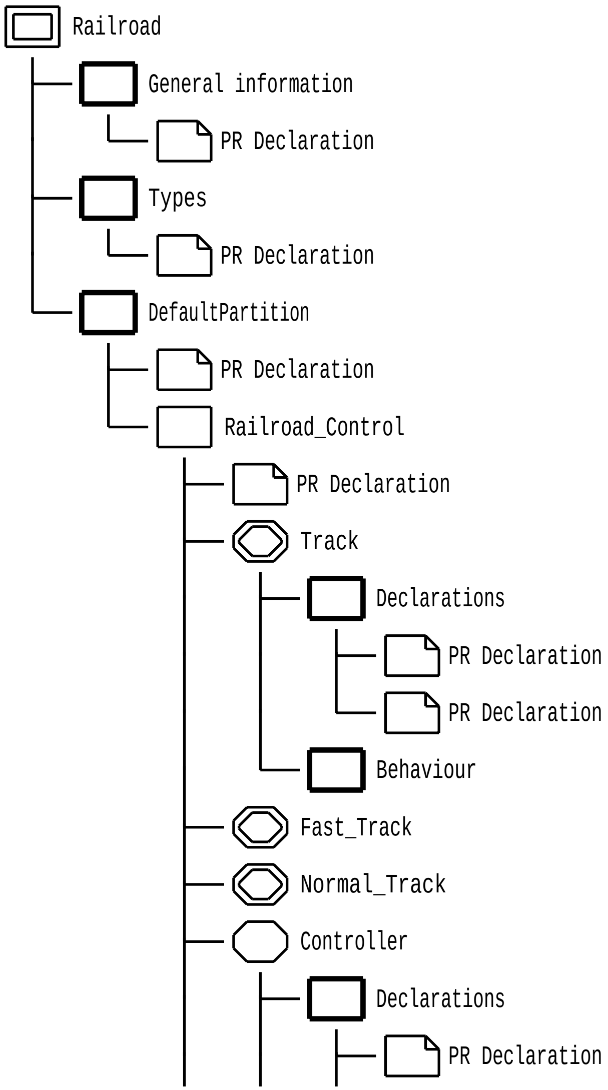
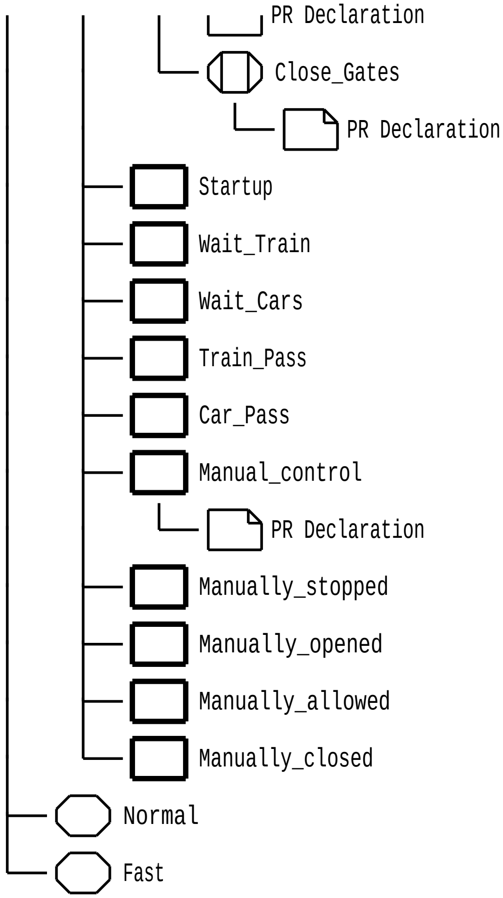

Railroad
General information
PR Declaration
Types
PR Declaration
DefaultPartition
PR Declaration
Railroad_Control
PR Declaration
Track
Declarations
PR Declaration
PR Declaration
Behaviour
Fast_Track
Normal_Track
Controller
Declarations
PR Declaration

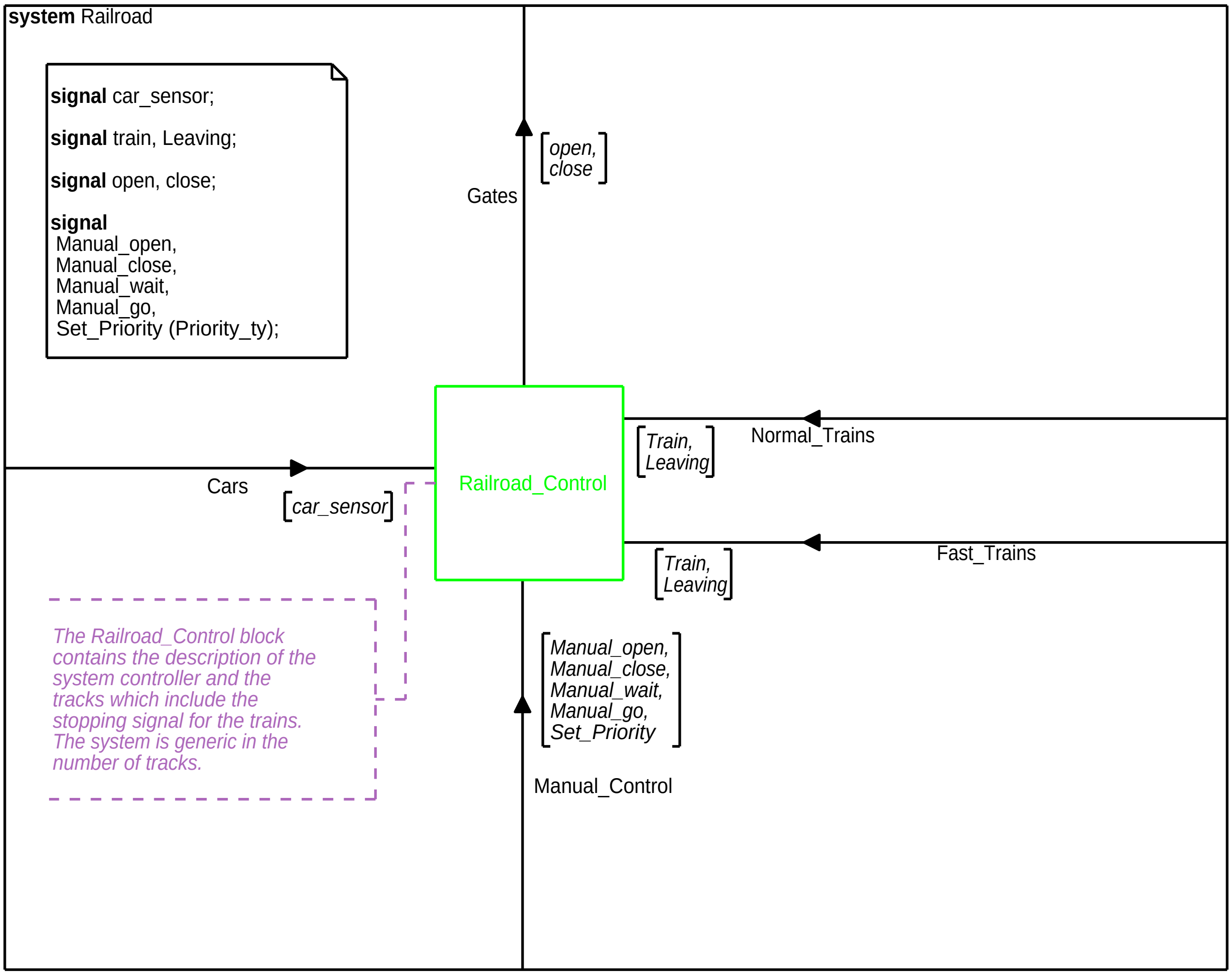

PR Declaration
Close_Gates
PR Declaration
Startup
Wait_Train
Wait_Cars
Train_Pass
Car_Pass
Manual_control
PR Declaration
Manually_stopped
Manually_opened
Manually_allowed
Manually_closed
Normal
Fast

system Railroad

```
/* Railway crossing controller specification
 *
 * Authors :
 *   Eric Conquet - Maxime Perrotin
 *   May 2002
 *
 *   Designed for the SDL Contest of the 3rd SAM Workshop
 *
*/
```

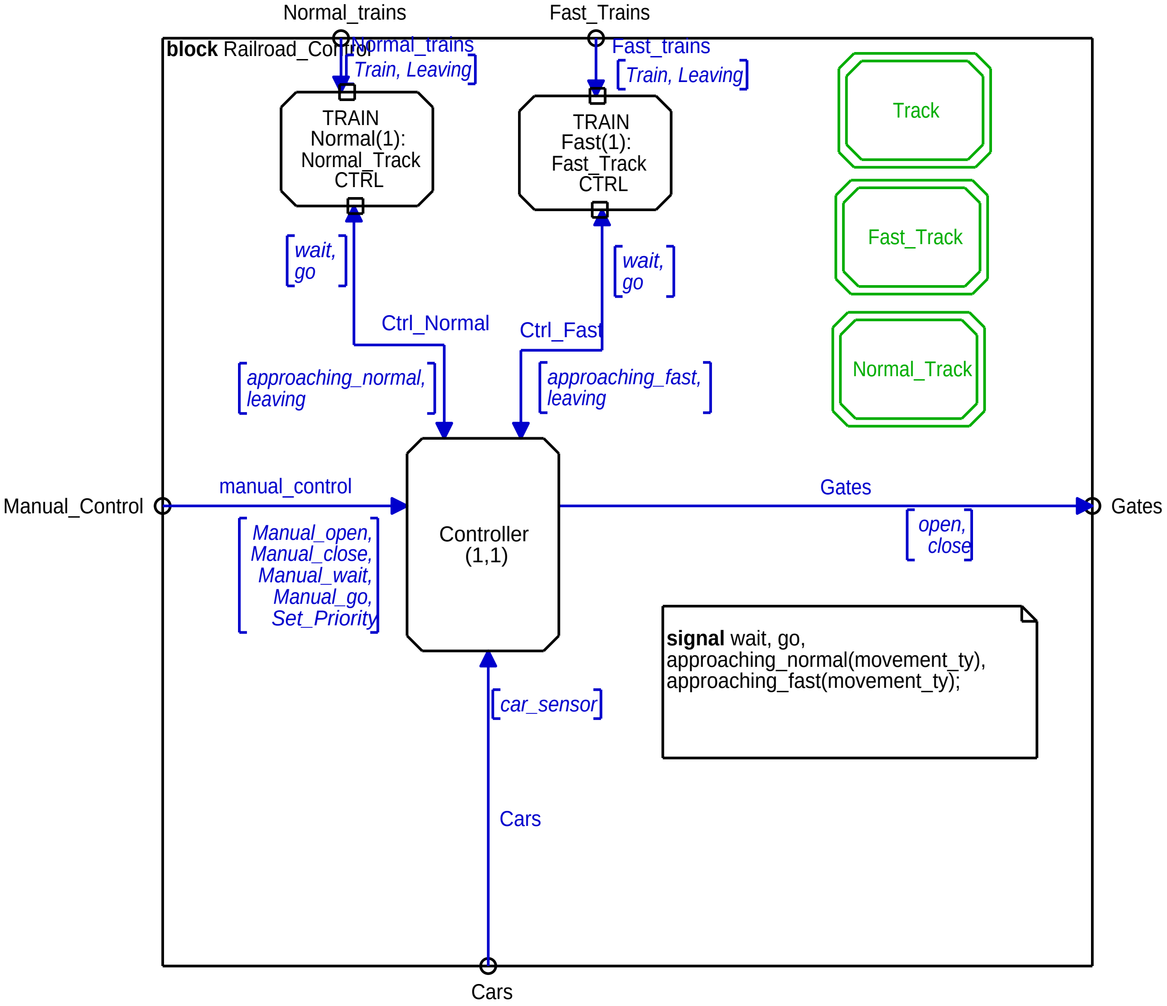
system Railroad

```
movement_ty ::= ENUMERATED
{ waiting, moving };
/* Type "movement_ty" defines if the train that approaches is moving
or is stopped  */

Priority_ty ::= ENUMERATED
{ all_trains, fast_trains, cars, automatic_priority };
/* Defines all the possible priority strategies */

Priority_time duration ::=10;
/* In automatic priority mode, the priority level
will decrease every "priority_time" ticks. */

High_priority natural ::= 3;
/* Defines the highest priority level for the automatic priority mode */

CLOSING_TIME DURATION ::= 5;
/* Closing_Time : constant time needed
for the gates to be closed. */

PASSING_TIME DURATION ::= 10;
/* Time after which it is considered that the entire
train has passed over the sensors */
```



```
signal car_sensor;

signal train, Leaving;

signal open, close;

signal
 Manual_open,
 Manual_close,
 Manual_wait,
 Manual_go,
 Set_Priority (Priority_ty);
```

The Railroad_Control block contains the description of the system controller and the tracks which include the stopping signal for the trains. The system is generic in the number of tracks.

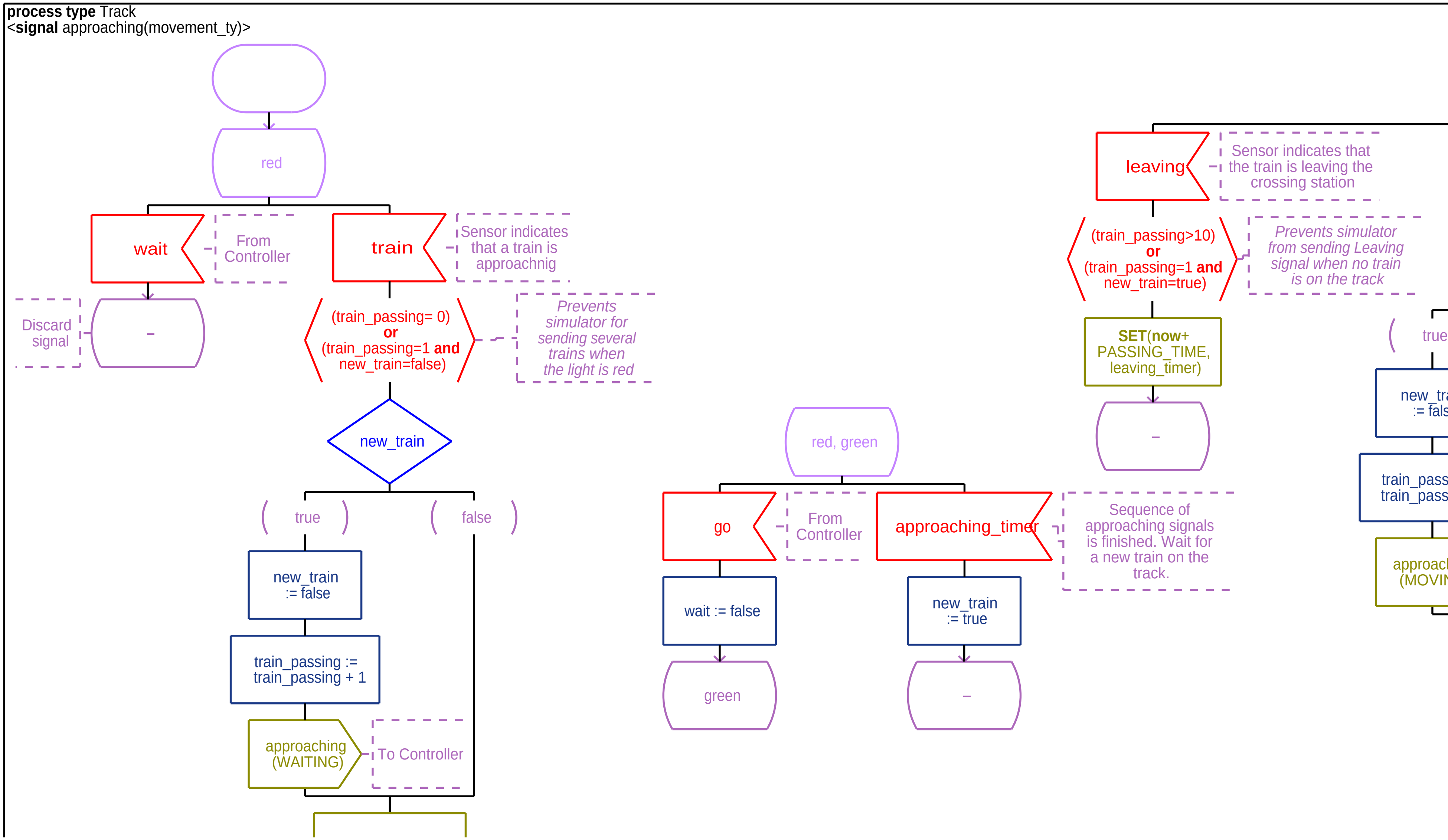


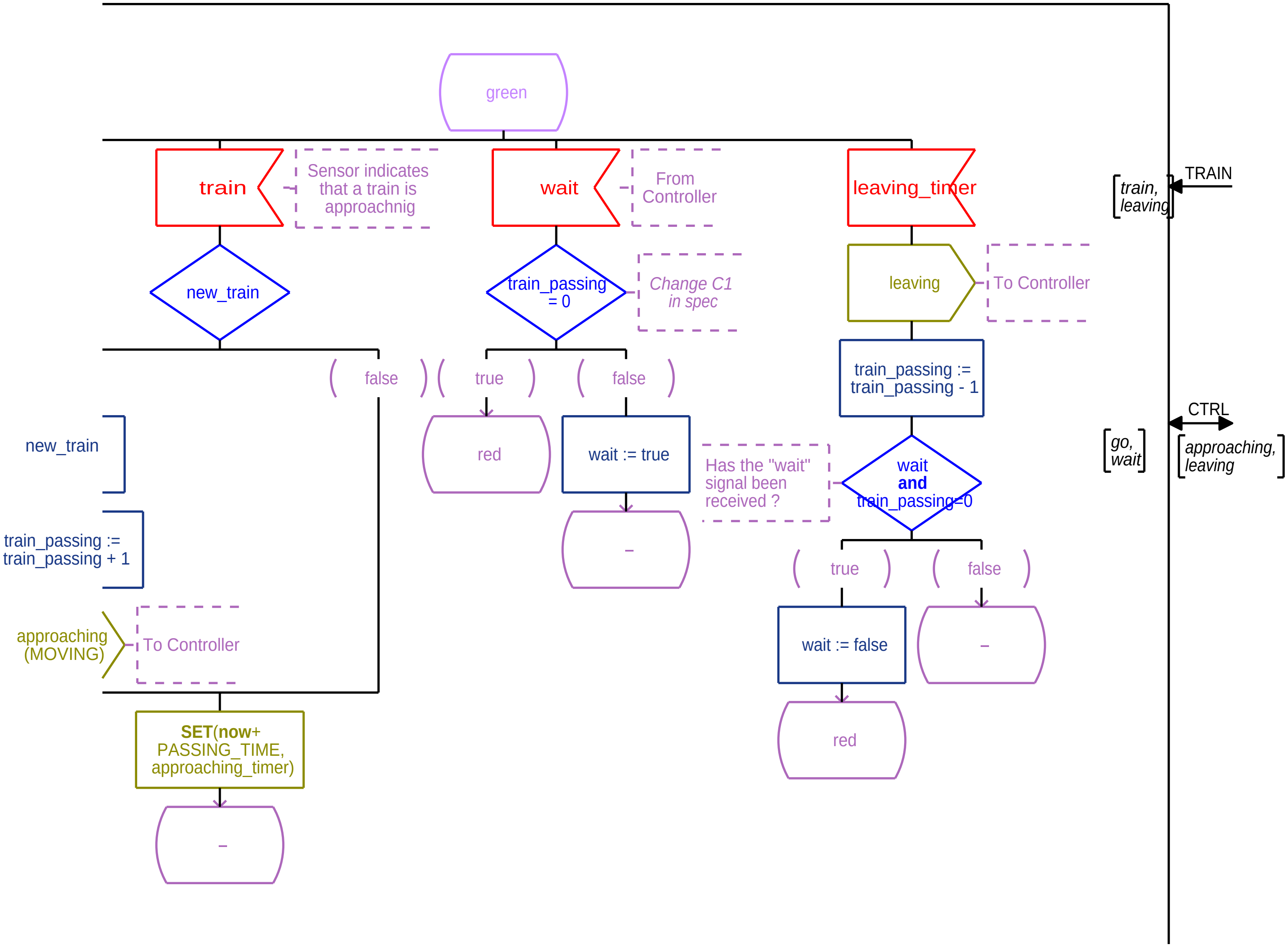
```
signal wait, go,
approaching_normal(movement_ty),
approaching_fast(movement_ty);
```

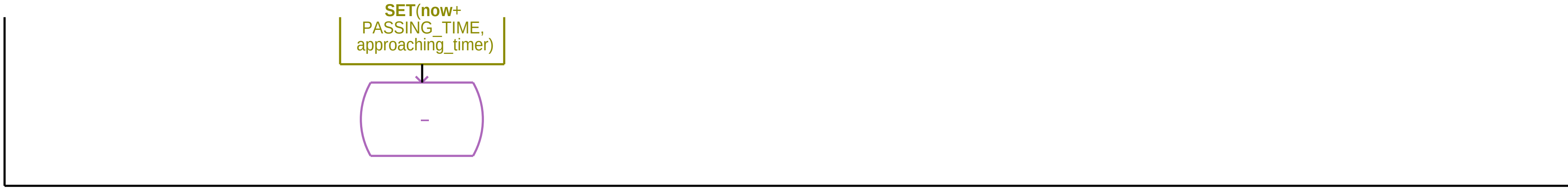
process type Track
<**signal** approaching(movement_ty)>

```
/* This process manages the traffic
 * light (the stopping signal) associated to
 * each track (Fast or normal).
 *
 *
 * "Track" is a process type with context parameter
 * for the signal "approaching". It can be used for fast and
 * normal speed tracks by using a different name for
 * this signal.
 *
 *
*/
```

```
dcl train_passing natural := 0;

dcl wait boolean := false;
/* variable "wait" to prevent setting the wait signal
while there are still some trains between the
two sensors */

dcl new_train boolean := true;
/* variable "new_train" is used to distinguish
an approaching signal from a new train than from
the wheels of a train currently on the sensor
(change C3 in the specificqtion) */

timer leaving_timer,approaching_timer;
/* Time after which it is considered that the entire
train has passed over the sensors */
```

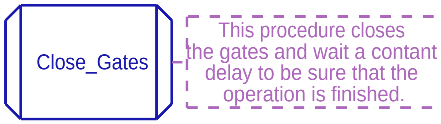

process type Track
<**signal** approaching(movement_ty)>
red
wait
From Controller
train
Sensor indicates that a train is approachnig
Discard signal
–
(train_passing= 0) **or** (train_passing=1 **and** new_train=false)
Prevents simulator for sending several trains when the light is red
new_train
true
false
new_train := false
train_passing := train_passing + 1
approaching (WAITING)
To Controller
red, green
go
From Controller
approaching_timer
Sequence of approaching signals is finished. Wait for a new train on the track.
wait := false
green
new_train := true
–
leaving
Sensor indicates that the train is leaving the crossing station
(train_passing>10) **or** (train_passing=1 **and** new_train=true)
Prevents simulator from sending Leaving signal when no train is on the track
SET(now+ PASSING_TIME, leaving_timer)
–
true
new_tra := fals
train_pass train_pass
approacl (MOVIN


green
train
Sensor indicates that a train is approachnig
wait
From Controller
leaving_timer
new_train
train_passing = 0
Change C1 in spec
leaving
To Controller
train_passing := train_passing - 1
false
true
false
red
wait := true
Has the "wait" signal been received ?
wait and train_passing=0
–
new_train
train_passing := train_passing + 1
true
false
approaching (MOVING)
To Controller
wait := false
–
SET(now+ PASSING_TIME, approaching_timer)
red
–
TRAIN
train, leaving
CTRL
go, wait
approaching, leaving

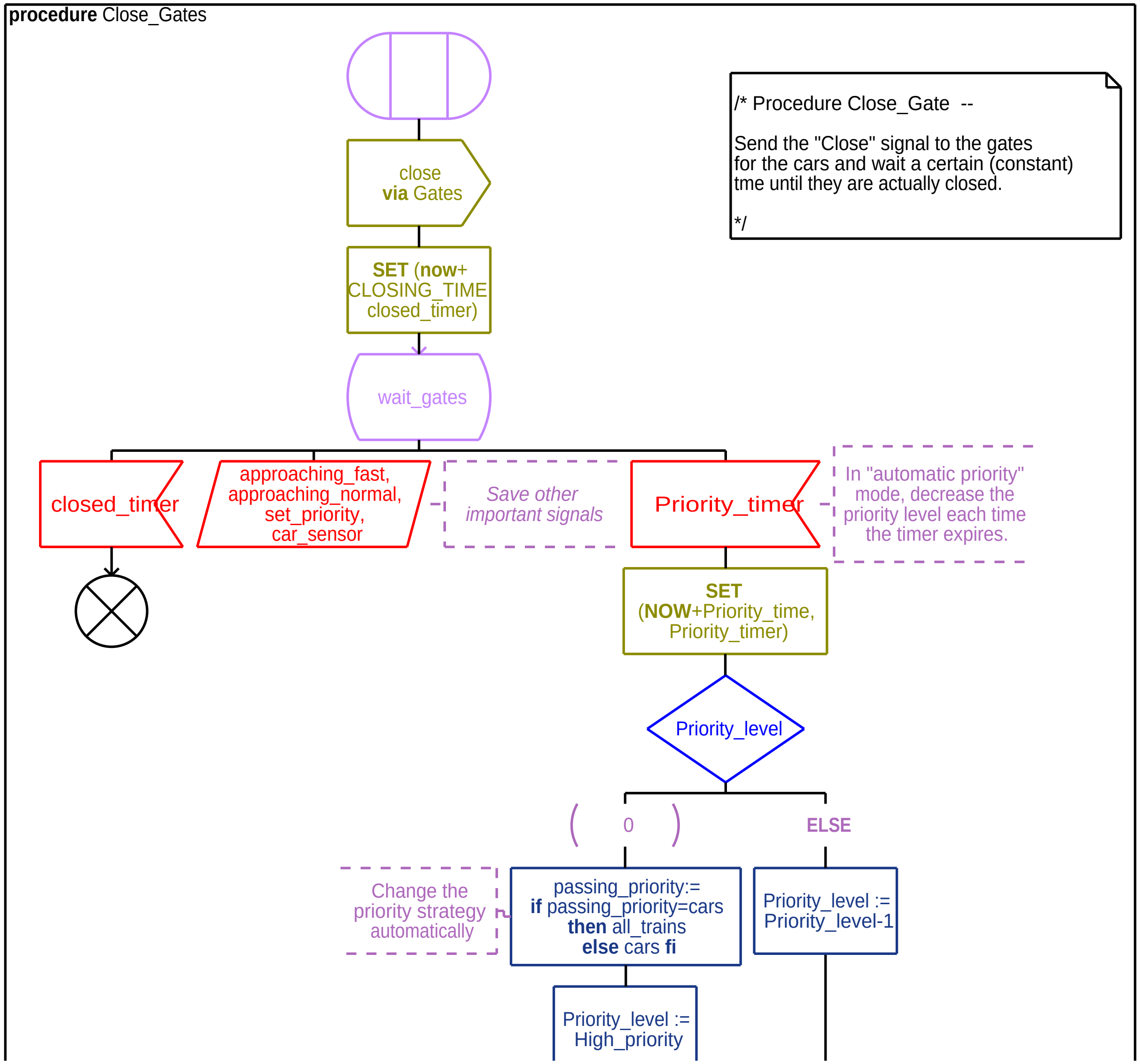

SET(now+
PASSING_TIME,
approaching_timer)
-

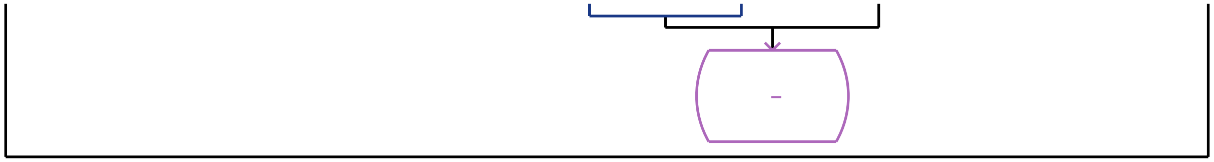
process Controller
(1,1)

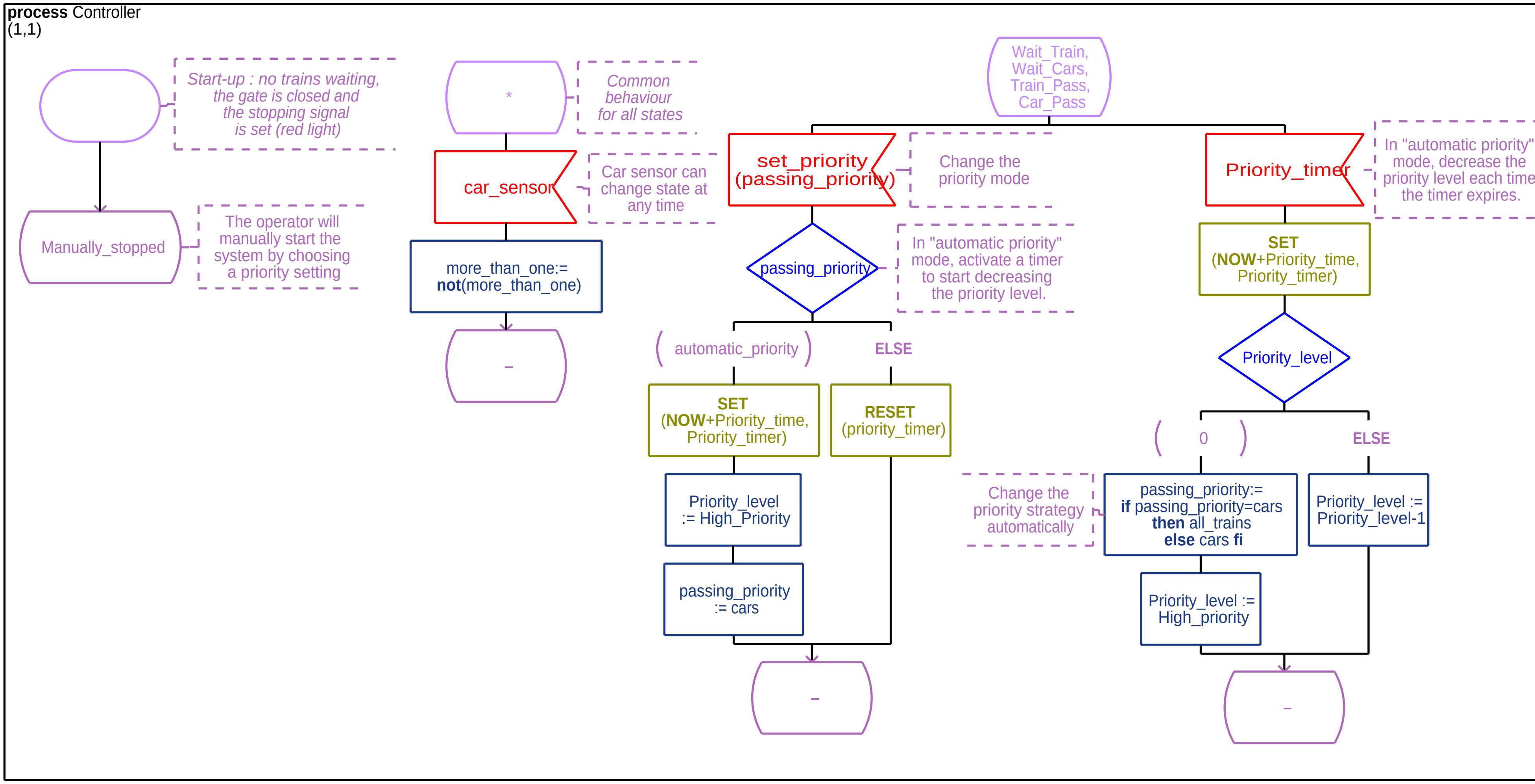
```
dcl passing natural := 0;
/* "passing" counts the number of trains currently passing on one track. */

dcl stopped natural :=0;
/* "stopped" counts the number of trains waiting for authorization to pass. */

dcl more_than_one boolean := false;
/* "more_than_one" is set to true if too many cars are waiting ar the gate.  */

Timer closed_timer;
/* "closed_timer" is the timer needed for closing the gates */

dcl Priority_train_id pid := null;
/* "Priority_train_id stores the PID of a train with priority over cars. It is used
for sending the "go" signal to this train after closing the gates
(the "Close_Gates" procedure uses a timer which change the "sender" value). */

dcl movement movement_ty;
/* "movement" is used to determine if the train which approaches is stopped or
 * not, in the particular case when the controller expects the lights to be all red : it can
 * actually be passing if it follows another train which has not left, as the track is not
 * allowed to set the stopping signal when there is a train between the two sensors.
 * Possible values are "WAITING" and "MOVING".
*/

dcl Passing_priority Priority_ty := fast_trains;
/* "Passing_priority" is used to set the priority strategy */

dcl Priority_level natural := 0;
/* "priority_level" is used in automatic priority mode to know when to change the
 * priority strategy. It start with a high value and starts decreasing as time goes on.
*/

timer Priority_timer;
/* "priority_timer" is used in automatic priority mode to decrease the priority level */
```

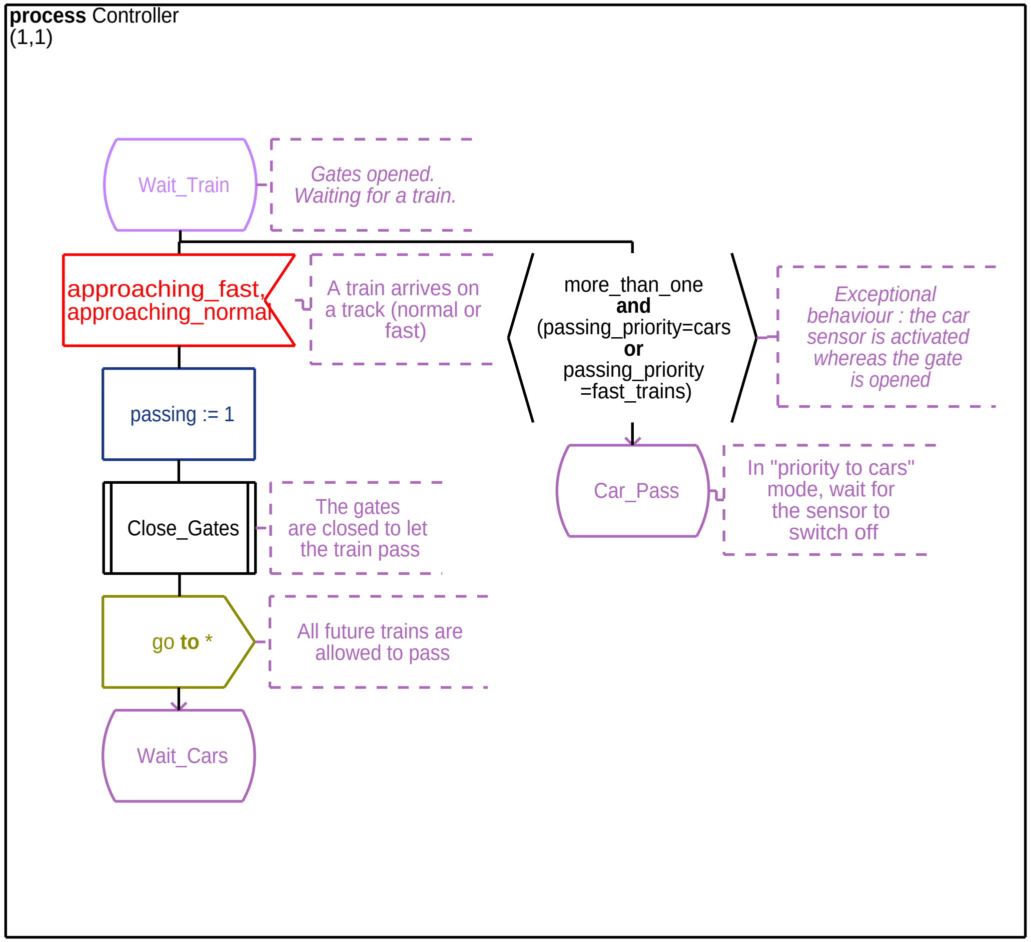


This procedure closes the gates and wait a contant delay to be sure that the operation is finished.

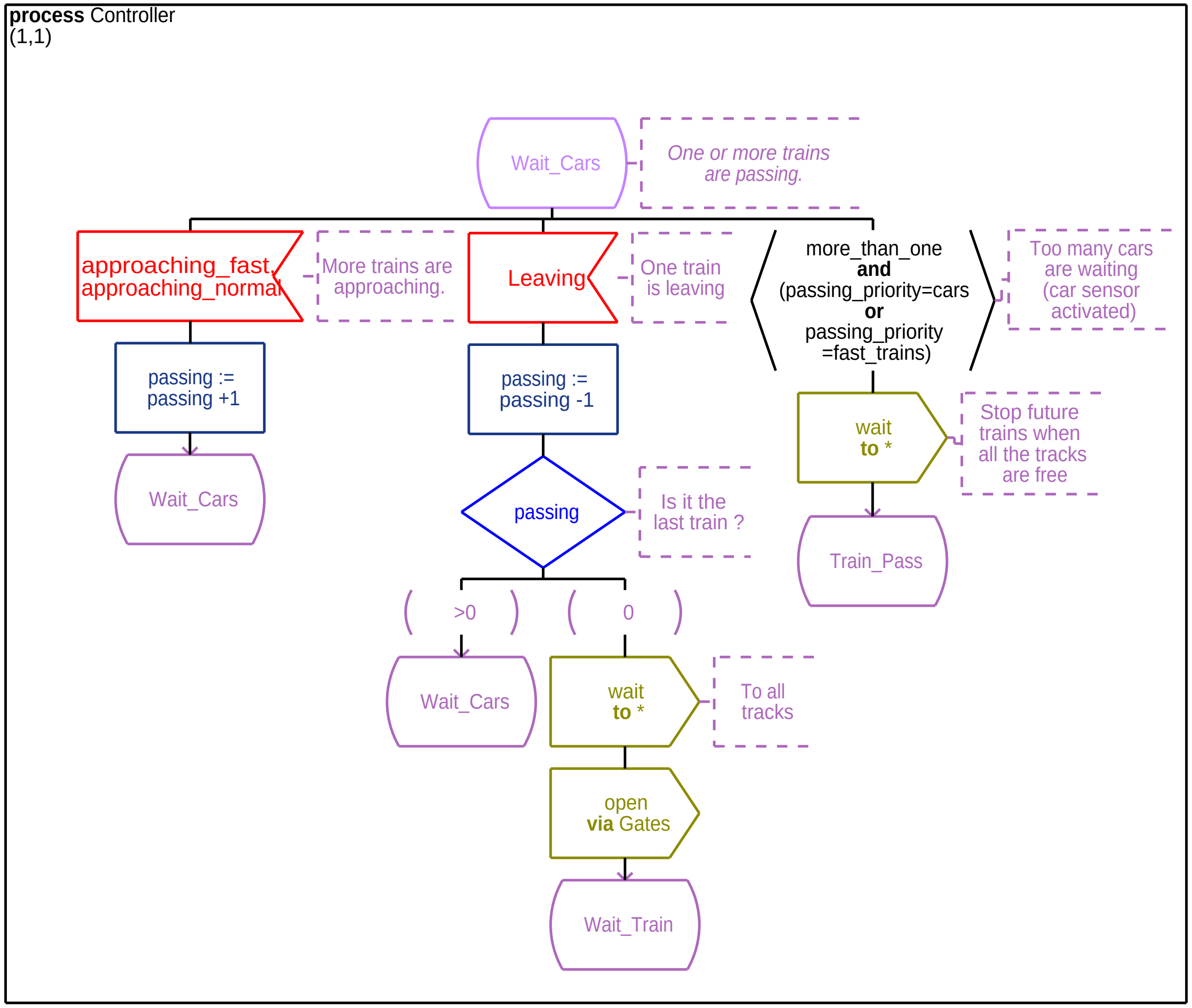

procedure Close_Gates
/* Procedure Close_Gate --
Send the "Close" signal to the gates
for the cars and wait a certain (constant)
tme until they are actually closed.
*/
close
via Gates
SET (**now**+
CLOSING_TIME
closed_timer)
wait_gates
closed_timer
approaching_fast,
approaching_normal,
set_priority,
car_sensor
Save other
important signals
Priority_timer
In "automatic priority"
mode, decrease the
priority level each time
the timer expires.
SET
(**NOW**+Priority_time,
Priority_timer)
Priority_level
(0)
ELSE
Change the
priority strategy
automatically
passing_priority:=
if passing_priority=cars
then all_trains
else cars **fi**
Priority_level :=
Priority_level-1
Priority_level :=
High_priority

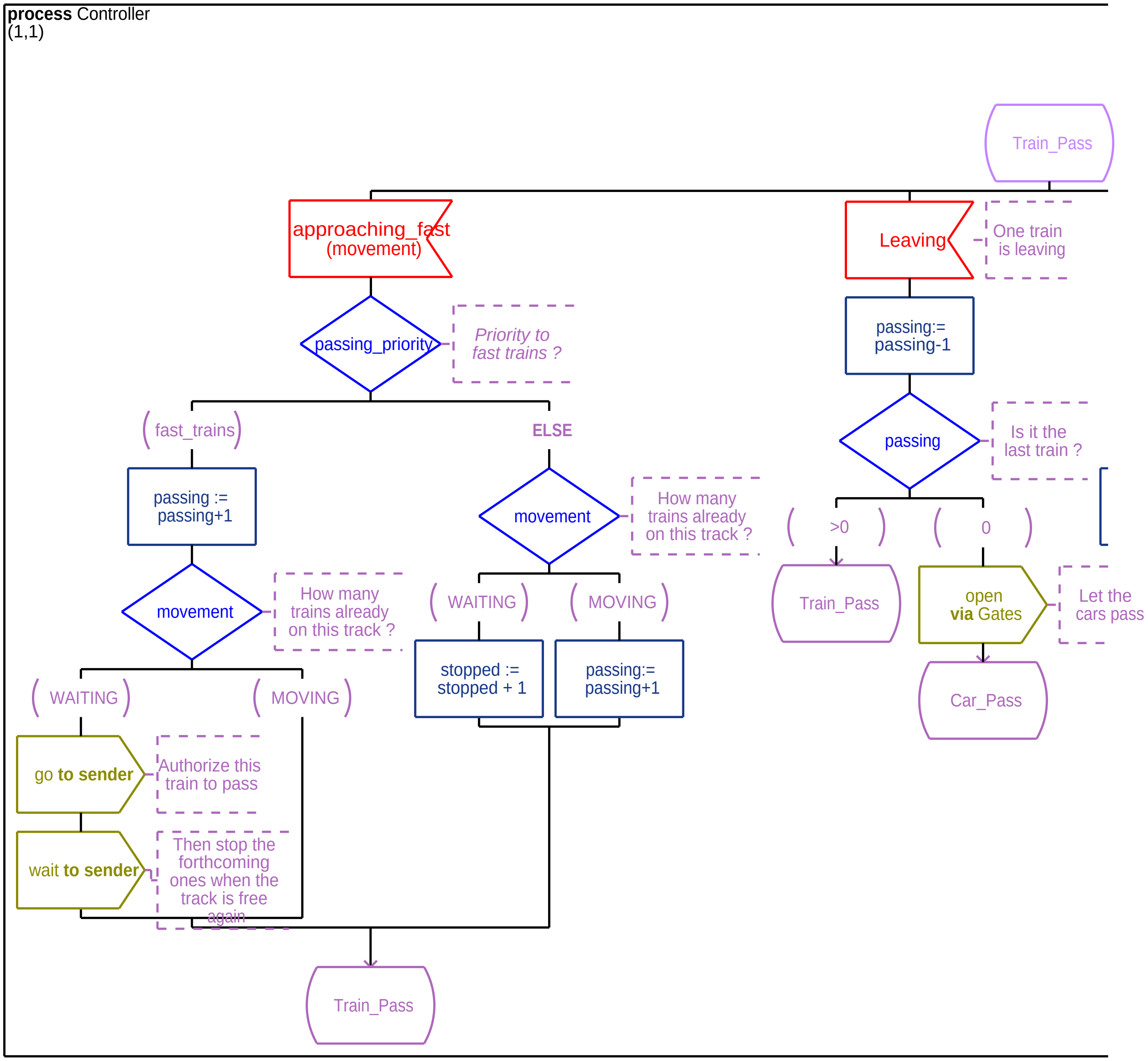


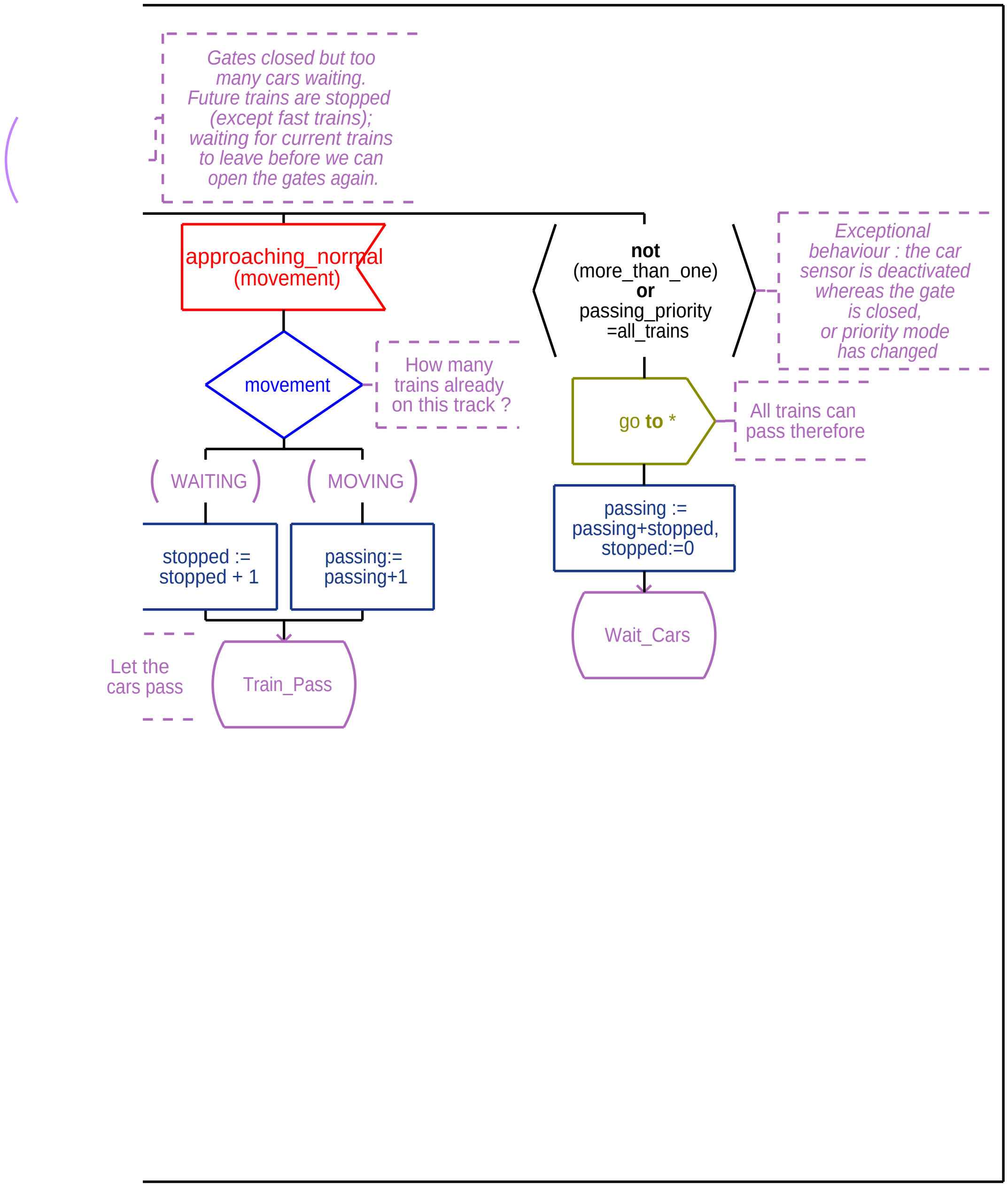

process Controller
(1,1)
Start-up : no trains waiting,
the gate is closed and
the stopping signal
is set (red light)
Manually_stopped
The operator will
manually start the
system by choosing
a priority setting
*
Common
behaviour
for all states
car_sensor
Car sensor can
change state at
any time
more_than_one:=
not(more_than_one)
–
Wait_Train,
Wait_Cars,
Train_Pass,
Car_Pass
set_priority
(passing_priority)
Change the
priority mode
passing_priority
In "automatic priority"
mode, activate a timer
to start decreasing
the priority level.
automatic_priority
ELSE
SET
(**NOW**+Priority_time,
Priority_timer)
RESET
(priority_timer)
Priority_level
:= High_Priority
passing_priority
:= cars
–
Priority_timer
In "automatic priority"
mode, decrease the
priority level each time
the timer expires.
SET
(**NOW**+Priority_time,
Priority_timer)
Priority_level
0
ELSE
Change the
priority strategy
automatically
passing_priority:=
if passing_priority=cars
then all_trains
else cars **fi**
Priority_level :=
Priority_level-1
Priority_level :=
High_priority
–

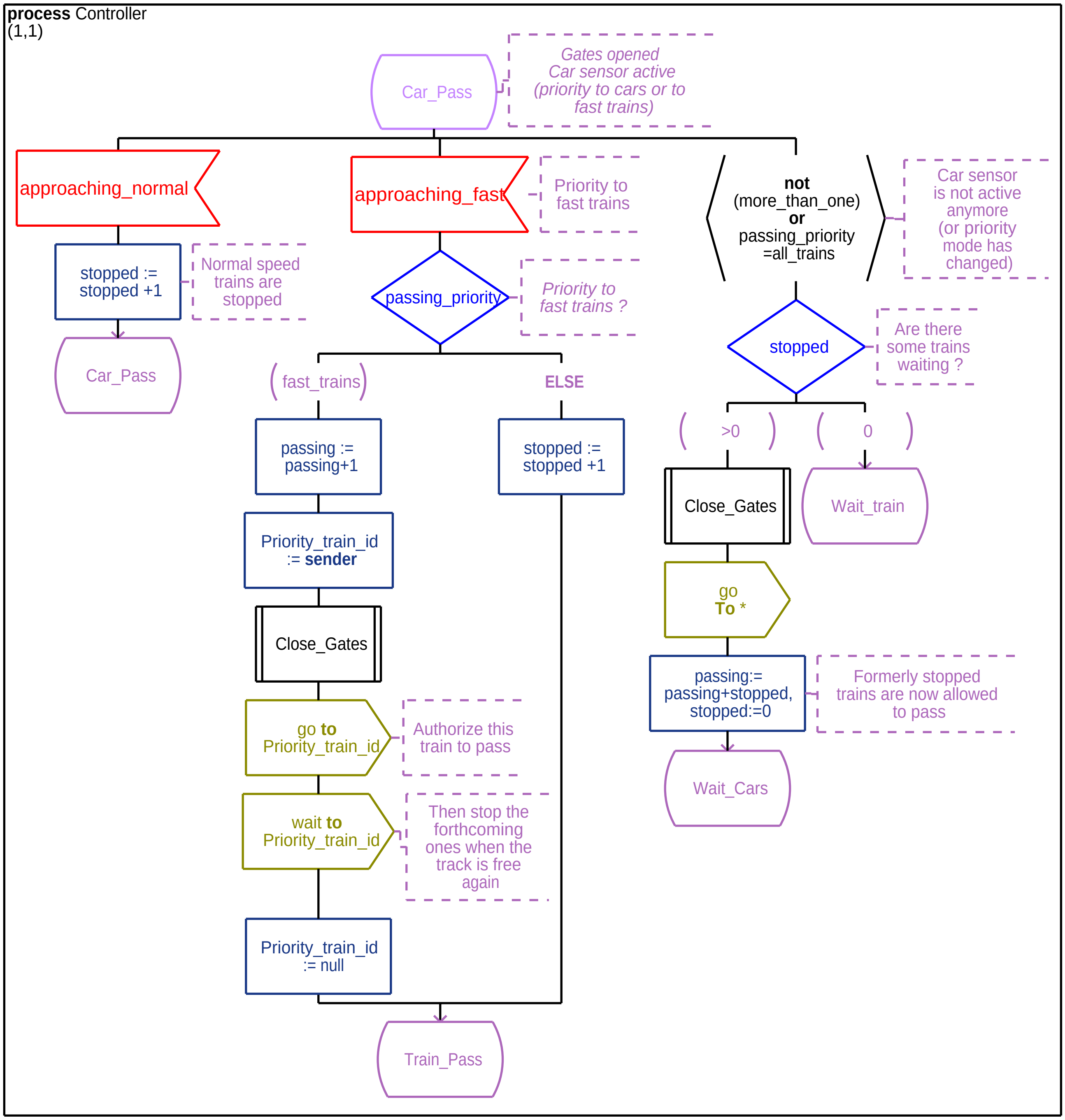

process Controller
(1,1)
Wait_Train
Gates opened. Waiting for a train.
approaching_fast, approaching_normal
A train arrives on a track (normal or fast)
more_than_one **and** (passing_priority=cars **or** passing_priority =fast_trains)
Exceptional behaviour : the car sensor is activated whereas the gate is opened
passing := 1
Close_Gates
The gates are closed to let the train pass
Car_Pass
In "priority to cars" mode, wait for the sensor to switch off
go **to** *
All future trains are allowed to pass
Wait_Cars

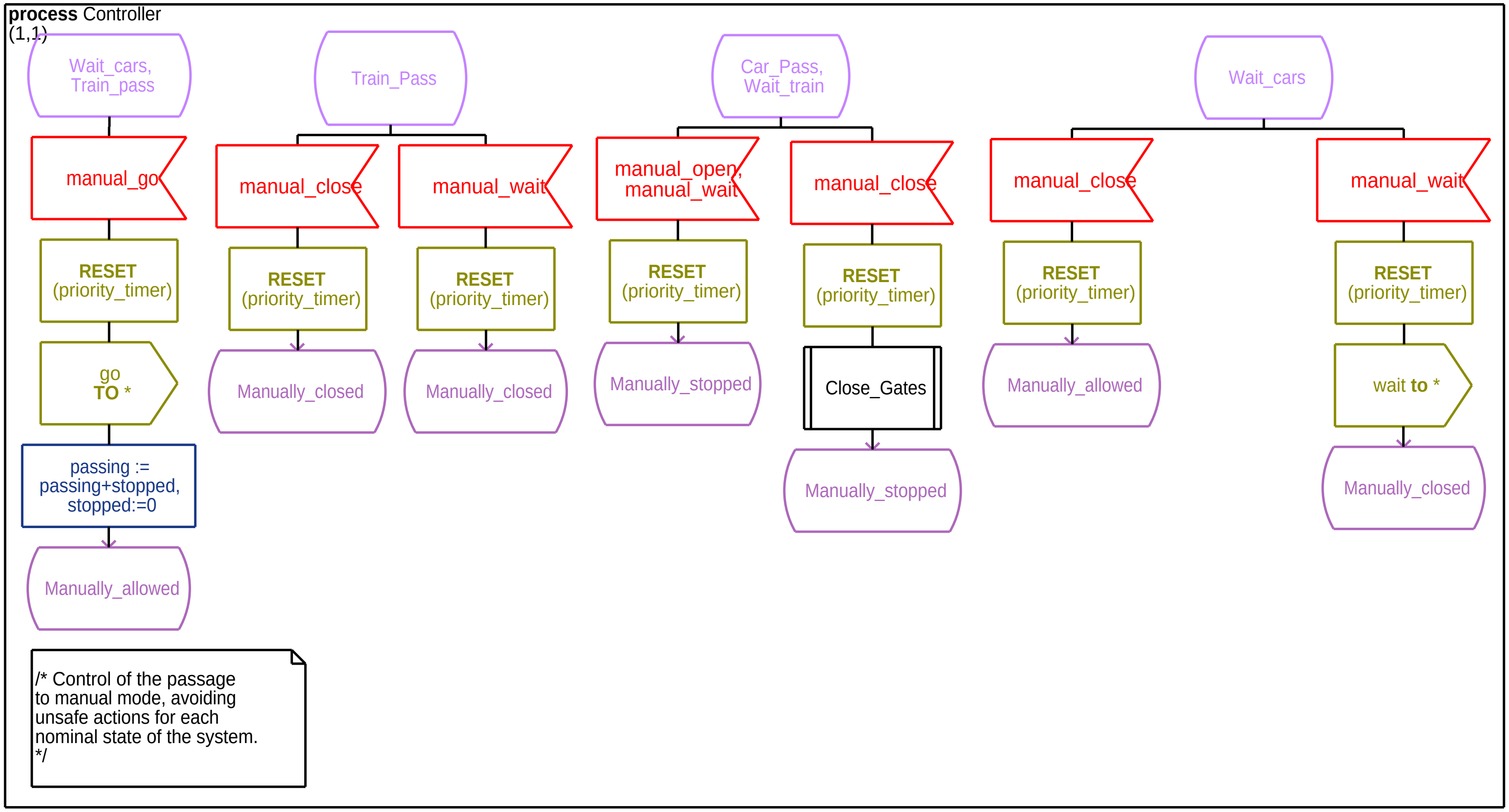
Controller | Wait_Cars | Thu May 23 18:22:38 2002

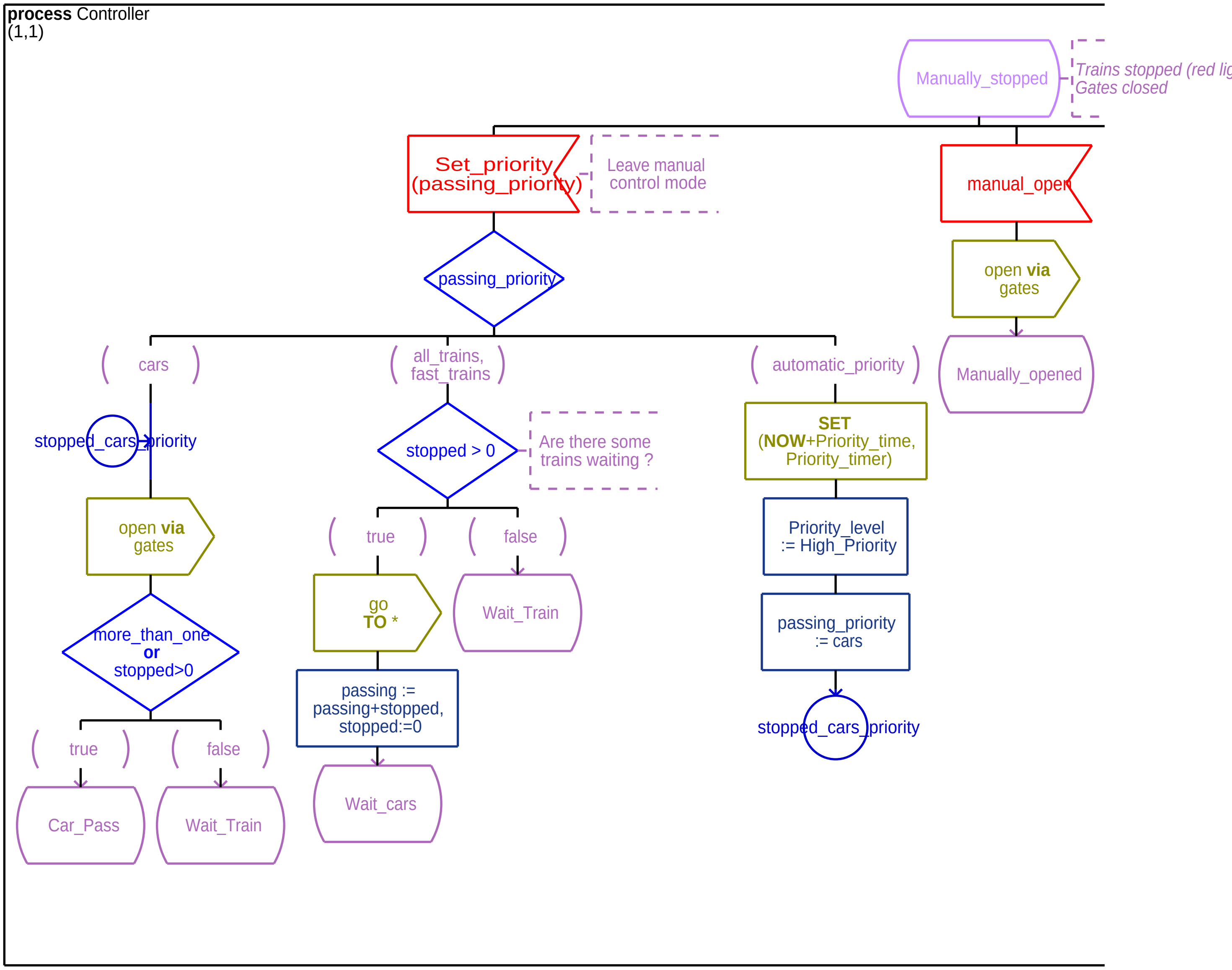

process Controller
(1,1)
Wait_Cars
One or more trains are passing.
approaching_fast, approaching_normal
More trains are approaching.
Leaving
One train is leaving
more_than_one **and** (passing_priority=cars **or** passing_priority =fast_trains)
Too many cars are waiting (car sensor activated)
passing := passing +1
Wait_Cars
passing := passing -1
passing
Is it the last train ?
>0
0
Wait_Cars
wait **to** *
To all tracks
open **via** Gates
Wait_Train
wait **to** *
Stop future trains when all the tracks are free
Train_Pass

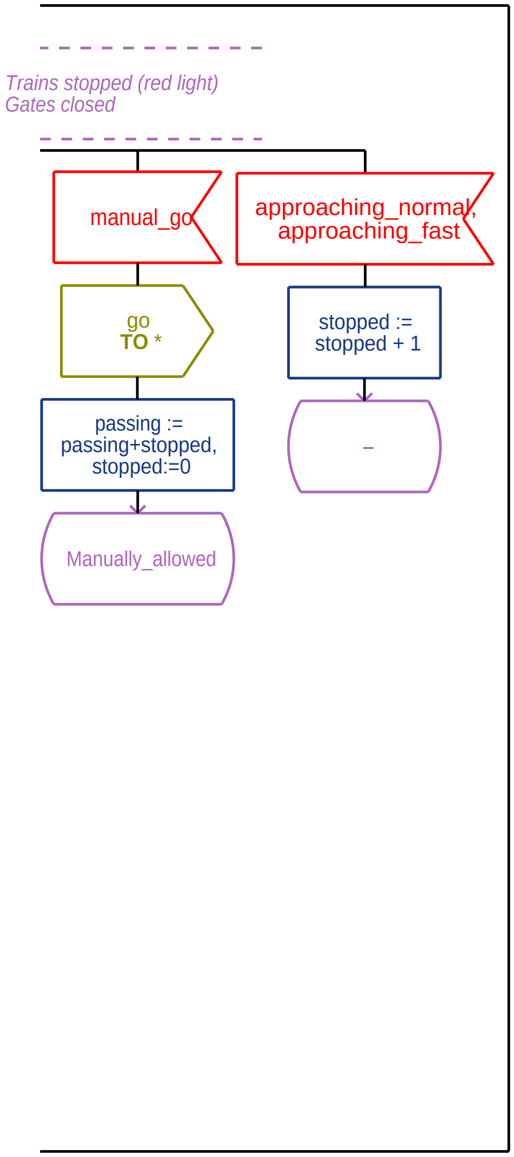

process Controller
(1,1)
Train_Pass
approaching_fast
(movement)
Leaving
One train
is leaving
passing_priority
Priority to
fast trains ?
passing:=
passing-1
fast_trains
ELSE
passing
Is it the
last train ?
passing :=
passing+1
movement
How many
trains already
on this track ?
>0
0
Train_Pass
open
via Gates
Let the
cars pass
movement
How many
trains already
on this track ?
WAITING
MOVING
stopped :=
stopped + 1
passing:=
passing+1
Car_Pass
WAITING
MOVING
go to sender
Authorize this
train to pass
wait to sender
Then stop the
forthcoming
ones when the
track is free
again
Train_Pass

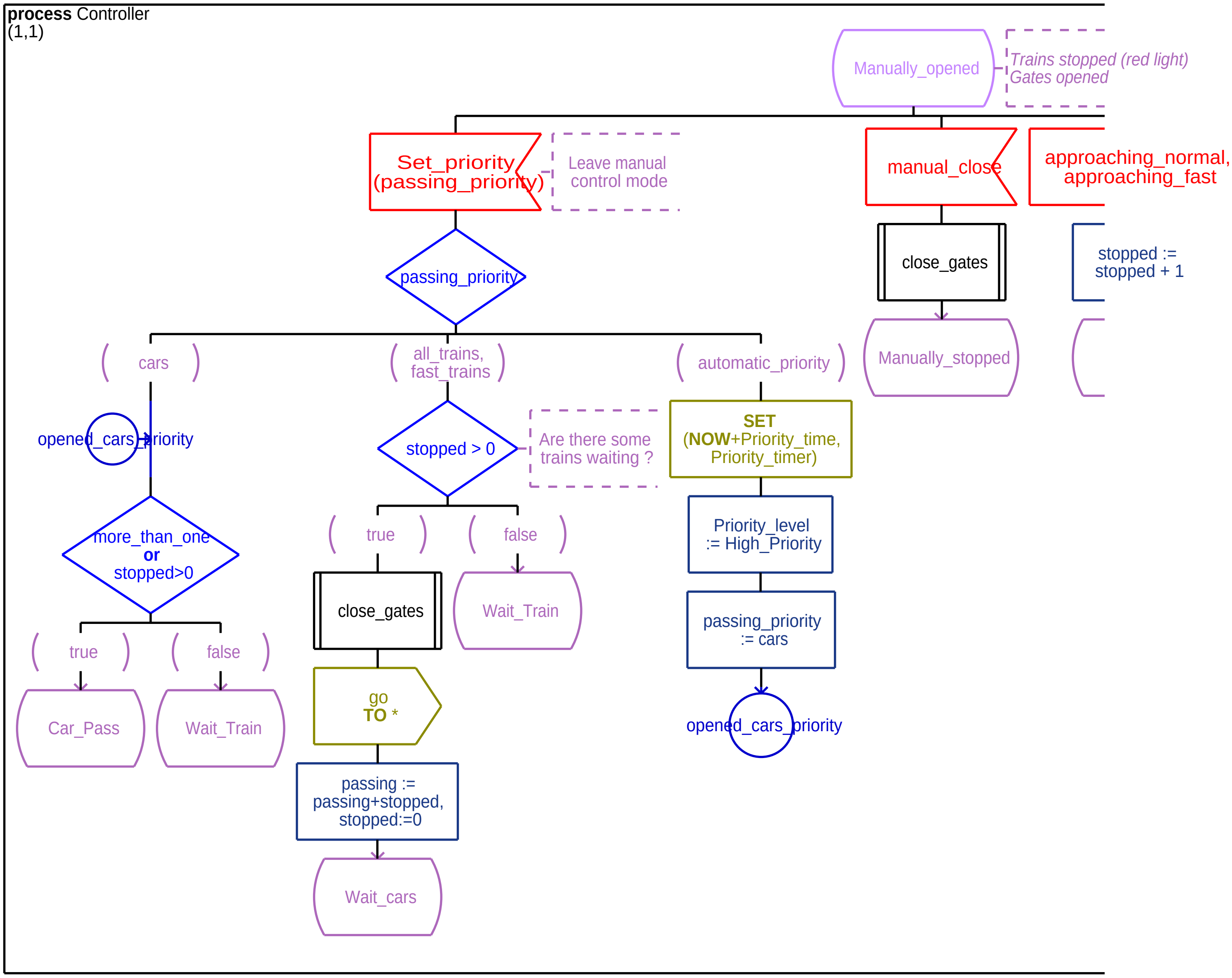

Gates closed but too many cars waiting. Future trains are stopped (except fast trains); waiting for current trains to leave before we can open the gates again.
approaching_normal (movement)
movement
How many trains already on this track ?
WAITING
MOVING
stopped := stopped + 1
passing:= passing+1
Let the cars pass
Train_Pass
not (more_than_one) or passing_priority =all_trains
Exceptional behaviour : the car sensor is deactivated whereas the gate is closed, or priority mode has changed
go to *
All trains can pass therefore
passing := passing+stopped, stopped:=0
Wait_Cars

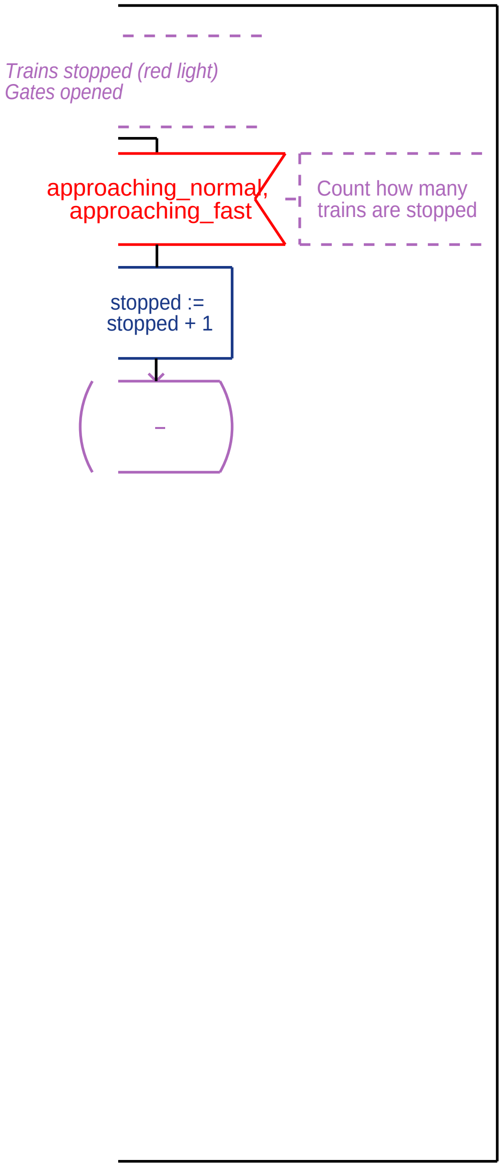

process Controller
(1,1)
Car_Pass
Gates opened
Car sensor active
(priority to cars or to
fast trains)
approaching_normal
stopped :=
stopped +1
Normal speed
trains are
stopped
Car_Pass
approaching_fast
Priority to
fast trains
passing_priority
Priority to
fast trains ?
fast_trains
ELSE
passing :=
passing+1
Priority_train_id
:= sender
Close_Gates
go to
Priority_train_id
Authorize this
train to pass
wait to
Priority_train_id
Then stop the
forthcoming
ones when the
track is free
again
Priority_train_id
:= null
stopped :=
stopped +1
Train_Pass
not
(more_than_one)
or
passing_priority
=all_trains
Car sensor
is not active
anymore
(or priority
mode has
changed)
stopped
Are there
some trains
waiting ?
>0
0
Close_Gates
Wait_train
go
To *
passing:=
passing+stopped,
stopped:=0
Formerly stopped
trains are now allowed
to pass
Wait_Cars

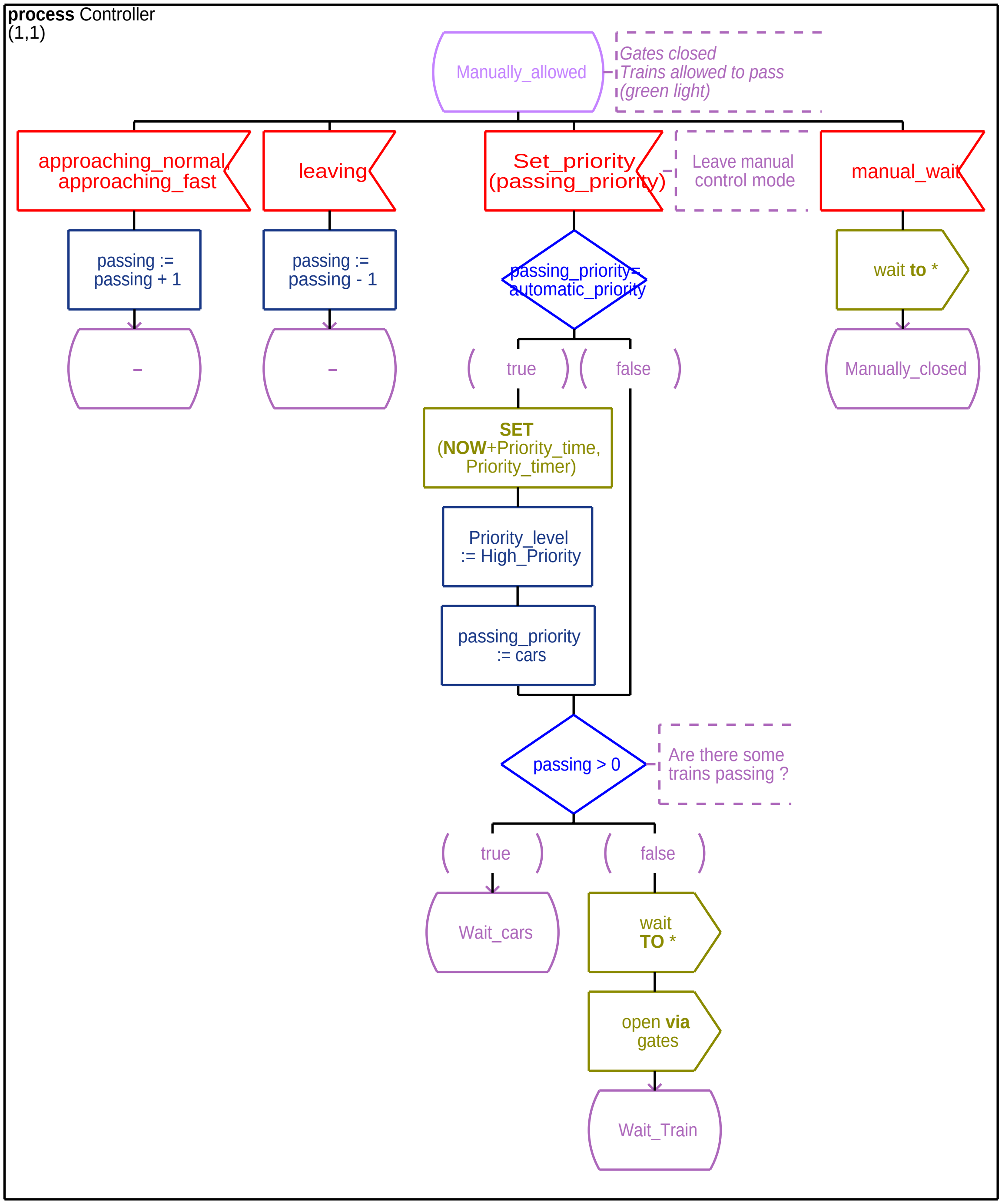

process Controller
(1,1)
Wait_cars,
Train_pass
manual_go
RESET
(priority_timer)
go
TO *
passing :=
passing+stopped,
stopped:=0
Manually_allowed
/* Control of the passage
to manual mode, avoiding
unsafe actions for each
nominal state of the system.
*/
Train_Pass
manual_close
RESET
(priority_timer)
Manually_closed
manual_wait
RESET
(priority_timer)
Manually_closed
Car_Pass,
Wait_train
manual_open,
manual_wait
RESET
(priority_timer)
Manually_stopped
manual_close
RESET
(priority_timer)
Close_Gates
Manually_stopped
Wait_cars
manual_close
RESET
(priority_timer)
Manually_allowed
manual_wait
RESET
(priority_timer)
wait to *
Manually_closed


process Controller
(1,1)
Manually_stopped
Trains stopped (red lig
Gates closed
Set_priority
(passing_priority)
Leave manual
control mode
manual_open
passing_priority
open via
gates
Manually_opened
cars
all_trains,
fast_trains
automatic_priority
stopped_cars_priority
stopped > 0
Are there some
trains waiting ?
SET
(NOW+Priority_time,
Priority_timer)
open via
gates
true
false
Priority_level
:= High_Priority
go
TO *
Wait_Train
more_than_one
or
stopped>0
passing_priority
:= cars
passing :=
passing+stopped,
stopped:=0
stopped_cars_priority
true
false
Car_Pass
Wait_Train
Wait_cars

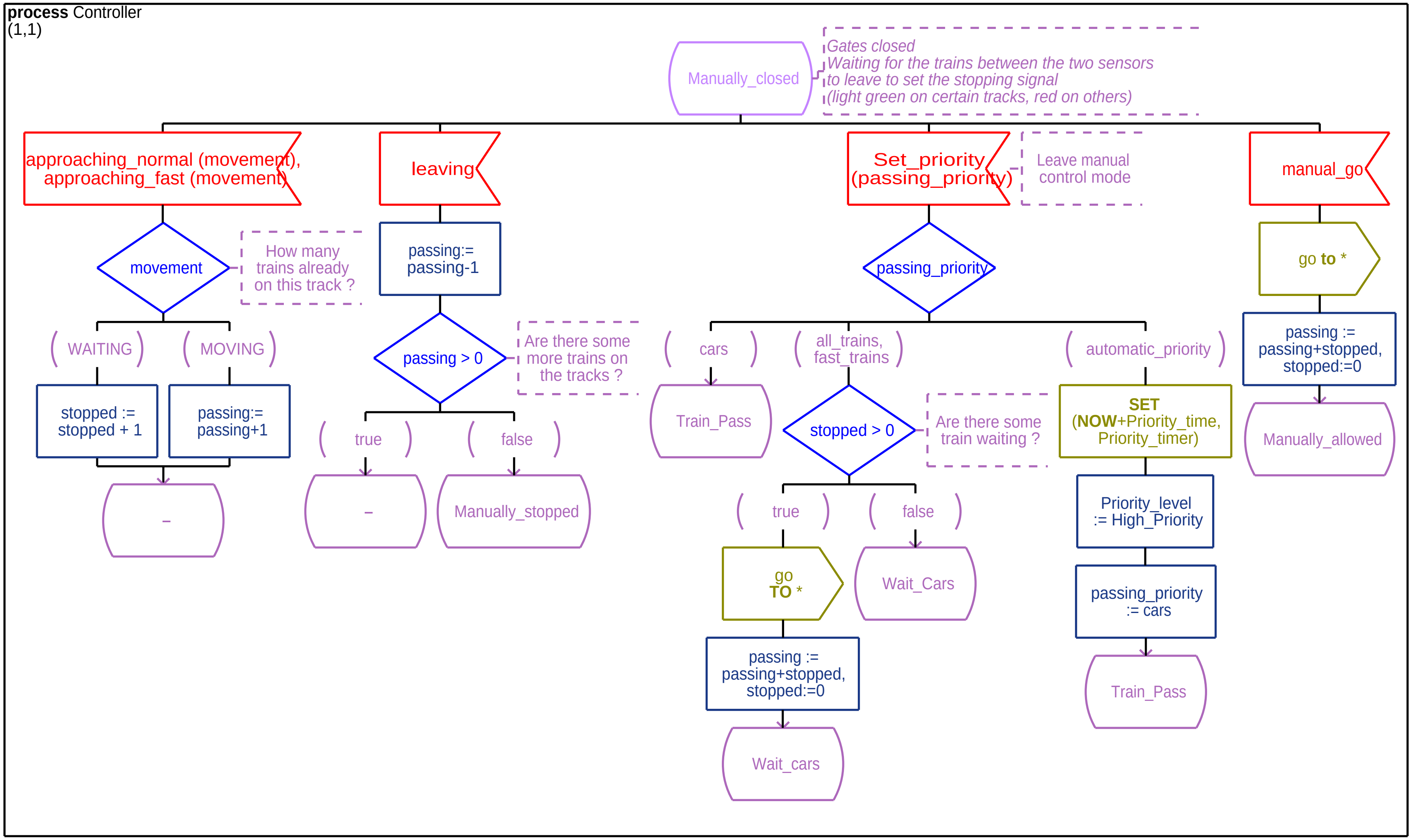

Trains stopped (red light)
Gates closed
manual_go
approaching_normal,
approaching_fast
go
TO *
stopped :=
stopped + 1
passing :=
passing+stopped,
stopped:=0
–
Manually_allowed


process Controller
(1,1)
Manually_opened
Trains stopped (red light)
Gates opened
Set_priority
(passing_priority)
Leave manual
control mode
manual_close
approaching_normal,
approaching_fast
passing_priority
close_gates
stopped :=
stopped + 1
cars
all_trains,
fast_trains
automatic_priority
Manually_stopped
opened_cars_priority
stopped > 0
Are there some
trains waiting ?
SET
(NOW+Priority_time,
Priority_timer)
more_than_one
or
stopped>0
true
false
Priority_level
:= High_Priority
close_gates
Wait_Train
passing_priority
:= cars
true
false
go
TO *
Car_Pass
Wait_Train
opened_cars_priority
passing :=
passing+stopped,
stopped:=0
Wait_cars


Trains stopped (red light)
Gates opened
approaching_normal,
approaching_fast
Count how many
trains are stopped
stopped :=
stopped + 1
–


process Controller
(1,1)
Manually_allowed
Gates closed
Trains allowed to pass
(green light)
approaching_normal,
approaching_fast
leaving
Set_priority
(passing_priority)
Leave manual
control mode
manual_wait
passing :=
passing + 1
passing :=
passing - 1
passing_priority=
automatic_priority
wait **to** *
–
–
Manually_closed
true
false
SET
(**NOW**+Priority_time,
Priority_timer)
Priority_level
:= High_Priority
passing_priority
:= cars
passing > 0
Are there some
trains passing ?
true
false
Wait_cars
wait
TO *
open **via**
gates
Wait_Train

Controller	Manually_closed	Thu May 23 18:22:38 2002


process Controller
(1,1)
Manually_closed
Gates closed
Waiting for the trains between the two sensors
to leave to set the stopping signal
(light green on certain tracks, red on others)
approaching_normal (movement),
approaching_fast (movement)
movement
How many trains already on this track ?
WAITING
MOVING
stopped := stopped + 1
passing:= passing+1
–
leaving
passing:= passing-1
passing > 0
Are there some more trains on the tracks ?
true
false
–
Manually_stopped
Set_priority
(passing_priority)
Leave manual control mode
passing_priority
cars
all_trains, fast_trains
automatic_priority
Train_Pass
stopped > 0
Are there some train waiting ?
true
false
go TO *
Wait_Cars
passing := passing+stopped, stopped:=0
Wait_cars
SET (NOW+Priority_time, Priority_timer)
Priority_level := High_Priority
passing_priority := cars
Train_Pass
manual_go
go to *
passing := passing+stopped, stopped:=0
Manually_allowed