Waste Incinerator Service

Requirements Analysis

Structure

analyzing the natural language requirements text we found out the following entities that should be somehow modelled:

- ServiceArea
  - Home
  - BurnIn port
  - BurnOut port
  - WasteIn
  - AshOut
- OpRobot
- DDRRobot
- WIS
- Incinerator
- WasteStorage
  - Scale
  - RP
  - WRP
- AshStorage
  - MonitoringDevice
  - Sonar
  - Led

Interaction and Behaviour

By requirements we inferred the following informations that need to be modelled:

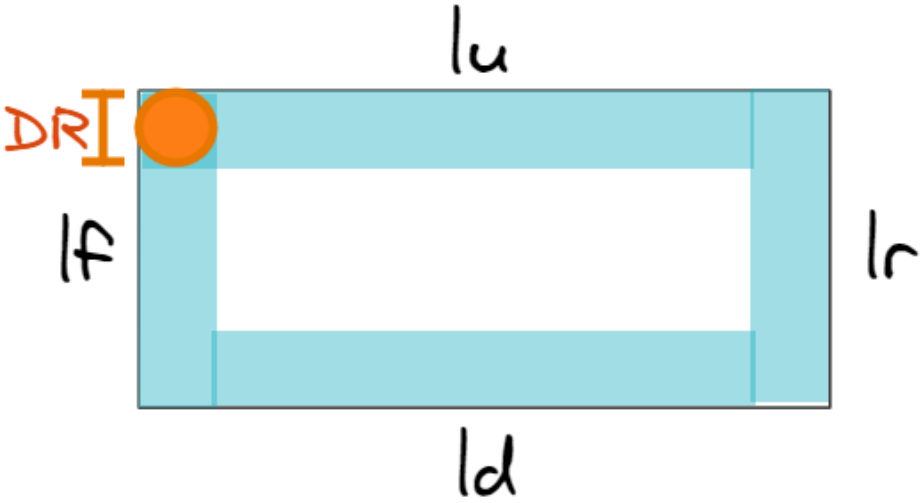
Information	Source	Destination	Description
activationCommand	unspecified	Incinerator	
isBurning	Incinerator	unspecfied	
ashLevel	Sonar	unspecified	
loadRP	WasteStorage	OpRobot	
unloadRp	OpRobot	Incinerator	
loadAsh	Incinerator	OpRobot	
unloadAsh	OpRobot	AshStorage	

> [!NOTE]
> we merged Interactions and Behaviour sections because at this stage of the project for the majority of this informations we don't know yet if they will be modelled as POJOs' methods or messages between actors

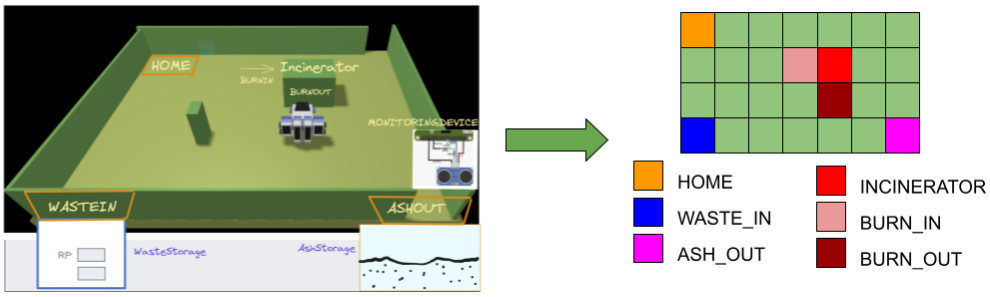
Service Area Model

the **ServiceArea** is modelled as an Euclidean space delimited by its edges(similar to what has been done in the BoundaryWalk and RobotCleaner projects):

- the **perimeter edge** has length `lf+ld+lr+lu`
- being the ServiceArea rectangular we have `lf=lr && ld==lu`
- we define `DR=2R` being `R` the radius of the DDRRobot circumscribable circle

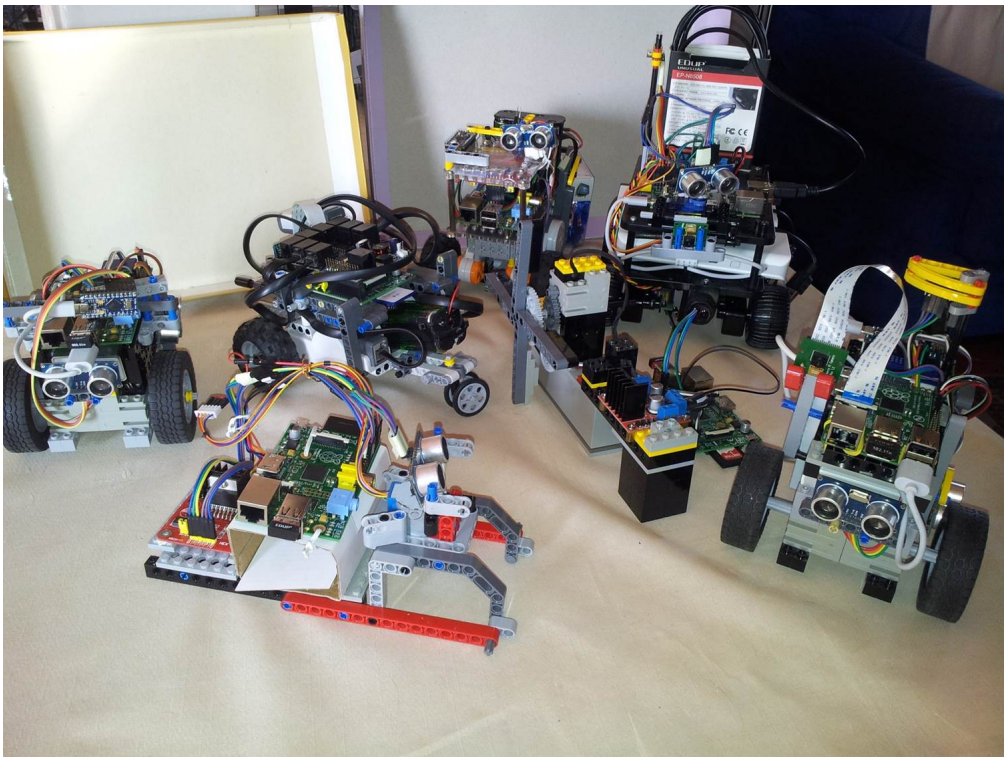


Given this model we have that **Home**, **BurnIn**, **BurnOut**,**WasteIn**, **AshOut** are all modelled as cells in the serviceArea:



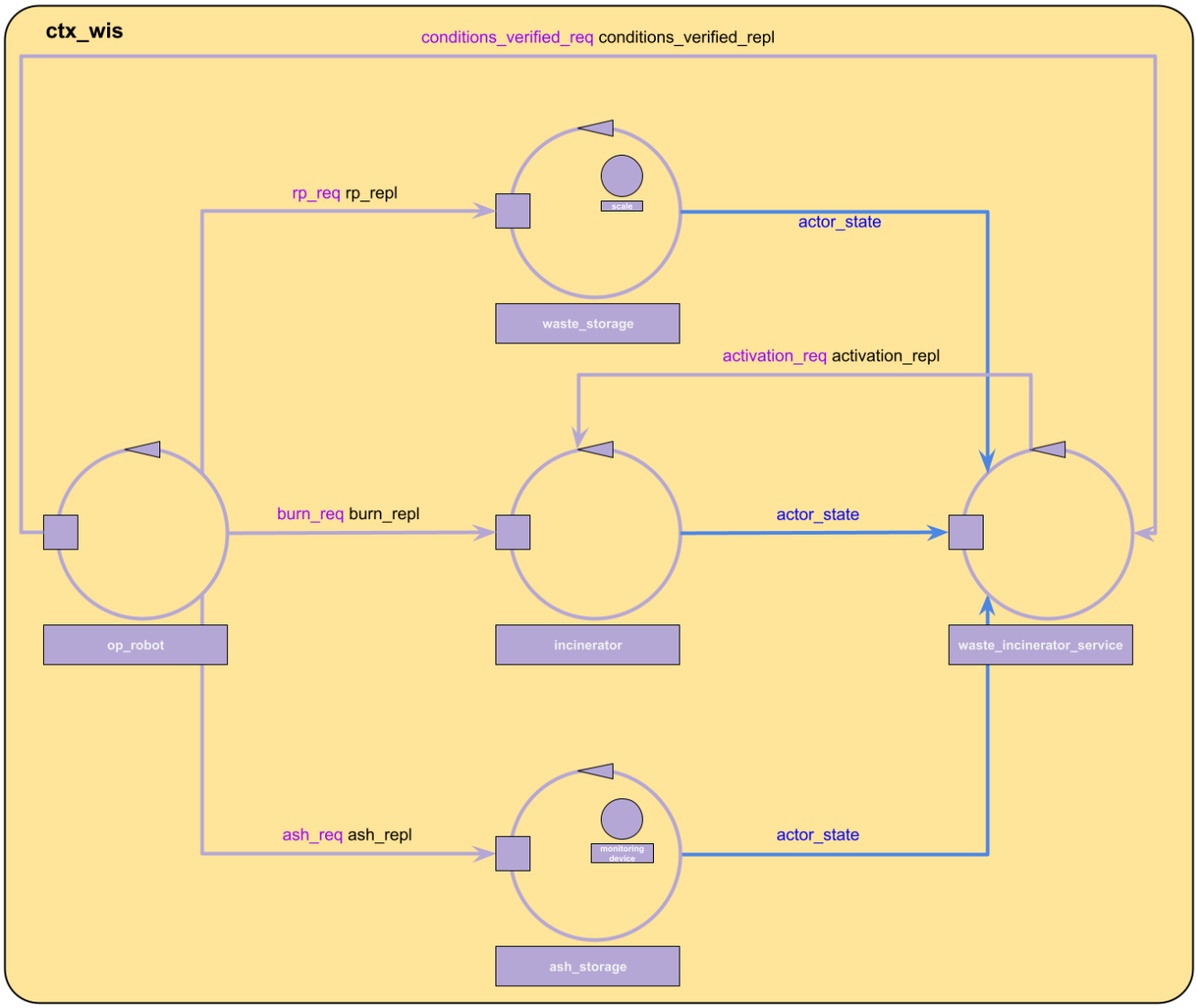
DDRRobot model

The **OpRobot**, defined in the requirements as the robot controlled by the WIS, makes use of a DDRRobot (and its control software) given by the customer, we link the detailed definition of DDRRobot and its qak control software.



WIS, WasteStorage, Incinerator and AshStorage models

WasteStorage, **Incinerator** and **AshStorage** need to exchange messages by requirements so they are modelled as Actors.
Scale can be modelled both as an actor or a POJO inside WasteStorage, for now we will represent it as a Pojo and **demand the discussion to another moment**.
The same can be said for **MonitoringDevice**, **Sonar** and **Led**, that will be for now modelled as POJOs inside AshStorage.
In the first prototype of the model we will have the OpRobot contain the majority of the buisness, asking **WIS** for the permission to start a cycle.
WIS will then act as an observer of WasteStorage,AshStorage and Incinerator,
checking that the constraints are satisfied.
However by requirements there are **no specification** about where to inject the businsess logic (injecting all buisness logic into WIS could also be an option), so we **demand the discussion to the Problem Analysis**.
The following diagram illustrate the structure of what has been produced basing on requirements:


ctx_wis
conditions_verified_req conditions_verified_repl
rp_req rp_repl
scale
actor_state
waste_storage
activation_req activation_repl
burn_req burn_repl
actor_state
op_robot
incinerator
waste_incinerator_service
ash_req ash_repl
monitoring device
actor_state
ash_storage