Bushfire JACK Design

1. Shelters, Routes and Evacuation Start Time

Shelters

Shelters (also called relief centres) are the evacuation centres to where the residents should move after receiving evacuation notification. Users can predefine shelters with their name, coordinates and capacity in geography.xml configuration file. If there is no viable shelter to a region, a point which is in the perpendicular direction to the fire progress direction is selected as the shelter for the region.

Routes

Route is a list of waypoint to pass when a resident evacuates to a shelter. Each route should contain at least one point, which is the shelter location. There can be multiple routes from a region to a shelter.

Users can define the routes in geography.xml configuration file. Each defined routes should have a name, starting region, destination shelter, list of waypoints and optionally a description. The description can be either "SAFE", "FAST" or "CUSTOM".

Evacuation Start Time

Evacuation start time is not a configurable parameter and it can be different from region to region.

2. Agents

EvacController

EvacController agent selects shelters, routes and evacuation start time for all regions. There is only one EvacController agent throughout the application.

Bushfire application triggers a fire alert to EvacController when the fire is started. Next, the EvacController starts reasoning to select shelters, routes and evacuation start time for regions. While reasoning, if the visualiser is active, EvacController calls visualiser to get inputs from user. After finalising the evacuation start time, the EvacController waits until the evacuation start time of each region is reached. Then, the agent notifies all BasicResidents in the particular region to evacuate.

BasicResident

BasicResident agent represents the residents live in regions. These agents can have kids and relations. BasicResident agents are triggered by the EvacController agent when the resident's home region should be evacuated.

After receiving the evacuation notification, Basic Residents who have kids and/or relations first pick them before evacuating to the allocated relief centre.

3. Bushfire Application and JACK model Integration

Execution Flow

1. Bushfire application creates all BasicResident agents to map with the already created MATSim agents.
2. Bushfire application initiates the EvacController agent.
3. At each time step, MATSim model publishes “matsim_agent_updates” to BushfireApplication. These update messages contain basic resident agents’ locations, statuses etc.
4. When the bushfire application receives the “matsim_agent_updates” for the first time (at the first time step), it updates BasicResident agents' start location and region according to the information received with the notification. This is called “Late initialisation”. BasicResident agents’ information is updated in this way because this information is unavailable to the bushfire application until after the main loop is initiated. The information is set during the first iteration of the loop.
5. Fire module (FireModule.class) sends fire alert at the fourth time step of the simulation.
6. When the bushfire application receives the fire notification, it sends an alert to EvacController agent.
7. When EvacController agent receives the fire alert, he decides a shelter, a route and evacuation start time for each region. During the reasoning, the EvacController agent calls visualiser if it is active, to get user preferences.
8. Next, EvacController agent waits until evacuation start time of each region. When the evacuation start time of a region is reached, the agent replans a shelter, a route and evacuation start time if is needed. Else, the agent sends a notification to all BasicResident agents in the region asking to evacuate.
9. The BasicResident agents, who received evacuation alert, start evacuating to the shelter decide by EvacController agent. If the basic resident has kids and/or relatives to pick up, he picks them up before evacuating to the shelter.

Sequence Diagram

Figure 1 shows the execution flow of the application using a sequence diagram.

4. Goal Plan Tree of EvacController

Diagram

Figure 2 shows the goal-plan tree of EvacController agent.

BeliefSets

1. Areas

This belief set is used to represent EvacController agent's knowledge about different regions.

Key fields;

- name (java.lang.String): name of the area

Value fields;

- location (com.vividsolutions.jts.geom.Coordinate): coordinates of the centre of the region.
- population (int): population of the region
- viableShelters (java.util.List): names of viable shelters of the area
- timeViable (long): timestamp for when the viableShelters were calculated

2. Shelters

This belief set is used to represent EvacController agent's knowledge about different relief centres (shelters).

Key fields;

- name (java.lang.String): name of the shelter

Value fields;

- location (com.vividsolutions.jts.geom.Coordinate): coordinates of the shelter
- capacity (int): capacity of the shelter
- assignedCapacity (int): the allocated capacity. When a shelter is allocated to an area, a part of shelter's capacity is allocated to the population of the area.

assignedCapcity = sum of (populations of allocated areas)

- remainingCapacity: the remaining capacity of the shelter.

  remainingCapacity = capacity - assignedCapacity

3. Fire

This belief set always contains only one entry about the latest fire polygon and its progress direction.

Key fields;

- id (String): This value is always set to “Fire”.

Value fields;

- fire (java.awt.Polygon): the latest fire polygon
- direction (double): fire progress direction from north in degrees
- time (long): timestamp for when the fire belief set was last updated.

4. Shelter_Vectors

This belief set contains the straight line drew from centre of each region to each shelter.

Key values;

- area (String): name of the area from where the line starts
- shelter (String): name of the shelter at which the line terminates

Value fields;

- path (java.awt.geom.Line2D): line which connects the centre of the area and the shelter.

4. Shelter_Assignments

This belief set contains each region and its assigned shelter.

Key fields;

- area (String): name of the area

Value fields;

- shelter (String): assigned shelter to the area

5. Routes

This belief set contains all routes defined in the geography configuration.

Key fields;

- area (java.lang.String): name of the area from where the route begins
- shelter (java.lang.String): name of the destination shelter of the route
- routeName (java.lang.String): name of the route

Value fields;

- route (java.util.List): coordinates of all waypoints of the route are stores as a List
- waypointNames (java.util.List): names of all waypoints of the route are stores as a List
- desc (java.lang.String): a description about the route (Can be either “FAST”, “SAFE” or “CUSTOM”)
- desc_time (long): timestamp for when the description (desc) of the route was last updated.

6. Route_Assignments

This belief set contains the information about route and the shelter assigned to each region.

Key fields;

- area (java.lang.String): name of the area

Value fields;

- shelter (java.lang.String): name of the assigned shelter
- route (List): coordinates of all waypoints of the route
- waypointNames (List): names of all waypoints of the route

7. Evac_Times

This belief set contains the execution start time assigned to each region.

Key fields;

- area (java.lang.String): name of the area

Value fields;

- time (long): execution start time of the area

8. Plan_Checks

This belief set contains information whether each region's assigned shelter, route and evacuation start time should be replanned or not.

Key fields;

- areaName (java.lang.String): name of the area

Value fields;

- checkOK (boolean): this is false if shelter, route and evacuation start time of the region should be replanned
- timestamp (long): timestamp for when the routes belief set was last updated.

Plans

1. Respond_FireP

Handles: Respond_fireG

Posts : PlanAllAreasG

When EvaController agent receives the fire alert percept from the bushfire application, he posts Respond_fireG event. As a result, the Respond_FireP plan gets executed and all it does is posting PlanAllAreasG event.

2. PlanAreasP

Handles: PlanAllAreasG

Posts : DoScheduleG, Decide_shelterG, Decide_routeG, Calc_viable_sheltersG

Initially, the PlanAreasP event calls a subtask to the event Calc_viable_sheltersG to calculate viable shelters for all regions. Next, it subtasks Decide_shelterG and Decide_routeG one after other for each region. Finally, is posts DoScheduleG event to schedule evacuation event.

3. Calc_viable_sheltersP

Handles: Calc_viable_sheltersG

Posts : Calculate_fire_vectorG, Calculate_sheltersG

First, the Calc_viable_sheltersP plan triggers a sub task to Calculate_fire_vectorG goal to calculate the latest fire vector. Next, for each region, it triggers a subtask to Calculate_sheltersG goal if the region does not have any viable shelters or the existing viable shelters were updated at least 30mins before the application time.

4. Calc_fire_vectorP

Handles: Calculate_fire_vectorG

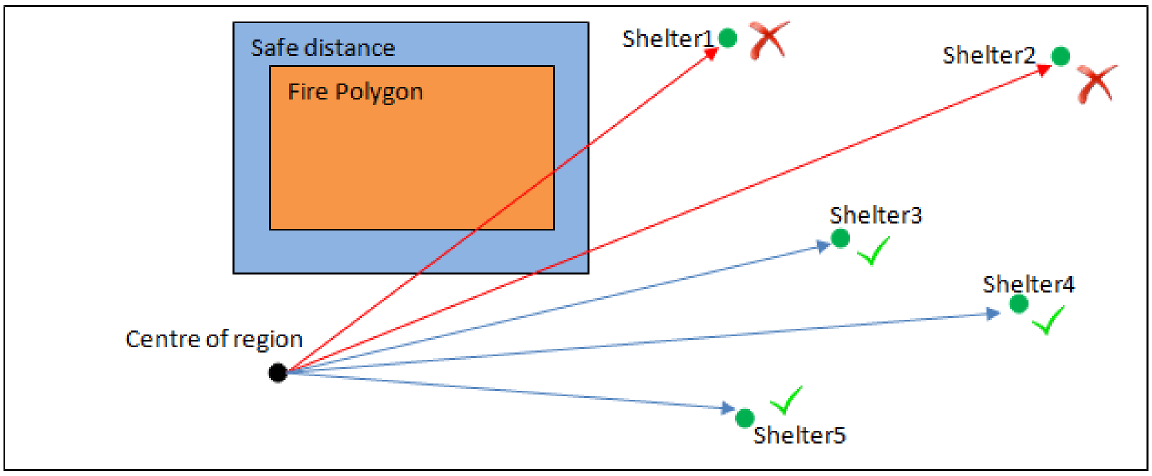
Gets the time stamp when did the fire belief of EvacController agent was last updated. If the time stamp is old, then update agent's fire belief with latest fire polygon and direction information read from FireModule.

5. Calc_sheltersP

Handles: Calculate_sheltersG

An instance of this plan is executed for each region separately.

The plan iterates through all defined shelters and gets the vector, which connects the centre of the region and each viable shelter from the "Shelter_Vectors" belief. Next, checks whether the vector intersects the fire polygon or the vector is closer to the fire polygon than safe distance. If not, the shelter is assigned as a viable shelter of the region.



6. Choose_by_capacityP

Handles: Decide_shelterG

This is one of the four plans, which handle the event Decide_shelterG.

The Choose_by_capacityP plan becomes applicable only if there is more than one viable shelter assigned for the area.

During the plan, the viable shelter of the region which has the greatest remaining capacity is selected. Next, the selected shelter's available capacity is reduced by the population of the region and a new entry is added to “Shelter_Assignments” belief set.

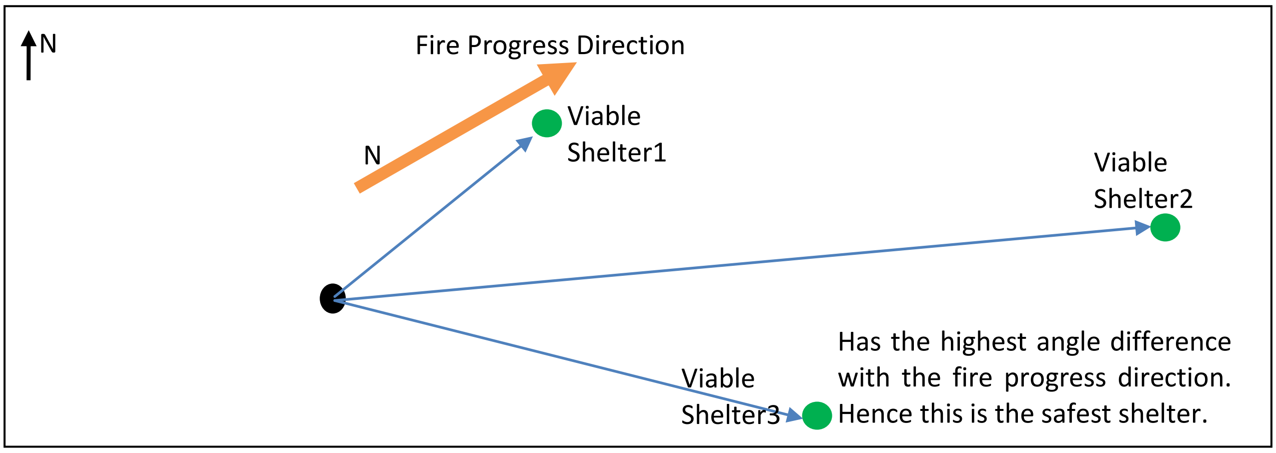
7. Choose_safestP

Handles: Decide_shelterG

This is one of the four plans, which handle the event Decide_shelterG.

The Choose_safestP plan becomes applicable only if there is more than one viable shelter assigned for the area.

Firstly, the angle from north of the vector, which connects the centre of fire polygon and viable shelter of the region is calculated. The shelter which has the greatest angle difference from the fire progress direction is considered as the safest shelter. Next, the selected shelter's available capacity is reduced by the population of the region and a new entry is added to “Shelter_Assignments” belief set.


N
Fire Progress Direction
N
Viable Shelter1
Viable Shelter2
Viable Shelter3
Has the highest angle difference with the fire progress direction. Hence this is the safest shelter.

8. Choose_onlyP

Handles: Decide_shelterG

This is one of the four plans, which handle the event Decide_shelterG.

The Choose_onlyP plan becomes applicable only if there is one viable shelter assigned for the area.

During the plan, the only viable shelter of the region is selected as the shelter of the area. Next, the shelter's available capacity is reduced by the population of the region and a new entry is added to “Shelter_Assignments” belief set.

9. No_sheltersP

Handles: Decide_shelterG

This is one of the four plans, which handle the event Decide_shelterG.

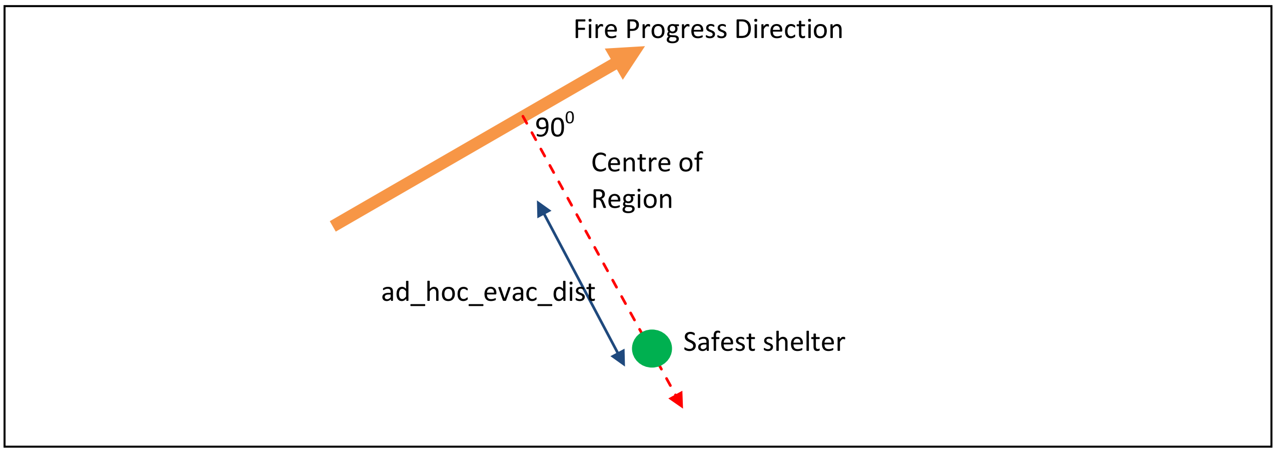
The No_sheltersP plan becomes applicable only if there are no viable shelters assigned for the area.

During the plan, a new location, which

- lies in a perpendicular direction from the fire progress direction
- is “ad_hoc_evac_dist” away from the centre of the region

is selected as the shelter.

Finally, a new entry is added to “Shelter_Assignments” belief set.



10. Fast_routeP

Handles: Decide_routeG

This is one of the four plans, which handles the event Decide_routeG.

The Fast_routeP plan becomes applicable only if there is a route configured from the region to the assigned shelter with the description “FAST”. The route with the description “FAST” is assigned as the route for the region and it is updated to the Route_Assignments belief set.

11. Safe_routeP

Handles: Decide_routeG

This is one of the four plans, which handles the event Decide_routeG.

The Safe_routeP plan becomes applicable only if there is a route configured from the region to the assigned shelter with the description “SAFE”. The route with the description “SAFE” is assigned as the route for the region and it is updated to the Route_Assignments belief set.

12. Custom_routeP

Handles: Decide_routeG

This is one of the four plans, which handles the event Decide_routeG.

The Safe_routeP plan becomes applicable only if there is a route configured from the region to the assigned shelter with the description “CUSTOM”. The route with the description “CUSTOM” is assigned as the route and it is updated to the Route_Assignments belief set. This gives users the space to assign customized routes by changing EvacControllers Belief's about Routes at runtime.

13. Free_choiceP

This is one of the four plans, which handles the event Decide_routeG.

The Free_choiceP plan is always applicable. In the plan, a new route is created without any waypoints. The idea is to let MATSim to select any route to reach the destination shelter without any restrictions from BDI side.

14. DoScheduleP

Handles: DoScheduleG

Posts: Decide_timeG

As the final step, the EvacController agent moves to this plan after deciding shelters and routes for all regions. The plan DoScheduleP iterates through all regions and calls a subtask to Decide_timeG instances created for each region separately.

15. Decide_timeP

Handles: Decide_timeG

If there’s nothing set in Evac_Times belief set for the region, a random value from “now”, “now+30minutes”, “now + 60 minutes” or “now + 90minutes” is selected as the evacuation time of the region. So, the user can manually set evacuation time by changing the “Evac_Times” belief set manually through the visualiser. When this step finishes, shelters, routes and evacuation start time are selected for the region. So, the only thing left is to send evacuation alert to residents.

16. Evac_timeP

Handles: Evac_timeG

Posts: CheckplanG, Replan_areaG

Sends: EvacMessage

Evac_timeG goal is triggered by EvacController agent when the application reaches the evacuation start time of each region. As a result the Evac_timeP plan is executed.

Initially, the plan calls the he CheckplanG goal as a subtask. Next, plan checks what is the value in the Plan_Checks belief set for the area. If the “CheckOK” value is “false”, the replanning is done by calling subtasks to;

- Decide_shelterG
- Decide_routeG
- Decide_timeG

events in the above order.

Else, if “CheckOK” is true, the plan sends the event “EvacMessage” to BasicResident agents in the region.

17. CheckPlanP

Handles: CheckplanG

This plan is used to update Plan_Checks belief set. If there are no entries in the belief set about the area or if the time stamp of the entry is old, updates the belief set with “CheckOK=true” value for the area.

Implementation of this plan should be improved to consider changes in fire progression to decide Plan_Checks.

18. Replan_areaP

Handles: Replan_areaG

Posts: Calculate_fire_vectorG, p_Calculate_sheltersG, Decide_shelterG, Decide_routeG, p_Decide_timeG.

The goal Replan_areaG is called only if Plan_Checks belief set contains a false value for “CheckOK” for the region.

During the plan, following events are called as subtask in below order to replan the area;

- Calculate_fire_vectorG
- p_Calculate_sheltersG
- Decide_shelterG,
- Decide_routeG
- p_Decide_timeG

Meta-Level Plans

1. Deside_shelter_metaplan

Chooses for: Decide_shelterG

There are four plans for the event Decide_shelterG. If more than one plan becomes applicable, then the Deside_shelter_metaplan is called to do the reasoning to select a single plan for the goal Decide_shelterG.

During the plan, if the visualiser is active, the goal, the applicable plans and their information are sent to visualiser to allow user to select a plan. Else, if the visualiser is not active, the meta-level plan Deside_shelter_metaplan itself selects one applicable plan randomly.

2. Deside_route_metaplan

Chooses for: Decide_routeG

There are four plans for the event Decide_routeG. If more than one plan becomes applicable, then the Deside_route_metaplan is called to do the reasoning to select a single plan for the goal Decide_routeG.

During the plan, if the visualiser is active, goal and the applicable plans and their information are sent to visualiser to allow user to select a plan. Else if the visualiser is not active, the meta-level plan Deside_route_metaplan itself selects one applicable plan randomly.

5. Goal Plan Tree of BasicResident

Diagram

The figure 3 shows the goal plan tree of BasicResident agent.

Plans

1. EvacMessageP

Handles: EvacMessage

Posts: EvacHouse

The goal, EvacMessage, is sent by EvacController agent and it is handled by BasicResident agents. Basically, the goal EvacMessage is used to send evacuation alert to BasicResident agents.

The EvacMessageP plan reads the waypoint information sent with EvacMessage and forwards them to EvacHouse event.

2. PckupKids

Handles: EvacHouse

Posts: GenericActGoal

The plan PickupKids becomes applicable only if the BasicResident agent has kids to pick up.

The plan initiates a drive-to action to school and when the agent arrives at the school, it reposts EvacHouse goal.

3. PickupRelatives

Handles: EvacHouse

Posts: GenericActGoal

The plan PickupRelatives becomes applicable only if the BasicResident agent has relatives to pick up.

The plan initiates a drive-to action to a random location and when the agent arrives at the random location, it reposts EvacHouse goal.

4. ImmediateEvac

Handles: EvacHouse

This action becomes applicable if the agent has neither kids nor relatives to pick up.

The plan iterates through all waypoints in the route and initiates drive-to actions to each waypoint one after another until the agent reaches the shelter.

Meta-Level Plans

1. EvacHouse_metaplan

Chooses for: EvacHouse

There are three plans for the goal EvacHouse. If more than one plan becomes applicable, then the EvacHouse_metaplan is called to do the reasoning to select a single plan.

The three plans of the goal EvacHouse are declared in the order;

- PickupKids
- PickupRelatives
- ImmediateEvac

The EvacHouse_metaplan always chooses the first applicable plan from the list. So, if the agent has kids and relatives, he firstly drives to picks up kids and secondly drives to pick up relatives. When the EvacHouse event is triggered for the third time, the agent does not have any kids or relatives to pick up. Hence the meta-level plan is not getting called.

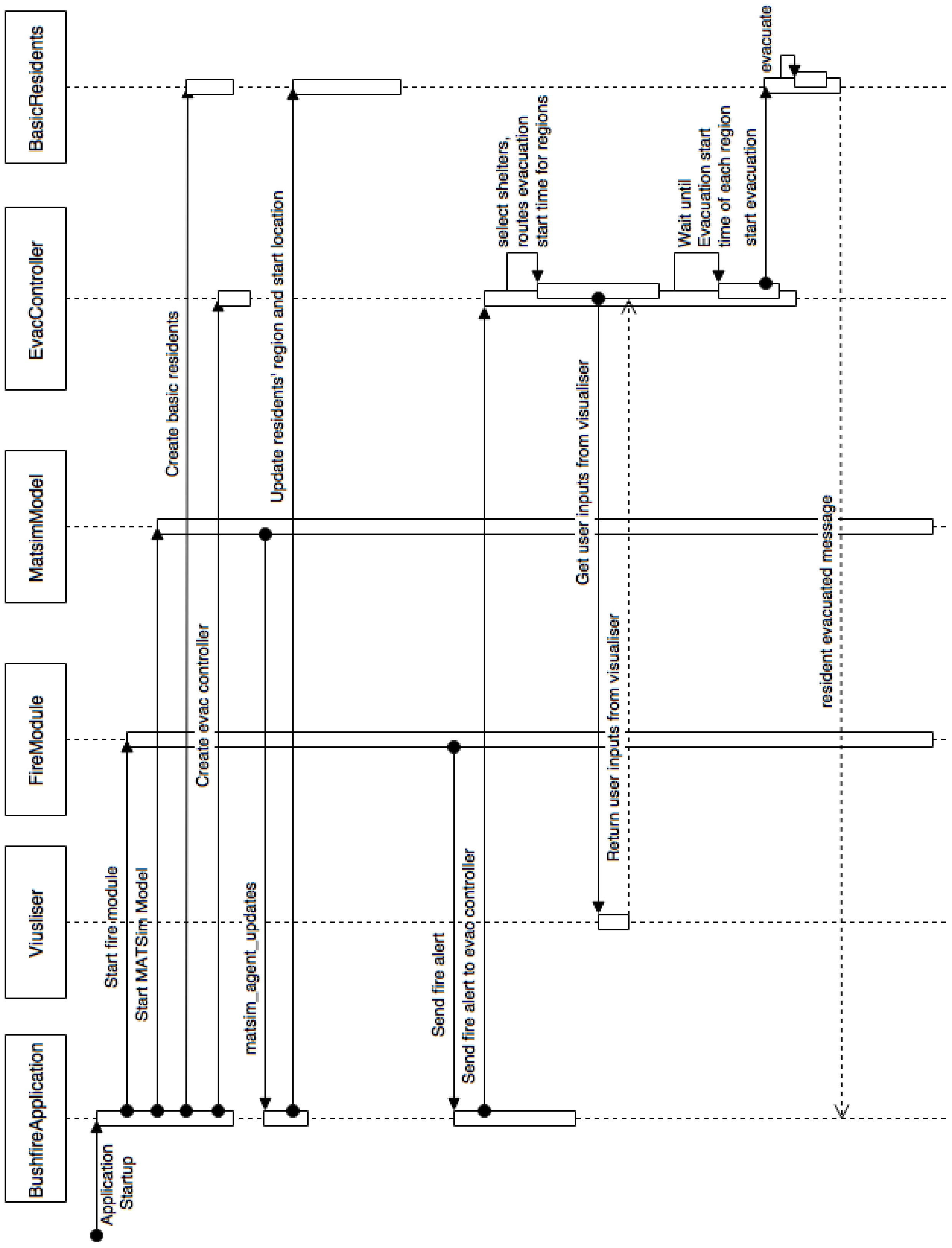

BushfireApplication
Viusliser
FireModule
MatsimModel
EvacController
BasicResidents
Application Startup
Start fire module
Start MATSim Model
Create basic residents
Create evac controller
matsim_agent_updates
Update residents' region and start location
Send fire alert
Send fire alert to evac controller
select shelters, routes evacuation start time for regions
Get user inputs from visualiser
Return user inputs from visualiser
Wait until Evacuation start time of each region
start evacuation
evacuate
resident evacuated message

Figure 1: Sequence Diagram

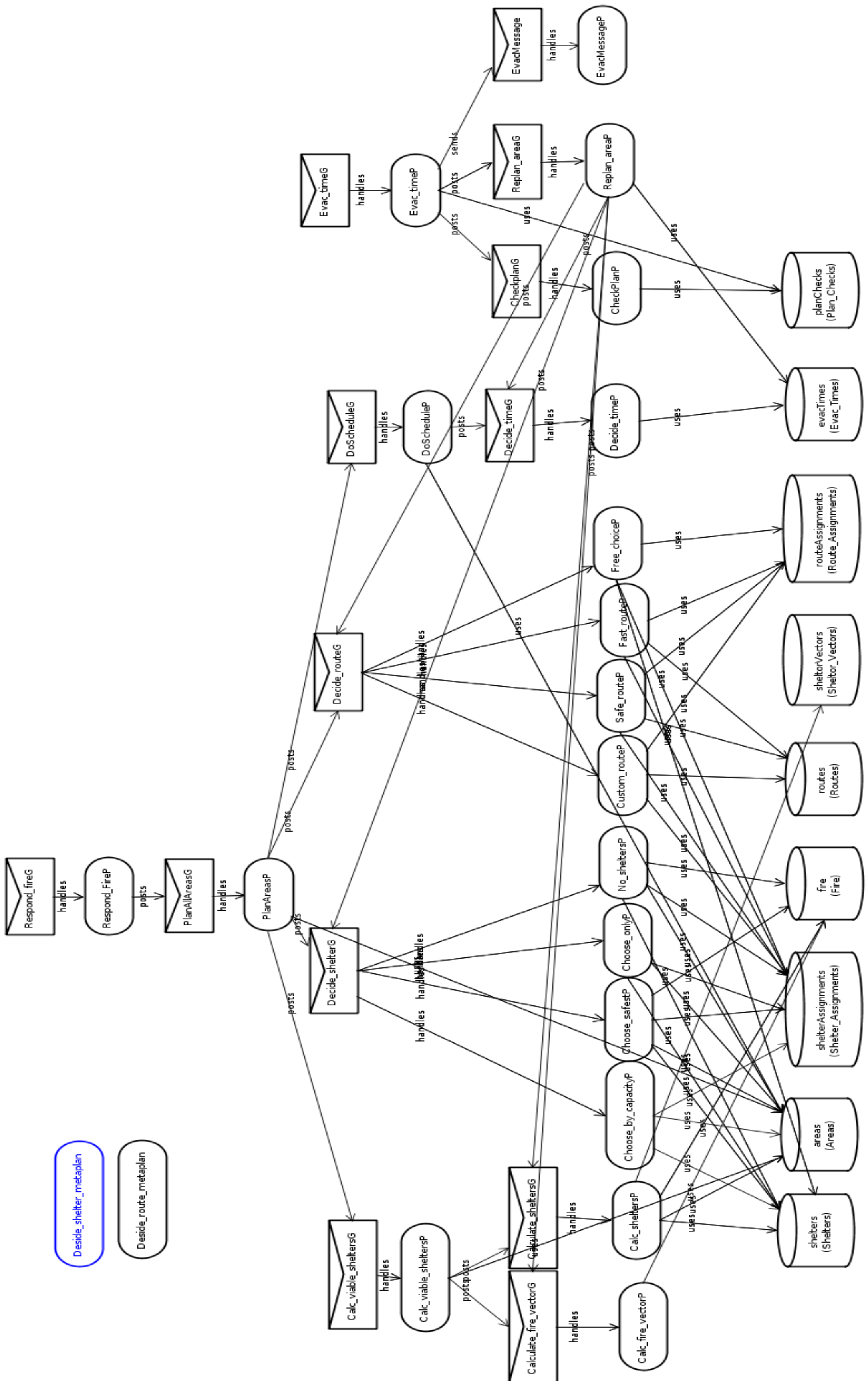


Figure 2: Goal Plan Tree of EvacController Agent

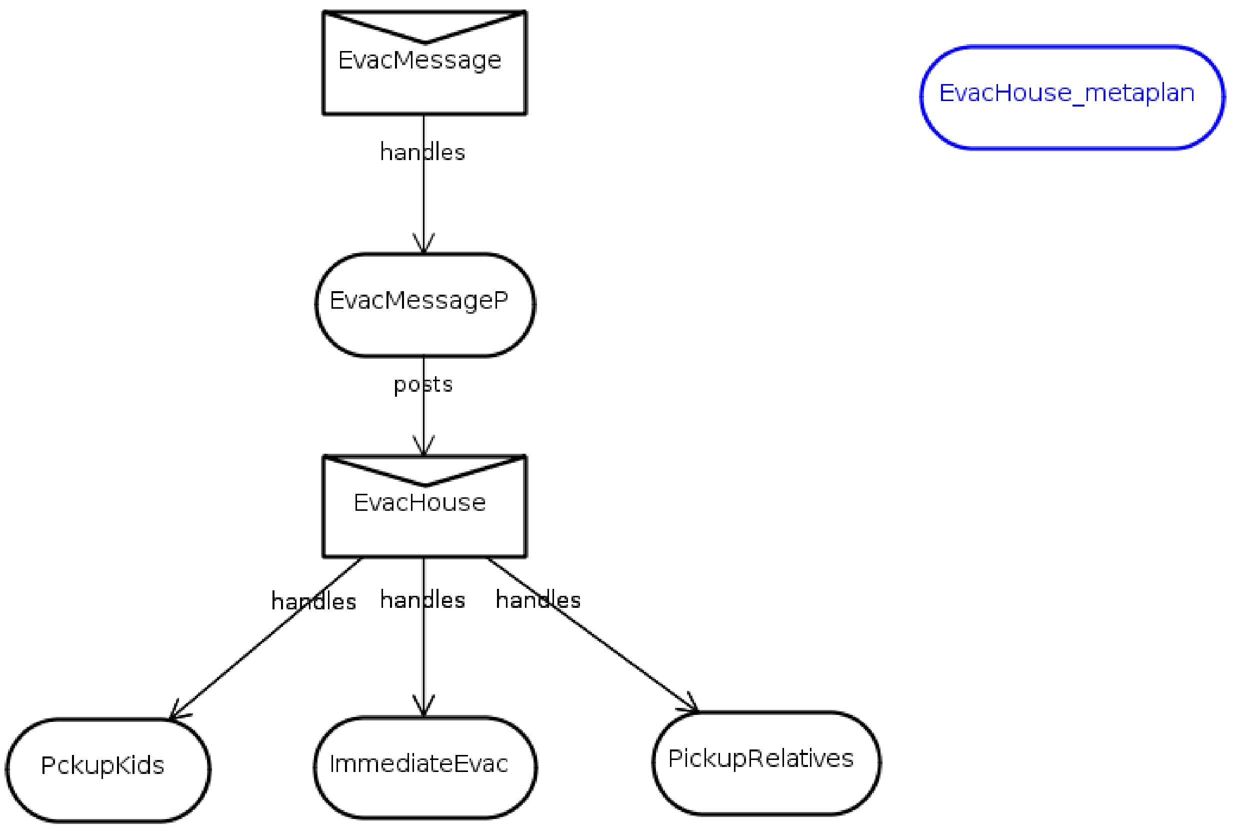


Figure 3: Goal Plan Tree of BasicResident Agent