E1.inUI.F3.BackgroundColor==[1,1,1] |
| Test Case T1.1.5.4 | Tcover1.1.5.4 |  | E1.upq = [1,2,3];  E1.floor = 1;  E1.status = -1; | E1.inUI.F1.BackgroundColor==[1,1,1] |
| Test Case T1.1.5.5 | Tcover1.1.5.5 |  | E1.upq = [1,2,3];  E1.floor = 1;  E1.status = -1;  E1.F1UI.down.ImageSource = 'down \_on.png' | E1.F2UI.down.ImageSource=='down \_off.png'  E1.inUI.F2.BackgroundColor==[1,1,1] |
| Test Case T1.1.5.6 | Tcover1.1.5.6 |  | E1.upq = [1,2,3];  E1.floor = 1;  E1.status = -1;  E1.F1UI.down.ImageSource = 'down \_on.png' | E1.F3UI.down.ImageSource=='down \_off.png'  E1.inUI.F3.BackgroundColor==[1,1,1] |

* Test coverage: 6/6=100%
* Test result: 6 passed

### T1.1.6: Test travelling()

function inposition(e)

if isempty(e.upq) && isempty(e.downq) Tcover1.1.6.1

e.status=0;e.inUI.status.Text="stop";e.updatepannel();SLEEP(e.sf);

return;

end

if e.status == 1

if ismember(e.floor,e.upq) Tcover1.1.6.2

e.status=1;e.updatepannel();STOP(e.sf);return

end

if ~isempty(e.upq(e.upq>e.floor)) Tcover1.1.6.3

disp("condition 2");e.status=1;e.updatepannel();UP(e.sf);

return

end

if ~isempty(e.downq(e.downq>e.floor)) Tcover1.1.6.4

e.status=1;e.updatepannel();UP(e.sf);

return

end

if ismember(e.floor,e.downq) Tcover1.1.6.5

disp("condition 4");e.status=-1;e.updatepannel();STOP(e.sf);

return

end

if ~isempty(e.downq(e.downq<e.floor)) Tcover1.1.6.6

e.status=-1;e.updatepannel();DOWN(e.sf);

return

end

if ~isempty(e.upq(e.upq<e.floor)) Tcover1.1.6.7

e.status=-1; e.updatepannel();DOWN(e.sf);

return

end

elseif e.status == -1

if ismember(e.floor,e.downq) Tcover1.1.6.8

e.status=-1;e.updatepannel();STOP(e.sf);

return

end

if ~isempty(e.downq(e.downq<e.floor)) Tcover1.1.6.9

e.status=-1;e.updatepannel();DOWN(e.sf);

return

end

if ~isempty(e.upq(e.upq<e.floor)) Tcover1.1.6.10

e.status=-1;e.updatepannel();DOWN(e.sf);

return

end

if ismember(e.floor,e.upq) Tcover1.1.6.11

e.status=1;e.updatepannel();STOP(e.sf);

return

end

if ~isempty(e.upq(e.upq>e.floor)) Tcover1.1.6.12

e.status=1;e.updatepannel();UP(e.sf);

return

end

if ~isempty(e.downq(e.downq>e.floor)) Tcover1.1.6.13

e.status=1;e.updatepannel();UP(e.sf);

return

end

else

if ismember(e.floor,e.upq) Tcover1.1.6.14

e.status=1;e.updatepannel();STOP(e.sf);

return

end

if ismember(e.floor,e.downq) Tcover1.1.6.15

e.status=-1;e.updatepannel();STOP(e.sf);

return

end

if ~isempty(e.upq(e.upq>e.floor)) Tcover1.1.6.16

e.status=1;e.updatepannel();UP(e.sf);

return

end

if ~isempty(e.downq(e.downq<e.floor)) Tcover1.1.6.17

e.status=-1;e.updatepannel();DOWN(e.sf);

return

end

if ~isempty(e.downq(e.downq>e.floor)) Tcover1.1.6.18

e.status=1;e.updatepannel();UP(e.sf);

return

end

if ~isempty(e.upq(e.upq<e.floor)) Tcover1.1.6.19

e.status=-1;e.updatepannel();DOWN(e.sf);

return

end

end

end

* Coverage Criteria: Branch coverage
* TestCase

|  |  |  |  |
| --- | --- | --- | --- |
|  | Coverage Item | State | Expected Output |
| Test Case T1.1.6.1 | Tcover1.1.6.1 | E1.upq = []  E1.downq = [] | testCase.E1.inUI.status.Text==char("stop") |
| Test Case T1.1.6.2 | Tcover1.1.6.2 | E1.status = 1;  UP(testCase.E1.sf);  E1.upq = [2];  E1.downq = []; | E1.floor == 2 |
| Test Case T1.1.6.3 | Tcover1.1.6.3 | E1.status = 1;  E1.upq = [2];  E1.downq = []; | E1.floor == 2 |
| Test Case T1.1.6.4 | Tcover1.1.6.4 | E1.status = 1;  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [3]; | E1.floor == 3 |
| Test Case T1.1.6.5 | Tcover1.1.6.5 | E1.status = 1;  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [2]; | E1.floor == 2 |
| Test Case T1.1.6.6 | Tcover1.1.6.6 | E1.status = 1;  UP(testCase.E1.sf);  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [2]; | E1.floor == 2 |
| Test Case T1.1.6.7 | Tcover1.1.6.7 | E1.status = 1;  UP(testCase.E1.sf);  UP(testCase.E1.sf);  E1.upq = [2];  E1.downq = []; | E1.floor == 2 |
| Test Case T1.1.6.8 | Tcover1.1.6.8 | E1.status = -1;  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [2]; | E1.floor == 2 |
| Test Case T1.1.6.9 | Tcover1.1.6.9 | E1.status = -1;  UP(testCase.E1.sf);  UP(testCase.E1.sf);s  E1.upq = [];  E1.downq = [2]; | E1.floor == 2 |
| Test Case T1.1.6.10 | Tcover1.1.6.10 | E1.status = -1;  UP(testCase.E1.sf);  E1.upq = [1];  E1.downq = [3]; | E1.floor == 1 |
| Test Case T1.1.6.11 | Tcover1.1.6.11 | E1.status = -1;  UP(testCase.E1.sf);  E1.upq = [2];  E1.downq = [3]; | E1.floor == 2 |
| Test Case T1.1.6.12 | Tcover1.1.6.12 | E1.status = -1;  E1.upq = [2];  E1.downq = [3]; | E1.floor == 2 |
| Test Case T1.1.6.13 | Tcover1.1.6.13 | E1.status = -1;  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [3]; | E1.floor == 3 |
| Test Case T1.1.6.14 | Tcover1.1.6.14 | E1.status = 0;  UP(testCase.E1.sf);  E1.upq = [2];  E1.downq = []; | E1.floor == 2 |
| Test Case T1.1.6.15 | Tcover1.1.6.15 | E1.status = 0;  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [2]; | E1.floor == 2 |
| Test Case T1.1.6.16 | Tcover1.1.6.16 | E1.status = 0;  E1.upq = [2];  E1.downq = []; | E1.floor == 2 |
| Test Case T1.1.6.17 | Tcover1.1.6.17 | E1.status = 0;  UP(testCase.E1.sf);  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [2]; | E1.floor == 2 |
| Test Case T1.1.6.18 | Tcover1.1.6.18 | E1.status = 0;  UP(testCase.E1.sf);  E1.upq = [];  E1.downq = [3]; | E1.floor == 3 |
| Test Case T1.1.6.19 | Tcover1.1.6.19 | E1.status = 0;  UP(testCase.E1.sf);  E1.upq = [1];  E1.downq = []; | E1.floor == 1 |

* Test coverage: 19/19=100%
* Test result: 19 passed

## T1.2: Controller\_Panel

### T1.2.1:Test acceptUpOrder()

**Because using stateflow will change the imagesource information, so I decided to comment out all stateflow to acquire a more accurate information**

function upImageClicked(app, event)

app.up.ImageSource='up\_on.png'; Tcover1.2.1.1

app.cc.callelevator(app.id,1);

end

* Coverage Criteria: Statement coverage
* Test case

|  |  |
| --- | --- |
|  | Test Case T1.2.1.1 |
| Coverage Item | Tcover1.2.1.1 |
| Input |  |
| State | F2.up.ImageSource = 'up\_off.png' |
| Expected Output | F2.up.ImageSource == 'up\_on.png' |

* Test coverage: 1/1=100%
* Test result: 1 passed

### T1.2.2:Test acceptDownOrder()

**Because using stateflow will change the BackgroundColor information, so I decided to comment out all stateflow to acquire a more accurate information**

function downButtonPushed(app, event)

app.down.BackgroundColor=[1,0,0]; Tcover1.2.2.1

app.cc.callelevator(app.id,-1);

end

* Coverage Criteria: Statement coverage
* Test case

|  |  |
| --- | --- |
|  | Test Case T1.2.2.1 |
| Coverage Item | Tcover1.2.2.1 |
| Input |  |
| State | F2.down.BackgroundColor = [0,0,0] |
| Expected Output | F2.down.BackgroundColor = [1,0,0] |

* Test coverage: 1/1=100%
* Test result: 1 passed

### T1.2.3:Test acceptOpenDoorOrder()

**Because using stateflow will change the** **BackgroundColor information, so I decided to comment out all stateflow to acquire a more accurate information**

function openPushed(app, event)

app.open.BackgroundColor=[1,0,0]; Tcover1.2.3.1

app.cc.open(app.id);

end

* Coverage Criteria: Statement coverage
* Test case

|  |  |
| --- | --- |
|  | Test Case T1.2.3.1 |
| Coverage Item | Tcover1.2.3.1 |
| Input |  |
| State | E1UI.open.BackgroundColor=[0,0,0] |
| Expected Output | E1UI.open.BackgroundColor==[1,0,0] |

* Test coverage: 1/1=100%
* Test result: 1 passed

### T1.2.4:Test acceptCloseDoorOrder()

**Because using stateflow will change the BackgroundColor information, so I decided to comment out all stateflow to acquire a more accurate information**

function closePushed(app, event)

app.close.BackgroundColor=[1,0,0]; Tcover1.2.4.1

app.cc.close(app.id);

end

* Coverage Criteria: Statement coverage
* Test case

|  |  |
| --- | --- |
|  | Test Case T1.2.4.1 |
| Coverage Item | Tcover1.2.4.1 |
| Input |  |
| State | E1UI.close.BackgroundColor=[0,0,0] |
| Expected Output | E1UI.close.BackgroundColor==[1,0,0] |

* Test coverage: 1/1=100%
* Test result: 1 passed

### T1.2.5:Test acceptStopOrder()

function stopPushed(app, event)

if (app.stop.BackgroundColor==[0.96,0.96,0.96]) Tcover1.2.5.1

disp("stoped!")

app.cc.emergency(app.id);

app.disableall();

elseif (app.stop.BackgroundColor==[1,1,1]) Tcover1.2.5.2

disp("stoped!")

app.cc.emergency(app.id);

app.disableall();

else Tcover1.2.5.3

app.enableall();

app.cc.restart(app.id);

disp("restart!")

end

end

* Coverage Criteria: Branch coverage
* Test case

|  |  |  |  |
| --- | --- | --- | --- |
|  | Coverage Item | State | Expected Output |
| Test Case T1.2.5.1 | Tcover1.2.5.1 | app.stop.BackgroundColor  ==[0.96,0.96,0.96] | app.F3.enable==0  app.F2.enable==0  app.F1.enable==0  app.open.enable==0  app.close.enable==0 |
| Test Case T1.2.5.2 | Tcover1.2.5.2 | app.stop.BackgroundColor  ==[1,1,1] | app.F3.enable==0  app.F2.enable==0  app.F1.enable==0  app.open.enable==0  app.close.enable==0 |
| Test Case T1.2.5.3 | Tcover1.2.5.3 | app.stop.BackgroundColor  ==[0,0,0] | app.F3.enable==1  app.F2.enable==1  app.F1.enable==1  app.open.enable==1  app.close.enable==1 |

* Test coverage: 3/3=100%
* Test result: 3 passed

### T1.2.6:Test acceptFloorOrder()

**Because using stateflow will change the upq information and downq information, so I decided to comment out all stateflow to acquire a more accurate information**

function F3Pushed(app, event)

app.F3.BackgroundColor=[1,0,0]; Tcover1.2.6.1

app.cc.goto(app.id,3);

end

function F2Pushed(app, event)

app.F2.BackgroundColor=[1,0,0]; Tcover1.2.6.2

app.cc.goto(app.id,2);

end

function F1Pushed(app, event)

app.F1.BackgroundColor=[1,0,0]; Tcover1.2.6.3

app.cc.goto(app.id,1);

end

* Coverage Criteria: State coverage
* Test case

|  |  |  |  |
| --- | --- | --- | --- |
|  | Coverage Item | State | Expected Output |
| Test Case T1.2.6.1 | Tcover1.2.6.1 |  | app.F3. BackgroundColor=[1,0,0] |
| Test Case T1.2.6.2 | Tcover1.2.6.2 |  | app.F3. BackgroundColor=[1,0,0] |
| Test Case T1.2.6.3 | Tcover1.2.6.3 |  | app.F3. BackgroundColor=[1,0,0] |

* Test coverage: 3/3=100%
* Test result: 3 passed

# T2: Integration Test

## T2.1: Elevator\_Core + Controller\_Core Integration

### T2.1.1:Test processRequest()

**Because using stateflow will change the upq information and downq information, so I decided to comment out all stateflow to acquire a more accurate information**

function goto(cc,id,floor)

if id == 1

if floor>=cc.E1.floor

cc.E1.addtoq(floor,1) Tcover2.1.1.1

else

cc.E1.addtoq(floor,-1) Tcover2.1.1.2

end

else

if floor>=cc.E2.floor%

cc.E2.addtoq(floor,1) Tcover2.1.1.3

else

cc.E2.addtoq(floor,-1) Tcover2.1.1.4

end

end

end

* Coverage Criteria: Condition coverage
* TestCase

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Coverage Item | Input | State | Expected Output |
| Test Case T2.1.1.1 | Tcover2.1.1.1 | Id : 1  Floor : 3 | E1.floor = 1; | E1.upq==[3] |
| Test Case T2.1.1.2 | Tcover2.1.1.2 | id : 1  Floor : 1 | E1.floor = 2; | E1.upq==[1] |
| Test Case T2.1.1.3 | Tcover2.1.1.3 | id : 2  Floor : 3 | E2.floor = 1; | E2.upq==[3] |
| Test Case T2.1.1.4 | Tcover2.1.1.4 | id : 2  Floor : 1 | E2.floor = 2; | E2.upq==[1] |

* Test coverage: 4/4=100%
* Test result: 4 passed

### T2.1.2: Test assignElevator()

function callelevator(cc,floor,dir)

if dir == cc.E1.status Tcover2.1.2.1

delta = floor-cc.E1.floor;

if (delta/dir)>=0

cc.E1.addtoq(floor,dir);

return

end

end

if dir == cc.E2.status Tcover2.1.2.2

delta = floor-cc.E2.floor;

if (delta/dir)>=0

cc.E2.addtoq(floor,dir);

return

end

end

if cc.E1.status == 0 Tcover2.1.2.3

cc.E1.addtoq(floor,dir);

WAKE(cc.E1.sf);

return

end

if cc.E2.status == 0 Tcover2.1.2.4

cc.E2.addtoq(floor,dir);

WAKE(cc.E2.sf);

return

end

r=rand; Tcover2.1.2.5

if r>=0.5

cc.E1.addtoq(floor,dir);

disp('add to E1');

else

cc.E2.addtoq(floor,dir);

disp('add to E2');

end

end

* Coverage Criteria: Condition coverage
* TestCase

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Coverage Item | Input | State | Expected Output |
| Test Case T2.1.2.1 | Tcover2.2.2.1 | floor : 2  dir : 1 | ctl\_core.E1.status = 1;  ctl\_core.E1.floor = 1; | E1.upq == [2] |
| Test Case T2.1.2.2 | Tcover2.1.2.2 | floor : 2  dir : 1 | ctl\_core.E1.status = 1;  ctl\_core.E1.floor = 3;  ctl\_core.E2.status = 1;  ctl\_core.E2.floor = 1; | E2.upq == [2] |
| Test Case T2.1.2.3 | Tcover2.1.2.3 | floor : 2  dir : 1 | ctl\_core.E1.status = 0;  ctl\_core.E1.floor = 3; ctl\_core.E2.status = 1;  ctl\_core.E2.floor = 3; | E1.upq == [2] |
| Test Case T2.1.2.4 | Tcover2.1.2.4 | floor : 2  dir : 1 | ctl\_core.E1.status = 1;  ctl\_core.E1.floor = 3; ctl\_core.E2.status = 0;  ctl\_core.E2.floor = 1; | E2.upq == [2] |
| Test Case T2.1.2.5 | Tcover2.1.2.5 | floor : 2  dir : 1 | ctl\_core.E1.status = 1;  ctl\_core.E1.floor = 3; ctl\_core.E2.status = 1;  ctl\_core.E2.floor = 3; | [E2.upq,E1.upq]  ==  [2] |

* Test coverage: 5/5=100%
* Test result: 5 passed

T2.2: ServerUI+OrderProcessor+OrderDB Integration

# T3: Functional Test

## T3.1: Test Single elevator

### T3.1.1:Test moveup

In this scenario, we will test all conditions that cause a single elevator to moveup.

function testMoveup(testCase)

testCase.press(testCase.E1UI.F2);

testCase.press(testCase.E1UI.F3);

testCase.press(testCase.E1UI.F1);

testCase.press(testCase.E1UI.F3);

testCase.press(testCase.E1UI.F1);

testCase.press(testCase.F2.up);

testCase.press(testCase.F3.down);

testCase.press(testCase.E1UI.F1)

testCase.press(testCase.F2.down);

testCase.press(testCase.F2.up);

end

Since the test result is passed, we can assume that all of the scenario of move up can’t be executed when in top floor.

### T3.1.2:Test movedown

In this scenario, we will test all conditions that cause a single elevator to movedown.

function testMovedown(testCase)

testCase.press(testCase.E1UI.F3);

testCase.press(testCase.E1UI.F2);

testCase.press(testCase.E1UI.F1);

testCase.press(testCase.E1UI.F3);

testCase.press(testCase.E1UI.F1);

testCase.press(testCase.E1UI.F3);

testCase.press(testCase.F2.up);

testCase.press(testCase.F1.up);

testCase.press(testCase.E1UI.F3)

testCase.press(testCase.F2.down);

end

Since the test result is passed, we can assume that all of the scenario of move down can’t be executed when in the bottom floor.