```

:name: follower

:Initial Beliefs:

:Reasoning Rules:

platoon_m(X, Y):- steering_contr, platoon_m;
leave_platoon:- ~steering_contr, no_platoon_m;
steering_cont(1):- steering_contr;
steering_cont(0):- ~ steering_contr;

:Initial Goals:

:Plans:

    /* Default plans for handling messages */
    +.received(:tell, B): {True} <- +B;
    +.received(:perform, G): {True} <- +!G [perform];
    +.received(:achieve, G): {True} <- +!G [achieve];



+id(X): {X== 1}<- perf(join_ok(1));

+id(X): {X== 2}<- perf(join_ok(1));

+id_front(X): {B id(3)}<- perf(join_position(X));

+ready_to_leave: {B name(follower3)}<- print(ready_to_leave), +! leave_platoon [achieve], print(follower3_drop_leaving_goal_or_successfully_left);


//==========initialisation phase-- vehicle joins to platoon after 50 sec
+platoon_set_up: {True}<- perf(speed_controller(1)), perf(steering_controller(1));

+ready_to_join: {B name(follower3), B name_front(FRONT)} <- print(ready_to_join), +!platoon_m(follower3, FRONT) [achieve], print(follower_3_drop_goal_or_successfully_joined_in_the_middle);

+wrong_front: {G platoon_m(SENDER, FRONT) [achieve]}<- -!
```

```
platoon_m(SENDER, FRONT) [achieve], print(belief_not_able_to_join_to_front_of_a_non_platoon_member);

//+!platoon_m(SENDER, FRONT) [achieve]: {B id(SENDER), ~B platoon_m, B distance(FRONT)} <-
+!platoon_m(SENDER, FRONT) [achieve]: {B name(SENDER), ~B platoon_m, ~B join_agreement(SENDER, FRONT)} <-
                                        .send(leader, :tell, message(SENDER, 1, FRONT)), *join_agreement(SENDER, FRONT);


+!platoon_m(SENDER, FRONT) [achieve]: {B join_agreement(SENDER, FRONT), B name(SENDER), B id(Y), ~B changed_lane} <- perf(join_ok(1)), print(perform_change_lane), *changed_lane;

+!platoon_m(SENDER, FRONT) [achieve]: {B changed_lane, ~B initial_distance} <- perf(speed_controller(1)), +sp_contr, *initial_distance;

+!platoon_m(SENDER, FRONT) [achieve]: {B changed_lane, B initial_distance, ~B steering_contr, ~B sp_contr } <- perf(speed_controller(1)), perf(steering_controller(1)), +steering_contr;

+!platoon_m(SENDER, FRONT) [achieve]: {B sp_contr, B initial_distance} <- perf(steering_controller(1)), -sp_contr, +steering_contr;

+!platoon_m(SENDER, FRONT) [achieve]: {B steering_contr} <- .send(leader, :tell, message(SENDER, 2)), *platoon_m;

+!steering_contr(X) [perform]: {B steering_contr, X==0}<- perf(steering_controller(X)), -steering_contr;

+!steering_contr(X) [perform]: {~B steering_contr, X==1}<- perf(steering_controller(X)), +steering_contr;

+!set_spacing(X) [achieve]: {~B spacing(X)} <- perf(set_spacing(X)), *spacing;

+!set_spacing(X) [achieve]: {B spacing, B name(V)} <- .send(leader, :tell, set_spacing_from(V)), -!set_spacing(X) [achieve];

```

```
// ============ leaving
+!leave_platoon [achieve]: {~G platoon_m(X, Y) [achieve], B name(X), ~B agreement(X)} <- .send(leader, :tell, message(X, 0)), *agreement(X);

//+!leave_platoon [achieve]: {B name(X), B agreement(X), ~B spacing(15)} <- +!set_spacing(15) [achieve];

+!leave_platoon [achieve]: {B name(X), B agreement(X), B steering_contr, B spacing} <- perf(speed_controller(0)), +!steering_contr(0) [perform];

+!leave_platoon [achieve]: {B name(X), B agreement(X), ~B steering_contr, ~B changed_lane} <- perf(join_ok(0)), print(change_lane_to_leave), *changed_lane;

+!leave_platoon [achieve]: {B changed_lane, B name(X)} <- .send(leader, :tell, message(X, 2)), *no_platoon_m;
```