

# TP Multitache

## Gestion d'un Parking Automobile

### Document de Spécification et de Conception

# Gestion d'un Parking Automobile

## Presentation :

Parking est une application de simulation de la gestion d'un Parking automobile. Elle permet d'informatiser la gestion des entrées et sorties pour ce parking, composé de 8 places et ayant 3 entrées et 1 sortie.

## Faiblesses de l'architecture

Un des problèmes rencontré au tout début a été la gestion des arrivées simultanées. Ce problème nous a permis de prendre conscience de l'intérêt d'avoir un processus par Entrée.

## Faiblesses générales

Pendant la phase de conception, nous avons du mal avec la modélisation choisie de l'environnement du parking : en effet celle-ci ne permet (de manière simple et directe) pas de faire arriver simultanément 2 voitures désirant se garer, et nous n'avions pas pris ce point en compte au début de la conception. Ce qui aurait débouché à des problèmes étranges, comme deux voitures attendant depuis longtemps de rentrer : si une place se libérait, et qu'une tierce voiture arrivait simultanément, cette tierce voiture, bien que nouvelle venue, aurait tout à fait prendre cette place au détriment des deux autres véhicules. (Vol de place)

Ainsi, Une des faiblesses générale de l'application est bien sur le fait qu'elle n'est qu'une simulation et donc ne reflète pas la réalité en certains points. Ainsi, on ne peut pas simuler l'arrivée de deux voitures à deux portes différentes au même moment.

## Modifications effectuées par rapport à l'architecture initiale :

Par rapport à l'architecture de base, nous avons finalement opté pour 5 sémaophores, qui nous ont permis pour le premier de pouvoir compter le nombre de places restantes dans le parking (Sémaaphore à compte) et pour 3 autres de gérer les attentes à l'entrée (Sémaaphore de synchronisation) et enfin un Mutex pour protéger la mémoire partagée.

Nous avons aussi pris conscience du fait que l'on a pas besoin d'un canal de communication entre l'entrée et ses voituriers (le voiturier n'aurait pas prévu ça de toute façon), chose que nous n'avons pas su voir lors de la 1ere séance. La communication se fait par l'envoi du SIGCHLD avec positionnement du numéro de la place lors de l'exit.

Finalement, nous avons ajouté bien sur la tâche heure, qui n'était pas présente, par oubli, dans notre conception initiale.

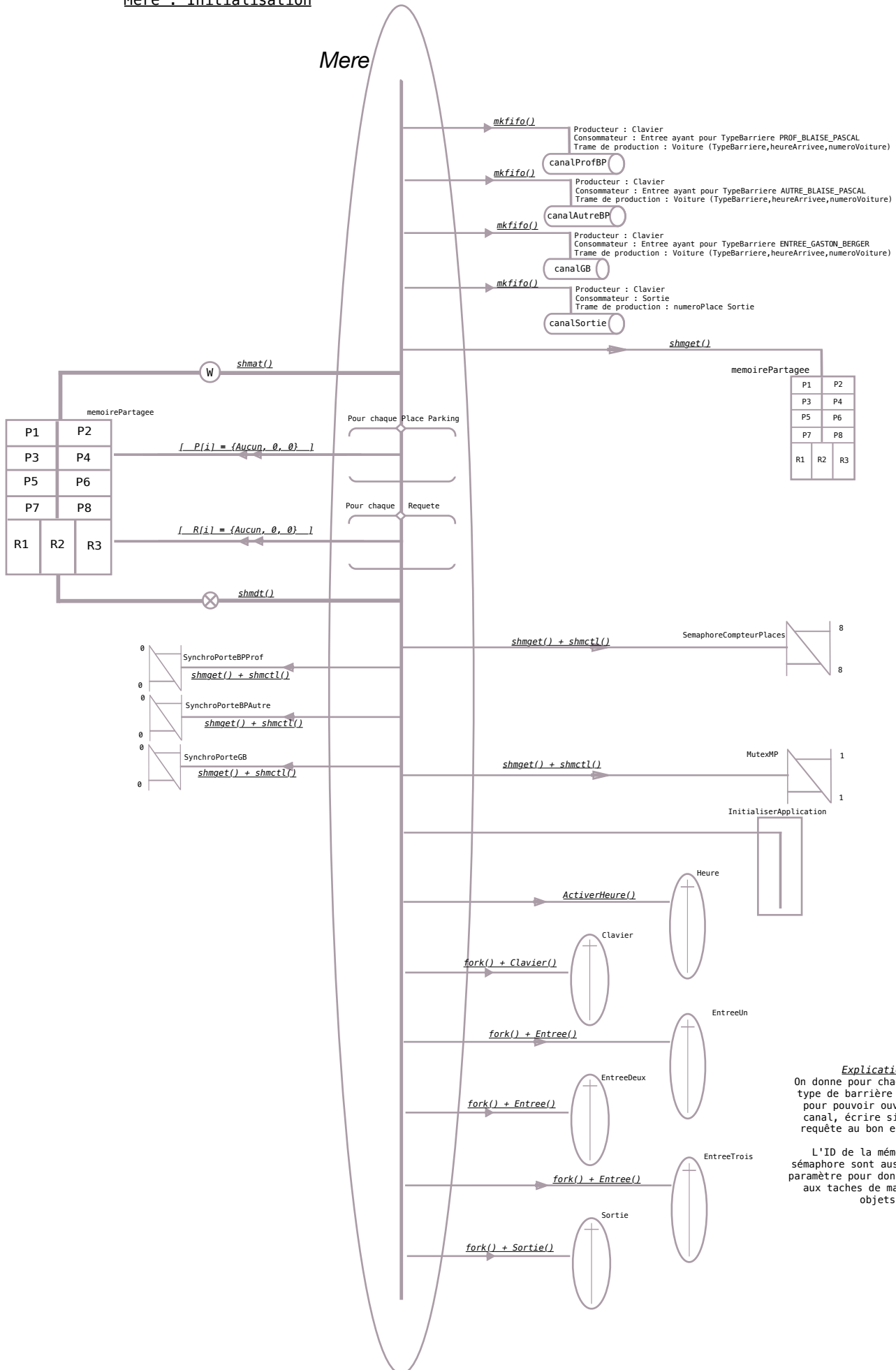
# TP Multitache

## Gestion d'un Parking Automobile

### Document de Spécification et de Conception

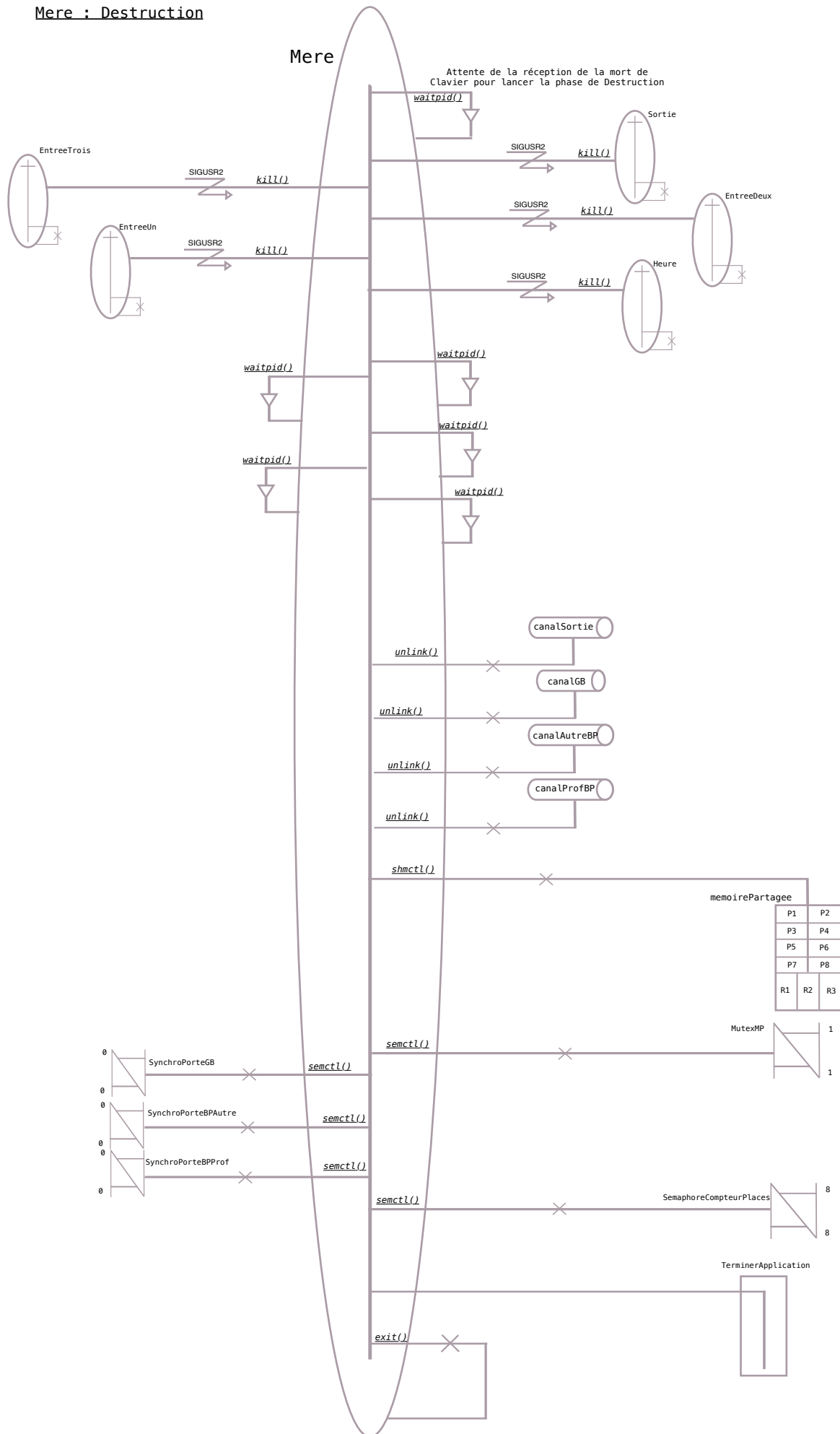
Schémas détaillés des tâches

## Mere : Initialisation

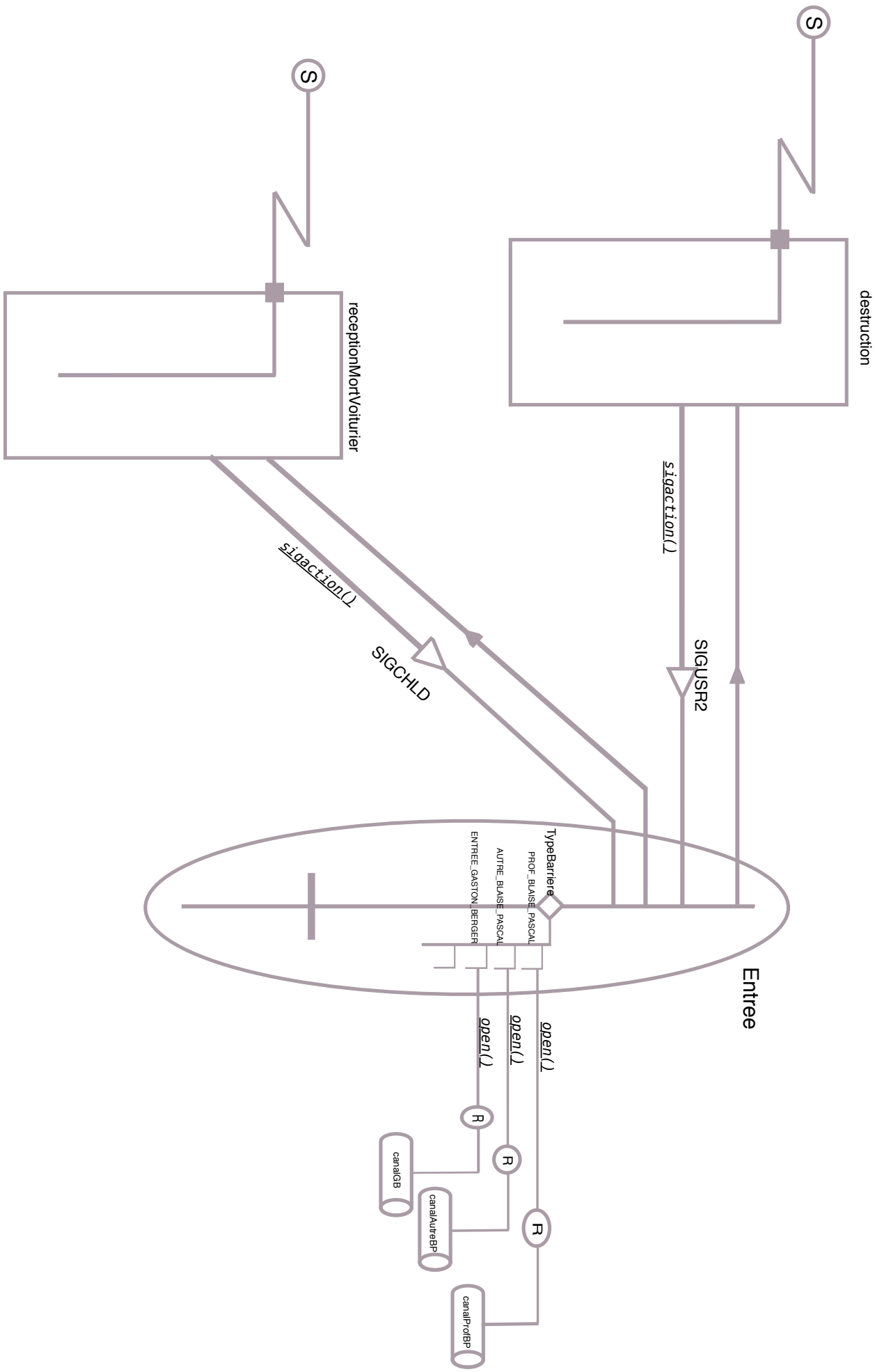


**Explications :**  
On donne pour chaque entree le type de barrière qu'elle gère pour pouvoir ouvrir son bon canal, écrire si il y'a une requête au bon endroit etc..  
  
L'ID de la mémoire et du sémaphore sont aussi fournis en paramètre pour donner l'occasion aux taches de manipuler ces objets !

## Mere : Destruction

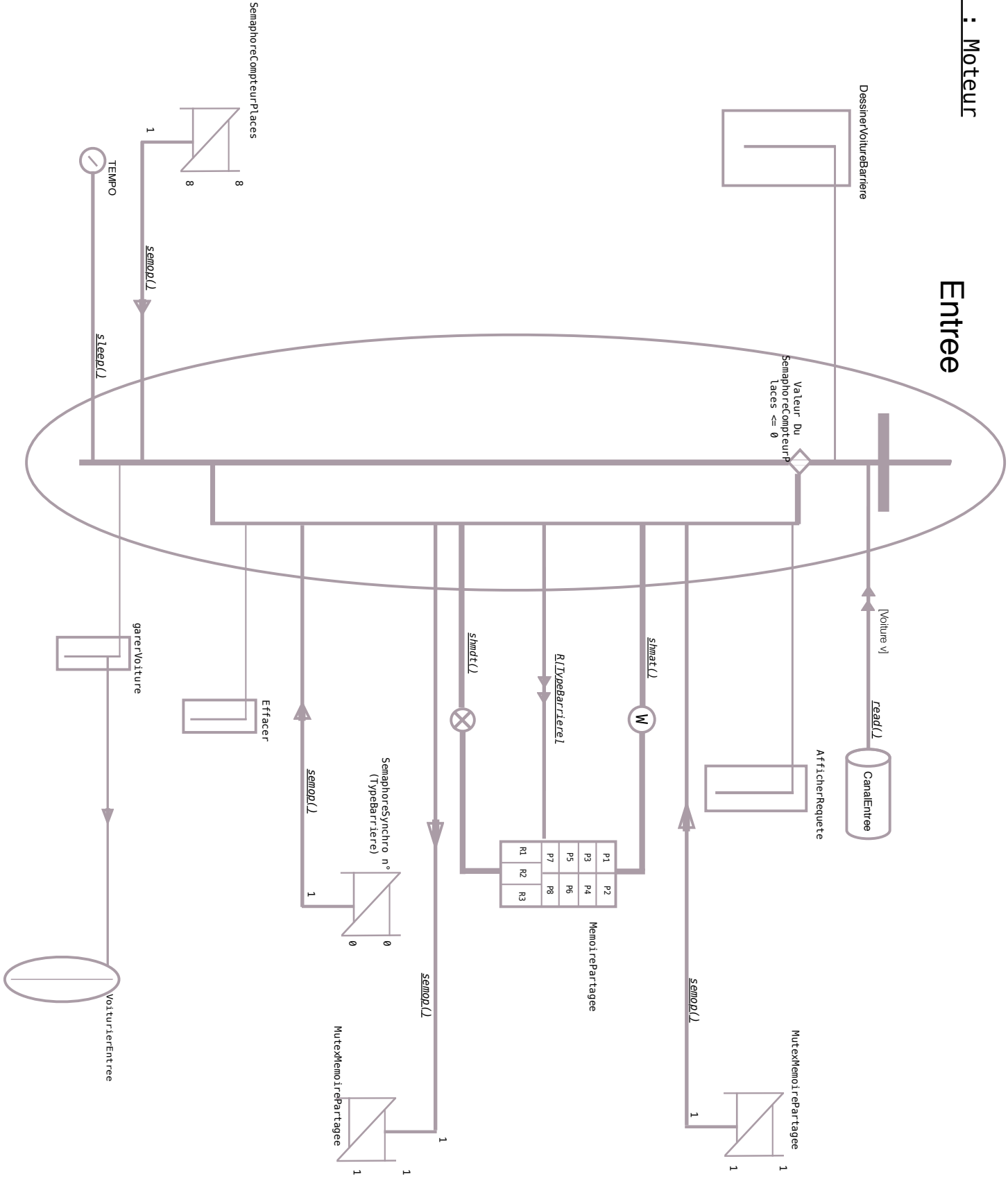


Entree : Initialisation

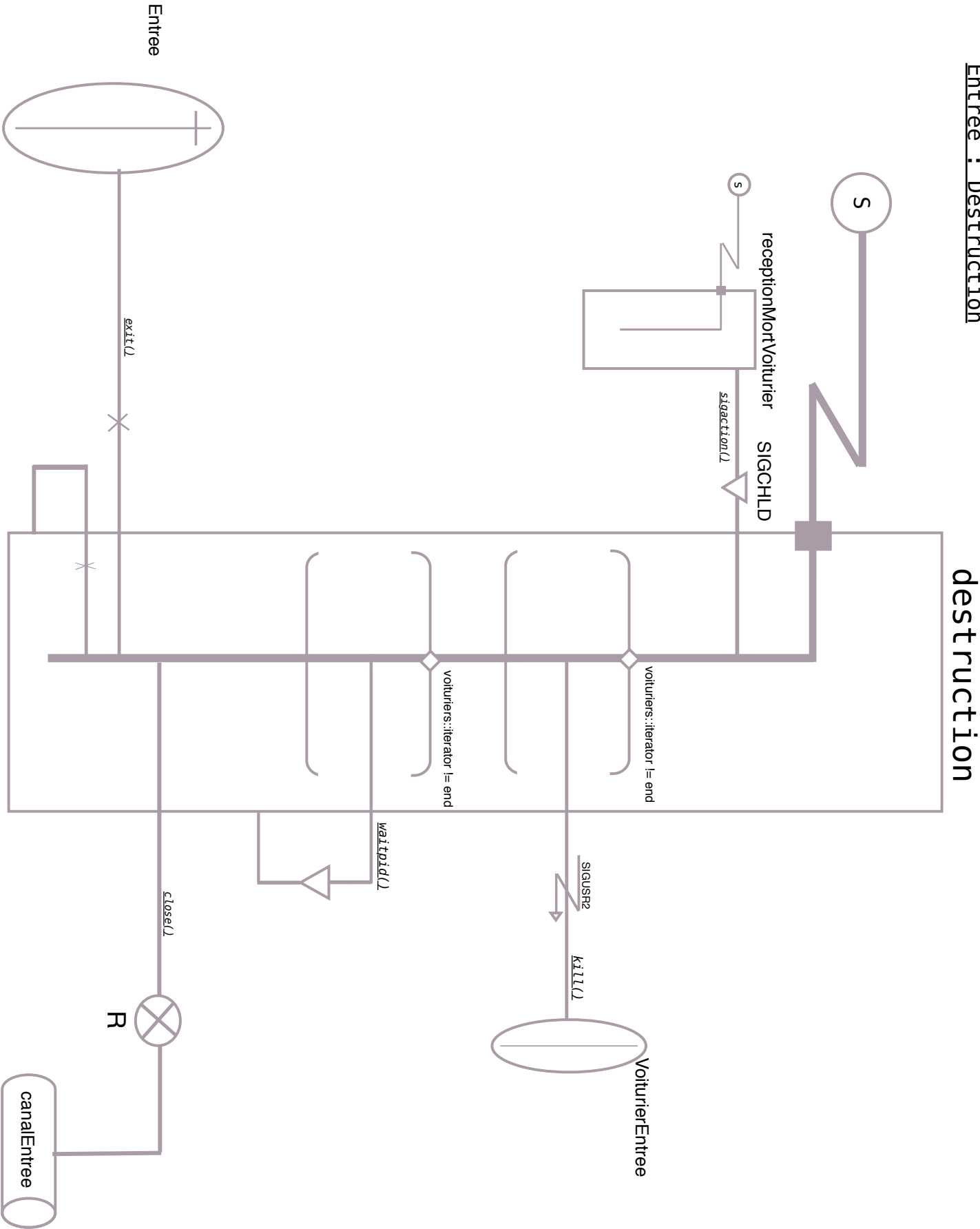


Entree : Moteur

Entree



Entree : Destruction





S

VoiturierEntree

Entree

SIGCHLD

SIGCHLD

receptionMortVoiturier

Entree : receptionMortVoiturier

AffichePlace

MutexMemoirePartagee

semop(1

1



1

shmat(1

W

PIWEXITSTATUS(status)-11

P1	P2
P3	P4
P5	P6
P7	P8
R1	R2
R3	

MemoirePartagee

shmctl(1

⊗

semop(1

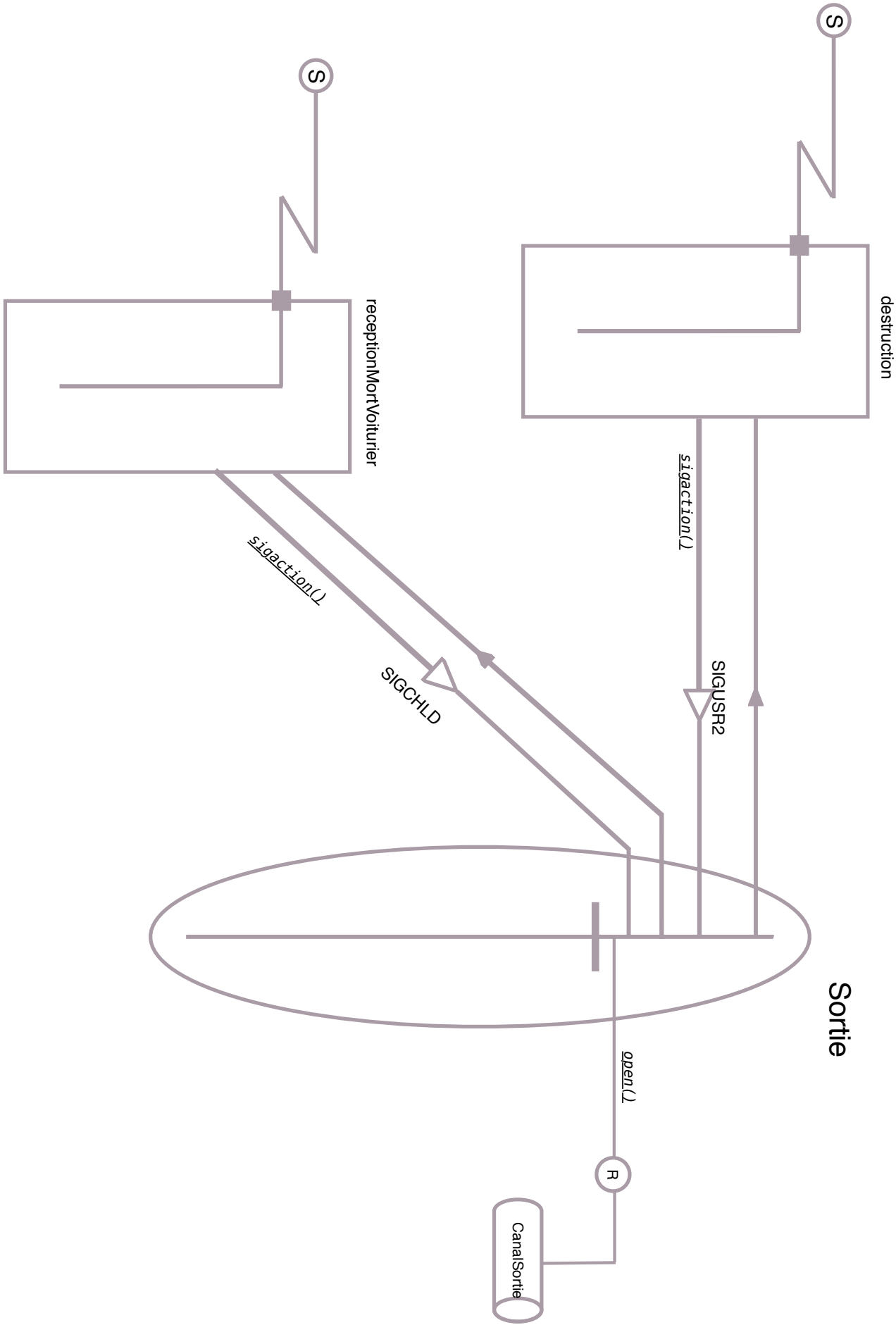
1



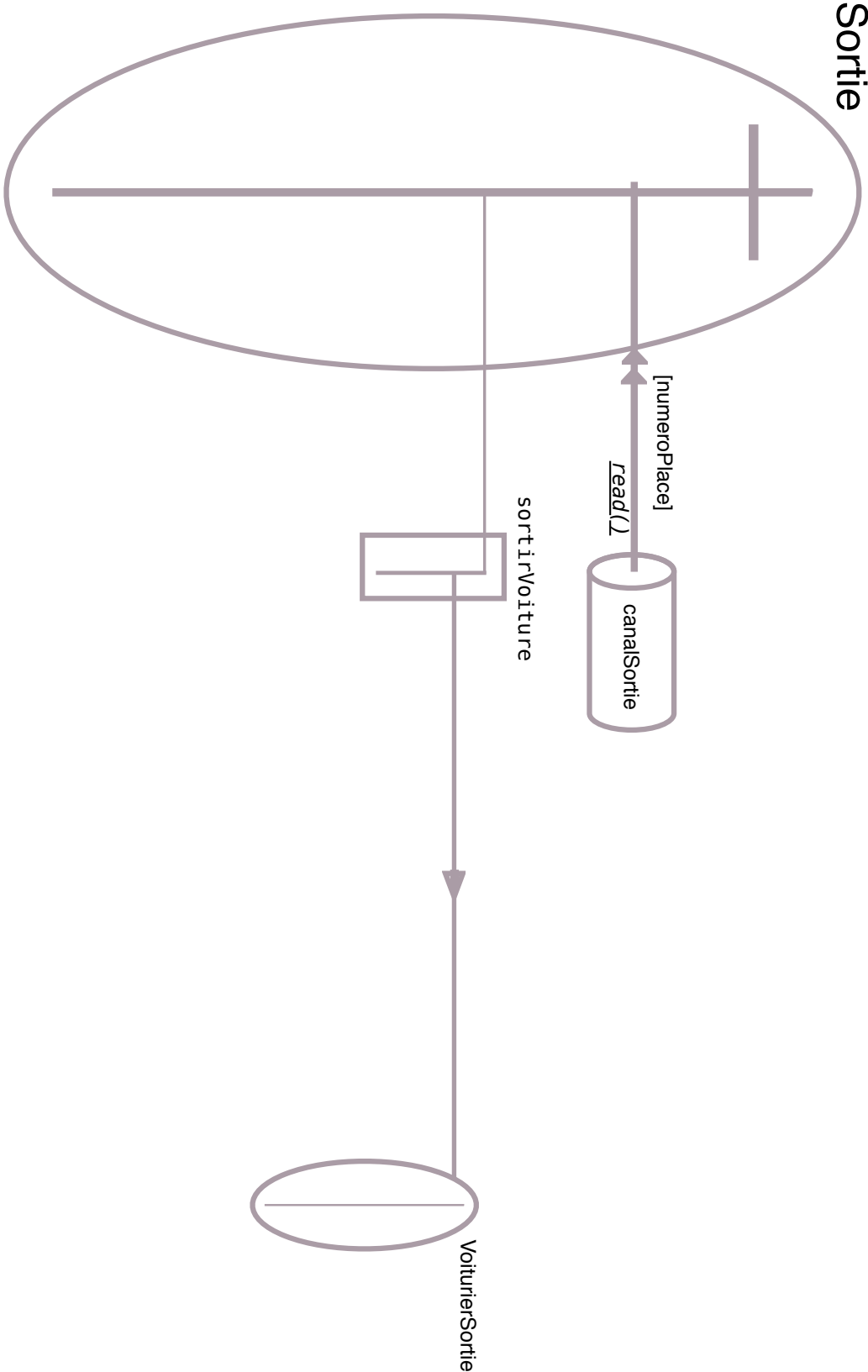
1

MutexMemoirePartagee

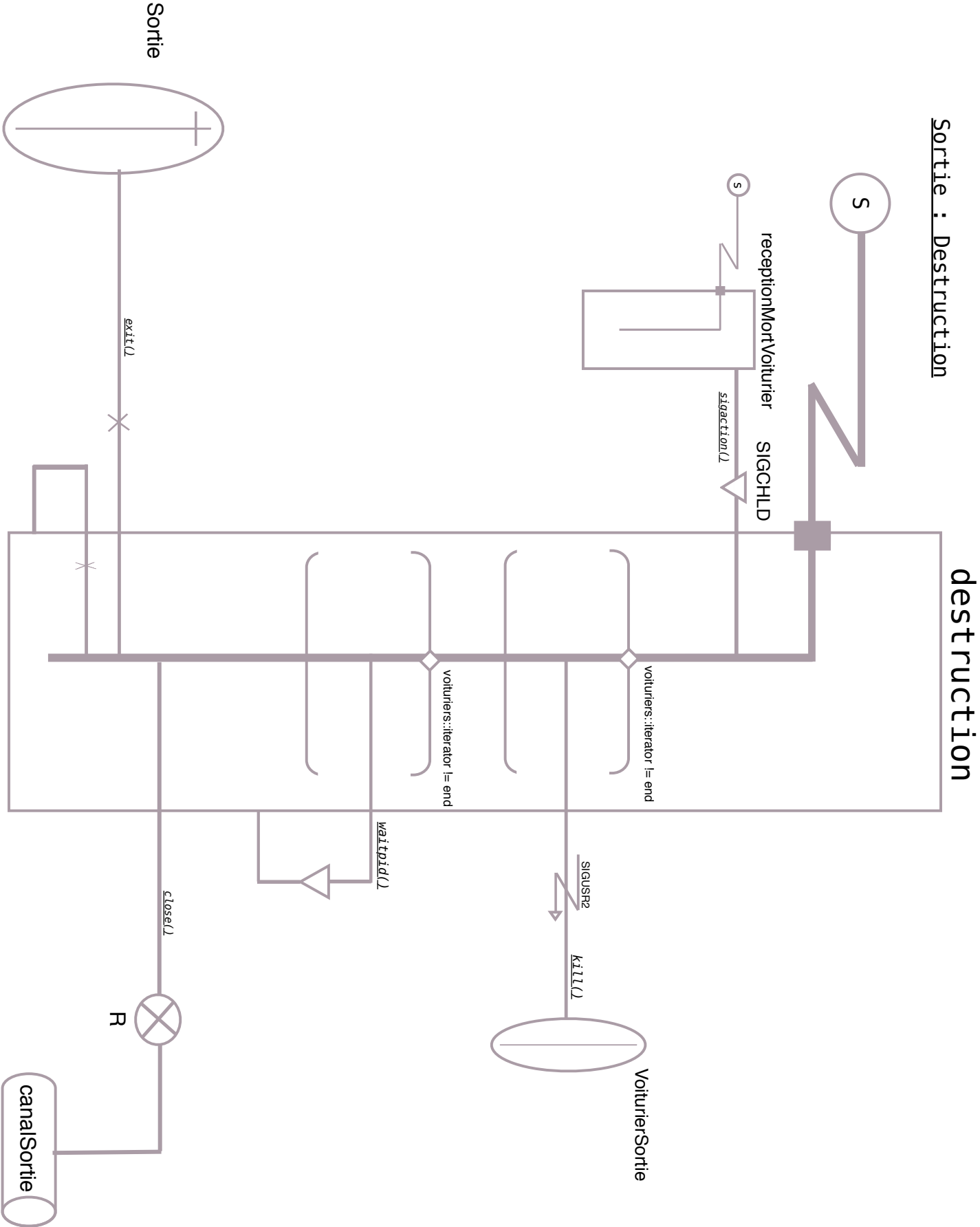
Sortie : Initialisation

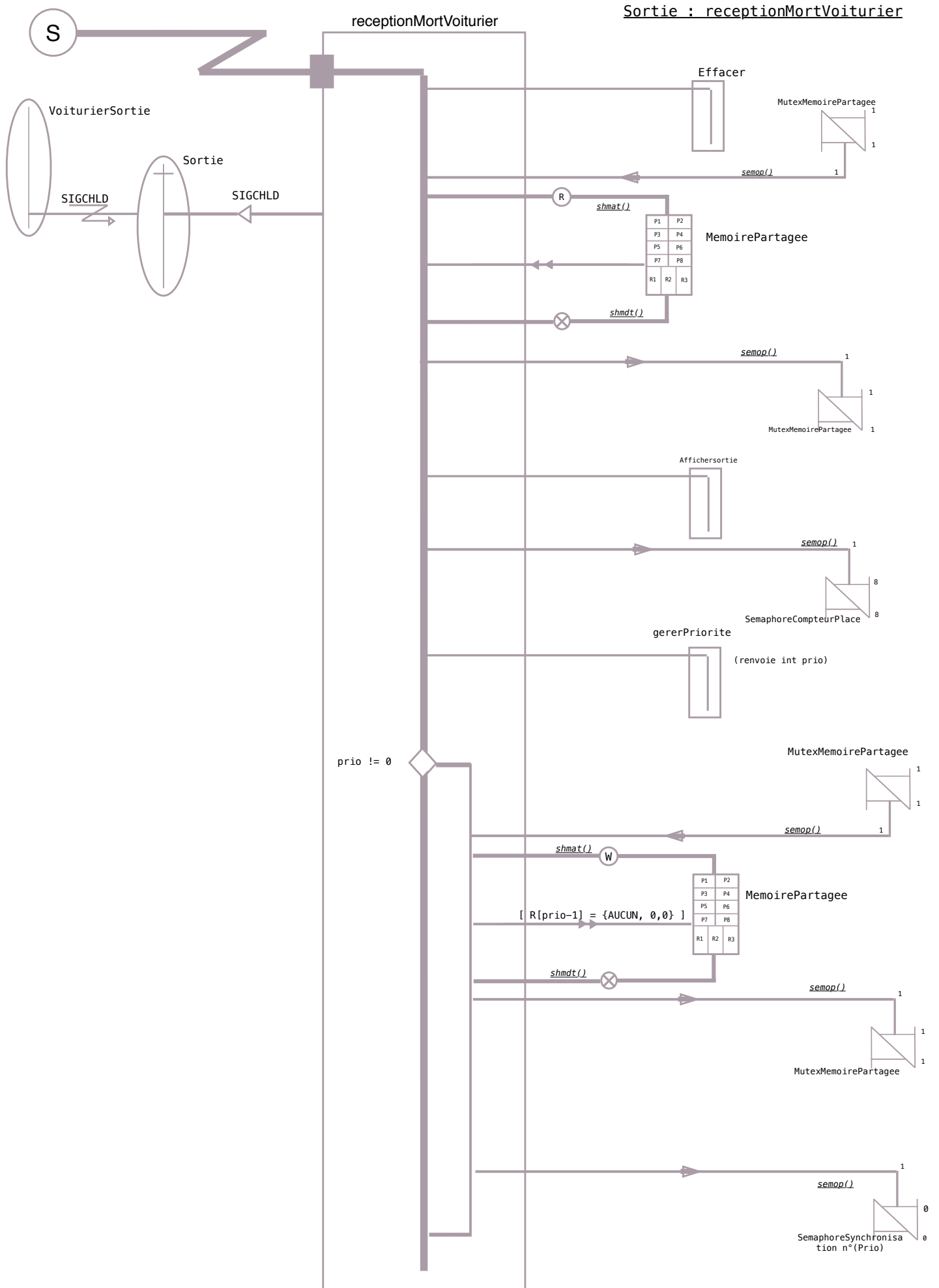


Sortie : Moteur

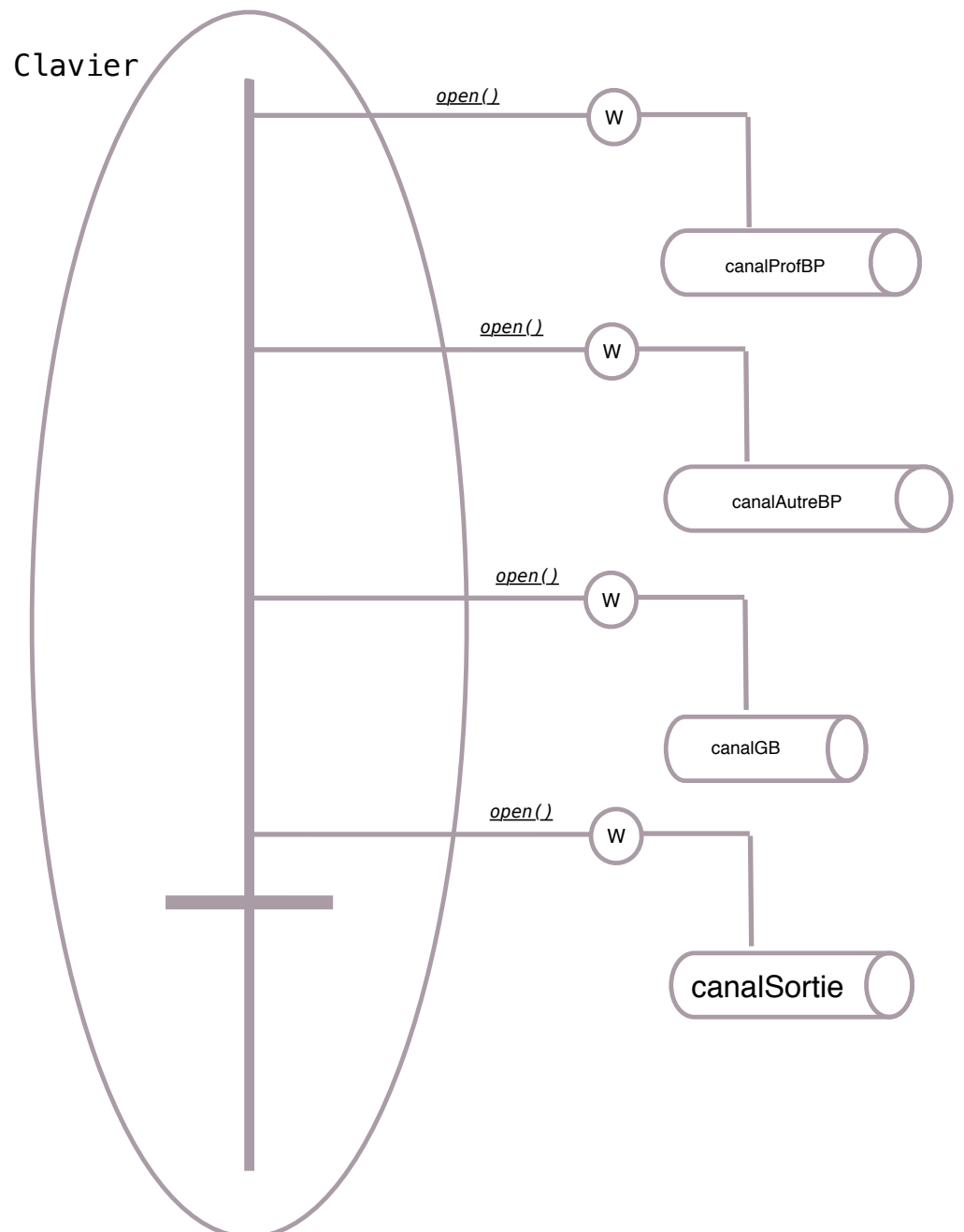


Sortie : Destruction

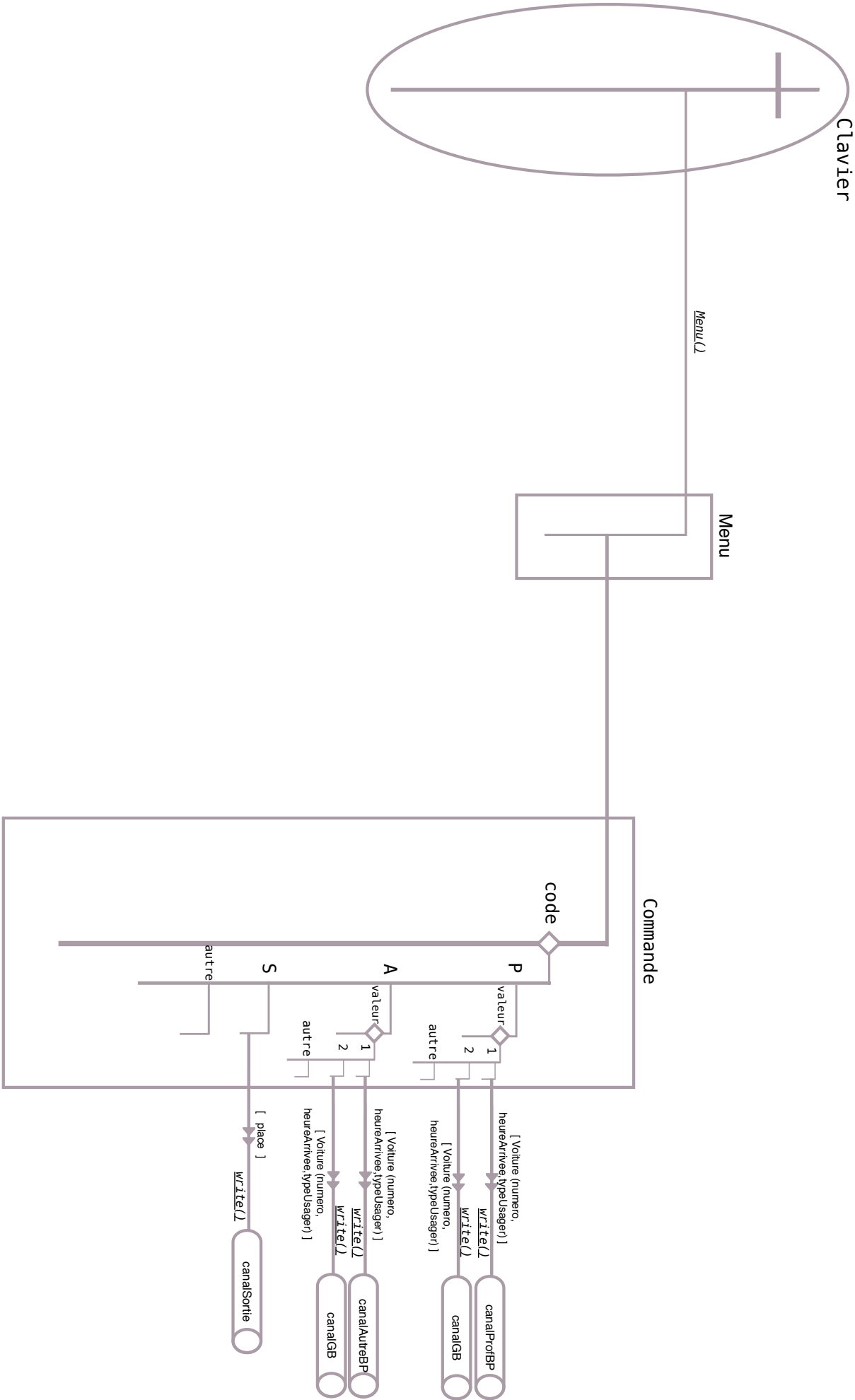




## Clavier : Initialisation



Clavier : Moteur



## Clavier : Destruction

