**Part 1) Conceptual Model Design**

**Description**

I am looking to model the evacuation of people from an arctic evacuation site along the northwest passage to a Forward Operating Location (FOL) where they would receive proper aid. This is a simpler version of the scenario outlined in the paper, “Modeling a Major Maritime Disaster Scenario using the Universal Modelling Framework for Sequential Decisions” [1]. In this scenario 500 people on a cruise ship travelling through the northern arctic need to be evacuated. Unfortunately, evacuation sites the people could safely reach are not accessible by large airplanes. Therefore, an FOL is established within helicopter range and people are moved from the evacuation site to the FOL via helicopter. My model would follow the assumptions made in the aforementioned paper and would use a subset of the benchmark policies listed within. It will model the deterioration of each evacuee’s health using the same parameters. I am interested in seeing whether I can get similar results to their case studies.

**Model**

**A diagram of a helicopter model

Description automatically generated**

**Components**

The Evacuee coupled model represents individual people and keeps track of their current location, triage status, and medical deterioration over time according to the distributions outlined in the paper [1]. The Helicopter coupled model represents helicopters and keeps track of their location, current occupancy, max occupancy, how much occupancy is taken up by stretchers, and how long each activity takes. The evacuation site model keeps track of the evacuees still at the evacuation site, what policy is used to load the helicopters, which helicopter is landing next, and when a helicopter is ready to leave. The FOL model keeps a list of evacuees who have been evacuated.

My plan currently is that the evacuee model outputs its state as a struct to be added to the lists of people in the helicopter, evacuation site, and FOL models. The evacuation site outputs to the helicopter model when to leave, and to evacuees when to enter the helicopter. The helicopter model outputs to the evacuation site that it has arrived and to the evacuee model when they arrive at the FOL. The FOL currently has no outputs and just stores evacuees, but this could be expanded upon. FilterIn and FilterOut atomic models make sure the leaving and entering messages sent by the evacuees reach the correct helicopter. The FilterLeave atomic model makes sure the “okay to leave” command sent by the evacuation site reaches the correct helicopter. The FilterLoc atomic model makes sure that the correct location information is sent to the correct evacuee.

**References**

[1] M. Rempel, “Modelling a major maritime disaster scenario using the universal modelling framework for sequential decisions”, Safety Science, Volume 171, 2024, 106379, ISSN 0925-7535, <https://doi.org/10.1016/j.ssci.2023.106379>.

**Part 2) Specification Document**

**Coupling Scheme**

|  |  |
| --- | --- |
| **Atomic Models** | **Coupled Models** |
| FilterLoc, FilterEvac, FilterES, Evacuee, Helicopter, FOL, Helicopter Queue, Helipad Manager, Evacuee Manager | Evacuee Coupled, Helicopter Coupled, Evacuation Site |

It will a slightly different coupling scheme as proposed in the conceptual design. The Evacuation Site has been changed into a coupled model that contains three atomic models: Helicopter Queue, Helipad Manager, and Evacuee Manager. The Helicopter Queue manages the helicopters waiting to land, the Helipad Manager manages what helicopter has landed and for how long. And the Evacuee Manager manages how many evacuees are in each triage status and decides who will be evacuated next. The Helicopter coupled model also has different filters. Instead of 3 filters it only has two. They are FilterEvac which filters EvacInfo messages to the Helicopter, and FilterES which filters HeloInfo messages to the Helicopter from the Evacuation Site. The Evacuee model now has three outputs. They are outES for EvacInfo messages to the Evacuation Site, outHelo for EvacInfo messages to the Helicopter model, and outFOL for EvacInfo messages to the FOL atomic model. It follows this diagram.

A screenshot of a diagram

Description automatically generated

**Coupled Models Formal Specification**

Evacuee Coupled=

< X, Y, {FilterLoc, Evacuee}, EIC, EOC, IC, SELECT >

X = in

Y = outES, outHelo, outFOL

EIC = {(EvacueeCoupled.in, FilterLoc.in)}

EOC = {(Evacuee.outES, EvacueeCoupled.outES), (Evacuee.outHelo, EvacueeCoupled.outHelo), (Evacuee.outFOL, EvacueeCoupled.outFOL)}

IC = {(FilterLoc.out, Evacuee.in)}

SELECT : ({FilterLoc, Evacuee}) = FilterLoc

Helicopter Coupled=

< X, Y, {FilterEvac, FilterES, Helicopter}, EIC, EOC, IC, SELECT >

X = inEvac, inES

Y = outEvac, outES

EIC = {(HelicopterCoupled.inEvac, FilterEvac.in), (HelicopterCoupled.inES, FilterES.in)}

EOC = {(Helicopter.outES, HelicopterCoupled.outES), (Helicopter.outEvac, HelicopterCoupled.outEvac)}

IC = {(FilterEvac.out, Helicopter.inEvac), (FilterES.out, Helicopter.inES)}

SELECT : ({FilterEvac, FilterES, Helicopter}) = FilterEvac

({FilterES, Helicopter}) = FilterES

Evacuation Site=

< X, Y, {HelicopterQueue, HelipadManager, EvacueeManager}, EIC, EOC, IC, SELECT >

X = inEvac, inHelo

Y = outHelo, outEvac

EIC = {(EvacuationSite.inEvac, EvacueeManager.inEvac), (EvacuationSite.inHelo, HelicopterQueue.inHelo)}

EOC = {(HelicopterQueue.outHelo, EvacuationSite.outHelo), (HelipadManager.outHelo, EvacuationSite.outHelo), (EvacueeManager.outEvac, EvacuationSite.outEvac)

IC = {(HelicopterQueue.outHM, HelipadManager.inHQ), (HelipadManager.outHQ, HelicopterQueue.inHM), (HelipadManager.outEM, EvacueeManager.inHM), (EvacueeManager.outHM, HelipadManager.inEM)}

SELECT : ({HelicopterQueue, HelipadManager, EvacueeManager}) = HelicopterQueue

({HelipadManager, EvacueeManager}) = HelipadManager

**Atomic Models Formal Specification/Model Testing**

I will explain how some tests execute using the DEVS graph as the formal model. My experimental strategy is to set the initial state and then provide test inputs at set times. If the model’s behavior is correct, it passes the test.

A diagram of a diagram

Description automatically generated

A diagram of a diagram

Description automatically generated

A diagram of a virus

Description automatically generated

A diagram of a diagram

Description automatically generated

A diagram of a helicopter

Description automatically generated

Helicopter Test

Initial State: Location is FOL, stop = false, and ta = 15

Test Input:

time 176, inES receives HeloInfo with stop = False

time 191, inES receives HeloInfo with stop = False

time 191, inEvac receives EvacInfo

time 510, inES receives HeloInfo with stop = True

Execution:

At time 0 it is in the FOL

At time 15 it is travelling to the Evacuation Site

At time 165 it has output a message of type HeloInfo to the Evacuation Site

At time 176 It receives an input from inES of type HeloInfo with the Boolean False

At time 176 it lands at the Evacuation Site

At time 191 it receives an input from inES of type HeloInfo with the Boolean False

At time 191 it receives an input from inEvac of type EvacInfo

At time 191 it is travelling to the FOL

At time 341 it sends an EvacInfo to outEvac

At time 341 it arrives at the FOL

At time 356 it is travelling to the Evacuation Site

At time 506 it outputs a message of type HeloInfo to the Evacuation Site

At 506 it is waiting to land at the Evacuation Site

At 510 it receives an input from inES of type HeloInfo with the Boolean True

At 510 it is travelling to the FOL

At 660 it arrives at the FOL

Execution Terminates as Helicopter passivates forever.

A diagram of a company

Description automatically generated

Helicopter Queue Test

Initial State: Queue Empty, needHelo = false, waitingForHelo = true, stop = false, ta = inf

Test Input:

At time 0, receives Boolean from inHM that is False

At time 165, receives HeloInfo from inHelo

At time 165, receives HeloInfo from inHelo

At time 180, receives Boolean from inHM that is True

Execution:

At time 0, receives Boolean from inHM that is False

At time 0, needHelo is now True, passivates

At time 165, receives 2 HeloInfo from inHelo

At time 165, waitingForHelo = false and Queue is not empty

At time 165, sends HeloInfo output to outHM and outHelo

At time 165, Queue is not empty, needHelo = false, waitingForHelo = false, passivates

At time 180, receives Boolean from inHM that is True

At time 180, Queue is not empty, needHelo = false, waitingForHelo = false, stop = true

At time 180, sends HeloInfo with Boolean True to outHelo

At time 180, Queue is empty, needHelo = false, waitingForHelo = false, stop = true, passivates

Execution Terminates as Helicopter Queue passivates forever.

A diagram of a company

Description automatically generated

A diagram of a company

Description automatically generated

I found that with the DEVS graphs it’s hard to graph the internal logic happening at each state. So I included the pseudo code explanation for the Evacuee Manager as well, because if I were able to display each state within the Evacuee Manager the DEVS graph would be too complex. I should look into decomposing the Evacuee Manager into two atomic models where one manages the evacuees’ triage statuses and the other selects evacuees based upon its triage policy in the future.

**Atomic Model: Evacuee Manager**

**State Variables:** sigma = inf, evacueesOnSite, evacueesToLoad, int heloID = -1, int capacityTaken = 0, bool stop = false, bool evacueesLoading = false

**Parameters:**

capacity = 10, with\_stretcher = 3, without\_stretcher = 1

**Dext(s,e,x):**

Case(port):

InHM:

For msgs from port inHM:

capacityTaken = 0

While (capacityTaken < capacity):

If there are evacuees with triage\_status Green:

Add first Green triage evacuee to evacueesToLoad list

Add without\_stretcher to capacityTaken

Otherwise, if there are evacuees with triage\_status White:

Add first White triage evacuee to evacueesToLoad list

Add without\_stretcher to capacityTaken

Otherwise, if there are evacuees with triage\_status Red:

Add first Red triage evacuee to evacueeToLoad list

Add with\_stretcher to capacityTaken

Otherwise, if there are evacuees with triage\_status Yellow:

Add first Yellow triage evacuee to evacuessToLoad list

Add with\_stretcher to capacityTaken

Otherwise, if there are no more evacuees to load:

Stop = true

capacityTaken = capacity

If capacityTaken > capacity:

Remove last added evacuee from evacueesToLoad

Sigma = 0

If capacityTaken > 0:

evacueesLoading = true

inEvac:

for msgs from port inEvac:

switch(msg.triage\_status):

case(White):

move evacuee to Green

case(Green):

move evacuee to Yellow

case(Yellow):

move evacuee to Red

case(Red):

move evacuee to Black

**Dint(s):**

if evacueesLoading == true:

evacueesLoading = false

evacueesToLoad is cleared

sigma = inf

**Output(s):**

if stop == false:

if evacueesLoading == true:

for all evacuees in evacueesToLoad:

send EvacInfo to outEvac

if stop == true:

send True to outHM

**TA(s):**

TA = sigma

A diagram of a number of objects

Description automatically generated

**Model Experiments to Execute**

**FilterLoc Tests**

Initial Conditions:

FilterLoc has EvacueeID = 1

1)

Test Input:

At time 0: EvacInfo message with evacueeID = 1, enteringOrLeaving = T, heloID = 2, triage\_status = ‘W’

At time 10: EvacInfo message with evacueeID = 1, enteringOrLeaving = F, heloID = 2, triage\_status = ‘W’

2)

Test Input:

At time 0: EvacInfo message with evacueeID = 2, enteringOrLeaving = T, heloID = 2, triage\_status = ‘W’

At time 10: EvacInfo message with evacueeID = 1, enteringOrLeaving = F, heloID = 2, triage\_status = ‘W’

**Evacuee Tests**

Initial Conditions:

triage\_status is White, curr\_loc is Evacuation Site, heloID is -1

1)

Test input:

At time 0: EvacInfo message with evacueeID = 1, enteringOrLeaving = T, heloID = 3, triage\_status = ‘W’

At time 10: EvacInfo message with evacueeID = 1, enteringOrLeaving = F, heloID = 3, triage\_status = ‘W’

2)

Test input:

At time 0: EvacInfo message with evacueeID = 1, enteringOrLeaving = F, heloID = 3, triage\_status = ‘W’

At time 10: EvacInfo message with evacueeID = 1, enteringOrLeaving = T, heloID = 5, triage\_status = ‘G’

**EvacueeCoupled Tests**

Initial Conditions:

FilterLoc with evacueeID = 1

Evacuee with evacueeID = 1, triage\_status is White, curr\_loc is Evacuation Site, heloID is -1

1)

Test Input:

At time 0: EvacInfo message received on EvacueeCoupled.in with evacueeID = 2, enteringOrLeaving = T, heloID = 3, triage\_status = ‘W’

At time 0: EvacInfo message received on EvacueeCoupled.in with evacueeID = 1, enteringOrLeaving = T, heloID = 3, triage\_status = ‘W’

At time 10: EvacInfo message received on EvacueeCoupled.in with evacueeID = 1, enteringOrLeaving = F, heloID = 3, triage\_status = ‘W’

2)

Test Input:

At time 0: EvacInfo message received on EvacueeCoupled.in with evacueeID = 1, enteringOrLeaving = F, heloID = 5, triage\_status = ‘W’

At time 10: EvacInfo message received on EvacueeCoupled.in with evacueeID = 1, enteringOrLeaving = T, heloID = 3, triage\_status = ‘W’

**FilterES Tests**

Initial Conditions:

FilterES has HeloID = 1

1)

Test Input:

At time 0: HeloInfo message with heloID = 1, stop = false

At time 10: HeloInfo message with heloID = 1, stop = true

2)

Test Input:

At time 0: HeloInfo message with heloID = 2, stop = false

At time 10: HeloInfo message with heloID = 3, stop = true

**FilterEvac Tests**

Initial Conditions:

FilterEvac has HeloID = 1

1)

Test Input:

At time 0: EvacInfo message with evacueeID = 1, enteringOrLeaving = T, heloID = 1, triage\_status = ‘W’

At time 10: EvacInfo message with evacueeID = 2, enteringOrLeaving = T, heloID = 1, triage\_status = ‘W’

2)

Test Input:

At time 0: EvacInfo message with evacueeID = 1, enteringOrLeaving = T, heloID = 3, triage\_status = ‘W’

At time 10: EvacInfo message with evacueeID = 2, enteringOrLeaving = T, heloID = 2, triage\_status = ‘W’

**Helicopter Tests**

Initial Conditions: Location is FOL, stop = false

Test Input:

time 176, inES receives HeloInfo with stop = False

time 191, inES receives HeloInfo with stop = False

time 191, inEvac receives EvacInfo

time 510, inES receives HeloInfo with stop = True

2)

Test Input:

At time 165, receives Boolean from inES that is True

**Helicopter Coupled Tests**

Initial Conditions:

FilterES with heloID = 1

filterEvac with heloID = 1

Helicopter with location in FOL, stop = false, and heloID = 1

1)

Test Input:

time 176, HelicopterCoupled.inES receives HeloInfo with stop = False

time 191, HelicopterCoupled.inES receives HeloInfo with stop = False

time 191, HelicopterCoupled.inEvac receives EvacInfo

time 510, HelicopterCoupled.inES receives HeloInfo with stop = True

2)

Test Input:

At time 165, HelicopterCoupled.inES receives Boolean from inHM that is True

**Helicopter Queue Tests**

Initial Conditions: Queue Empty, needHelo = false, waitingForHelo = true, stop = false

1)

Test Input:

At time 0, receives Boolean from inHM that is False

At time 165, receives HeloInfo from inHelo

At time 165, receives HeloInfo from inHelo

At time 180, receives Boolean from inHM that is True

2)

Test Input:

At time 0, receives Boolean from inHM that is True

**Helipad Manager Tests**

Initial Conditions: heloID = -1, requesting = true, heloLeaving = false, stop = false

1)

Test Input:

At time 0, inEM receives Boolean that is True

2)

Test Input:  
At time 0: inHQ receives HeloInfo

At time 20: inHQ receives HeloInfo

At time 35: inEM receives Boolean that is True

**Evacuee Manager Tests**

Initial Conditions: Has Evacuees, evacueesLoading = false, stop = false, heloID = -1

1)

Test Input:

At time 0, inEvac receives EvacInfo

At time 10, inHM receives HeloInfo

At time 20, inEvac receives EvacInfo

At time 35, inHM receives HeloInfo

2)

Test Input:

At time 0, inEvac receives EvacInfo

At time 5, inEvac receives EvacInfo

At time 5, inEvac receives EvacInfo

At time 5, inEvac receives EvacInfo

At time 10, inHM receives HeloInfo

**Evacuation Site Tests**

Initial Conditions:

Helicopter Queue with Queue Empty, needHelo = false, waitingForHelo = true, stop = false

Helipad Manager with heloID = -1, requesting = true, heloLeaving = false, stop = false

Evacuee Manager with Has Evacuees, evacueesLoading = false, stop = false, heloID = -1

1)

Test Input:

At time 0, EvacuationSite.inEvac receives EvacInfo

At time 5, EvacuationSite.inEvac receives EvacInfo

At time 165, EvacuationSite.inHelo receives HeloInfo

At time 165, EvacuationSite.inHelo receives HeloInfo

2)

Test Input

At time 0, EvacuationSite.inHelo receives HeloInfo

At time 0, EvacuationSite.inEvac receives EvacInfo

At time 0, EvacuationSite.inEvac receives EvacInfo

At time 0, EvacuationSite.inEvac receives EvacInfo

At time 10, EvacuationSite.inHelo receives HeloInfo

**FOL Tests**

Initial Conditions: No evacuees at FOL

1)

Test Input:

At time 0, in receives EvacInfo

At time 5, in receives EvacInfo

2)

Test Input:

At time 0, in receives EvacInfo

At time 300, in receives EvacInfo

**Integration Tests**

As there are no inputs to the top model the change in how the models are initialized will be the changing variable

1)

Initial Conditions:

Helicopter Queue with Queue Empty, needHelo = false, waitingForHelo = true, stop = false

Helipad Manager with heloID = -1, requesting = true, heloLeaving = false, stop = false

Evacuee Manager with Has Evacuees, evacueesLoading = false, stop = false, heloID = -1

FilterES with heloID = 1

filterEvac with heloID = 1

Helicopter with location in FOL, stop = false, and heloID = 1

FilterLoc with evacueeID = 1

Evacuee with evacueeID = 1, triage\_status is White, curr\_loc is Evacuation Site, heloID is -1

Expected Result:

Evacuee should be transported to the FOL at time 330

2)

Initial Conditions:

Helicopter Queue with Queue Empty, needHelo = false, waitingForHelo = true, stop = false

Helipad Manager with heloID = -1, requesting = true, heloLeaving = false, stop = false

Evacuee Manager with Has Evacuees, evacueesLoading = false, stop = false, heloID = -1

FilterES with heloID = 1

filterEvac with heloID = 1

Helicopter with location in FOL, stop = false, and heloID = 1

FilterLoc with evacueeID = 1

Evacuee with evacueeID = 1, triage\_status is Red, curr\_loc is Evacuation Site, heloID is -1

Expected Result:

Evacuee should be dead within time 90 on average, simulation should then end as it is the only evacuee all other models should passivate.

**Part 3) Top Model Simulations**

**Simulation 1**

Initial Conditions:

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Values at S0 |
| *m* | Vector of mean time (hour) for individual to deteriorate from one triage category to another. is the mean transition time from the white to green category. | [120,48,8,1.5] |
| *C* | Total capacity for individuals onboard a helicopter. | 10 |
|  | Vector of capacity consumed by each triage category onboard a helicopter. | [1,1,3,3] |
|  | Total time for helicopter to load individuals, transport them to FOL, unload, and return to evacuation site. | 3 h |
|  | Time for helicopter to travel between ES and FOL | 2.5 h |
|  | Time to unload/load the helicopter | 0.25 h |
| *H* | Number of helicopters | 2 |
| *e* | Vector of evacuees in each triage category {white, green, yellow, red, black} | [10,10,5,5,1] |

Evacuees always start at the Evacuation Site, and Helicopters always start as if they just arrived at the FOL. The Helipad Manager is waiting for a helicopter, the Helicopter Queue is empty, the Evacuee Manager is expecting a helicopter, and the FOL has no evacuees yet. The top model has no inputs or outputs.

Execution Analysis:

At time 0 we can see the simulation was initialized correctly as the state of each model is shown to be as I expected it to be in the logger file.

At time 0, Evacuee Manager has White: 10, Green: 10, Yellow: 5, Red: 5, Black: 1

At time 0, HelicopterQueue has no helos waiting to land

At time 0, HelipadManager has no helos at the Evacuation Site

A white grid with black text

Description automatically generated

Then the helipad manager acted correctly as at time 0 it sends a message to the HelicopterQueue requesting a helicopter, but none are sent as the HelicopterQueue is empty.

A screenshot of a computer

Description automatically generated

Before helicopters can arrive at the Evacuation Site some evacuees’ triage status deteriorates as can be seen by messages sent by evacuees to the Evacuation Site. The Evacuation Site responds correctly as the evacuee manager records the changes in their triage status. The time of transition is also being randomly generated correctly as this early on we have 5 people in the Red category deteriorating to the Black and 2 people in the Yellow category deteriorating to the Red, but none from the Green or White which have on average a much longer time to wait before their triage status deteriorates.

A screenshot of a computer

Description automatically generatedThe helicopters leave the FOL at time 15 like they are supposed to, and arrive 150 minutes later at the Evacuation Site, joining the Helicopter Queue.

At time 165, Helicopter Queue has two helos waiting with ID’s 1 and 2

A white grid with black text

Description automatically generated

The Helicopter Queue sends a message to the helo to land and a message to the Helipad Manager with the helicopter’s ID.

A white grid with black text

Description automatically generated

Upon receiving the helo’s ID the Helipad Manager sends the helo’s ID to the evacuee Manager who immediately selects the next group of evacuees to load into the helicopter with that ID. It follows a green first policy where people are selected in order of Green, White, Red, Yellow. It then selects the 10 people in the green triage category. These 10 evacuees are then sent a message with their assigned helo ID, and they each send a message to the helicopter with that ID saying they are entering the helicopter. The helicopter then registers that it has a full 10 capacity taken up by these people.



A white grid with black text

Description automatically generated

A screenshot of a computer

Description automatically generated



After waiting 15 minutes until time 180, for the time to load the evacuees, the Helipad Manager sends a message to the Helicopter indicating it is time to leave for the FOL. The helicopter receives the message and begins travelling.





At the same time the Helipad manager sends a message to the helicopter Queue to request another helicopter. The helo with ID 2 is waiting in the queue so its details are sent to the Helipad Manager and a message to land is sent to the helicopter. The Helipad Manager upon receiving the helo’s ID sends the ID to the evacuee Manager who immediately selects the next group of people. As there are now no people in the Green it adds people from the White category of which there are ten. Loading the helicopter follows the same process as before and the helicopter begins travelling to the FOL at time 195 minutes.

A white grid with black text

Description automatically generated

The helipad Manager immediately sends a message to the Helicopter Queue requesting another helicopter, but the queue is empty so none are sent to the Helipad Manager.

A grid with black text

Description automatically generated

At time 315.957 another evacuee has deteriorated from the Red to the Black. The Evacuee Manager now has 0 evacuees in White, 0 in Green, 3 in Yellow, 1 in Red, and 7 in Black.

A screenshot of a cell phone

Description automatically generated

At time 330 the Helicopter with ID 1 arrives at the FOL and sends a message to each evacuee that it is time to leave the helicopter and go to the FOL. The FOL now lists 10 people in the Green triage category as having arrived.

A screenshot of a computer

Description automatically generated

At time 345 the helicopter with ID 2 arrives at the FOL and does the same as helicopter with ID 1. Now the FOL lists 10 evacuees in the Green, and 10 in the White as having arrived at the FOL. At this time the helicopter with ID 1 has spent 15 minutes at the FOL and so takes off for the Evacuation Site.

A screenshot of a computer

Description automatically generated

At time 360 the helicopter with 2 has spent 15 minutes at the FOL and now takes off for the evacuation site.



While waiting for the second round of helicopters 1 person deteriorates to the Black category and 1 to the Red. At times 439.097 and 444.894 respectively.

A screenshot of a table

Description automatically generated

At time 495 helicopter with ID 1 arrives at the Evacuation Site and joins the Helicopter Queue. The Helicopter Queue immediately sends a message to the helicopter telling them to land and another to the Helipad Manager with the helicopter’s ID. The Helipad Manager sends the helo’s ID to the evacuee Manager who then selects one person from the Red, and 2 people from the Yellow categories to be loaded into the Helicopter. The evacuee Manager sends messages to each evacuee with the helo’s ID, and they send their messages to the helo saying they are entering. As there are no more living evacuees the Evacuee Manager sends a boolen with value True to the Helipad Manager indicating it’s time to stop. The Helipad Manager sends a message to the Helicopter Queue with Boolean True to indicate it’s time to stop. It also immediately sent a message to the helicopter telling it to leave and with a Boolean with value True so it does not return to the Evacuation Site. This is a bug as it should still take 15 minutes for the helicopter to load the evacuees. I fixed this by changing the internal transition and output functions so that it still waits the 15 minutes before sending the helicopter a message.

A white grid with black text

Description automatically generated

A white sheet with black text

Description automatically generated

A white grid with black text

Description automatically generated

At time 510 the helicopter with ID 2 arrives at the Evacuation Site and joins the Queue. The Helicopter Queue sends a message to the helicopter with Boolean True and it immediately starts returning to the FOL.







At time 645 the helicopter with ID 1 arrives at the FOL and sends messages to each evacuee on board telling them to disembark to the FOL. The helicopter then passivates.

A screenshot of a computer

Description automatically generated

At time 660 the helicopter with ID 2 arrives at the FOL and has no one to unload so it passivates. The simulation then ends as all of the models have become passive. The FOL ends with 23 evacuees with 10 in the White, 10 in the Green, 2 in the Yellow, and 1 in the Red. The evacuee Manager ends the simulation with only 8 people in the Black category. It can then be seen that helicopters will not try to evacuate people in the Black category.



You can see the csv logger file for this simulation in the simulation\_results folder as top\_model1.csv.

**Simulation 2**

Initial Conditions:

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Values at S0 |
| *m* | Vector of mean time (hour) for individual to deteriorate from one triage category to another. is the mean transition time from the white to green category. | [120,48,8,1.5] |
| *C* | Total capacity for individuals onboard a helicopter. | 10 |
|  | Vector of capacity consumed by each triage category onboard a helicopter. | [1,1,3,3] |
|  | Total time for helicopter to load individuals, transport them to FOL, unload, and return to evacuation site. | 3 h |
|  | Time for helicopter to travel between ES and FOL | 2.5 h |
|  | Time to unload/load the helicopter | 0.25 h |
| *H* | Number of helicopters | 5 |
| *e* | Vector of evacuees in each triage category {white, green, yellow, red, black} | [0,0,31,0,0] |

In simulation 1 it was mainly those in the Green and White categories that survived. This is because their mean time to deteriorate is much longer than for those in the Red and Yellow categories and the Evacuee Manager prioritizes people in those categories. For simulation 2 then I want to see how many people are evacuated if we have the same number of people except, they’re all in the Yellow category and there are 3 more helicopters. I predict the people in the Yello and Red categories will have a better outcome than in simulation 1.

Execution Analysis:

At time 0 the initialization is correct with 31 people in the Yellow category and all of the helicopters spending 15 minutes at the FOL. The Helipad Manager does the same as before and sends a message to the Helicopter Queue.

A white grid with black text

Description automatically generated



Before the helicopters can arrive at time 165, 13 people have deteriorated to the Red category, 4 of which deteriorated further to the Black category. This leaves 18 people in the Yellow, 9 in the Red, and 4 in the Black.



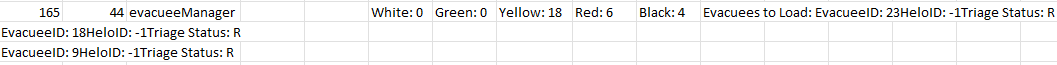
At time 165 the 5 helicopters arrive and enter the queue. Helicopter with ID 1 is sent to the Helipad Manager and the Helipad Manager sends the helo’s ID to the Evacuee Manager. The Evacuee Manager then selects 3 people from the Red category to be loaded into the helicopter as they each require a stretcher they take up 3 spaces each and the helicopter has a capacity of 10. Therefore only 3 people in the Red category can fit at a time. The messages are sent as they were in simulation 1 and the helicopter with ID 1 reports 9 spots taken.



A screenshot of a computer

Description automatically generated





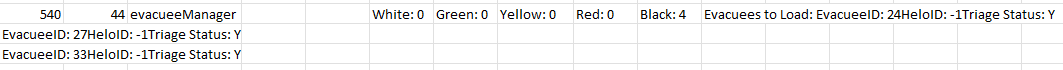
A screenshot of a phone

Description automatically generated



The simulation executes as it did before, so to avoid repeating anything I’ll just elaborate on anything different that occurs.

At time 540 all living evacuees have been loaded into helicopters with only 4 people in the Black category this time. I noticed the same bug as in simulation 1 occurred again, and I see how my attempt at fixing it missed something. I was able to make the change and will run another test to make sure it is fixed. The Helicopter Queue reacts to incoming helicopters correctly when it has received the message to stop, as it sends a message to the new helicopters telling them to stop as well.



A white grid with black text

Description automatically generated

At time 990 the simulation ends as all the models have passivated. The helicopters all returned to the FOL, the helicopter queue is empty, the helipad manager does not have a helicopter, and there are no living evacuees left at the Evacuation Site. The FOL has 10 evacuees in the Yellow category and 17 evacuess in the Red category.

My prediction was correct, as without any people in the Green and White categories those in the Yellow and Red were prioritized. And despite the evacuees taking up 3 times the space in a helicopter as those in the Green and White categories the 3 additional helicopters made up for that and was able to evacuate the majority of them. The csv logger file can be found in the simulation\_results folder as top\_model2.csv.

After running another simulation I can see that the bug where the Helipad Manager was not waiting 15 minutes to tell the helicopter to take off after receiving the stop signal True has now been fixed.