Deliverable 2 – Fully Dressed Format + System Sequence Diagram + Operation Contracts

Group – Members:

Khumam Alzaeem – 85319

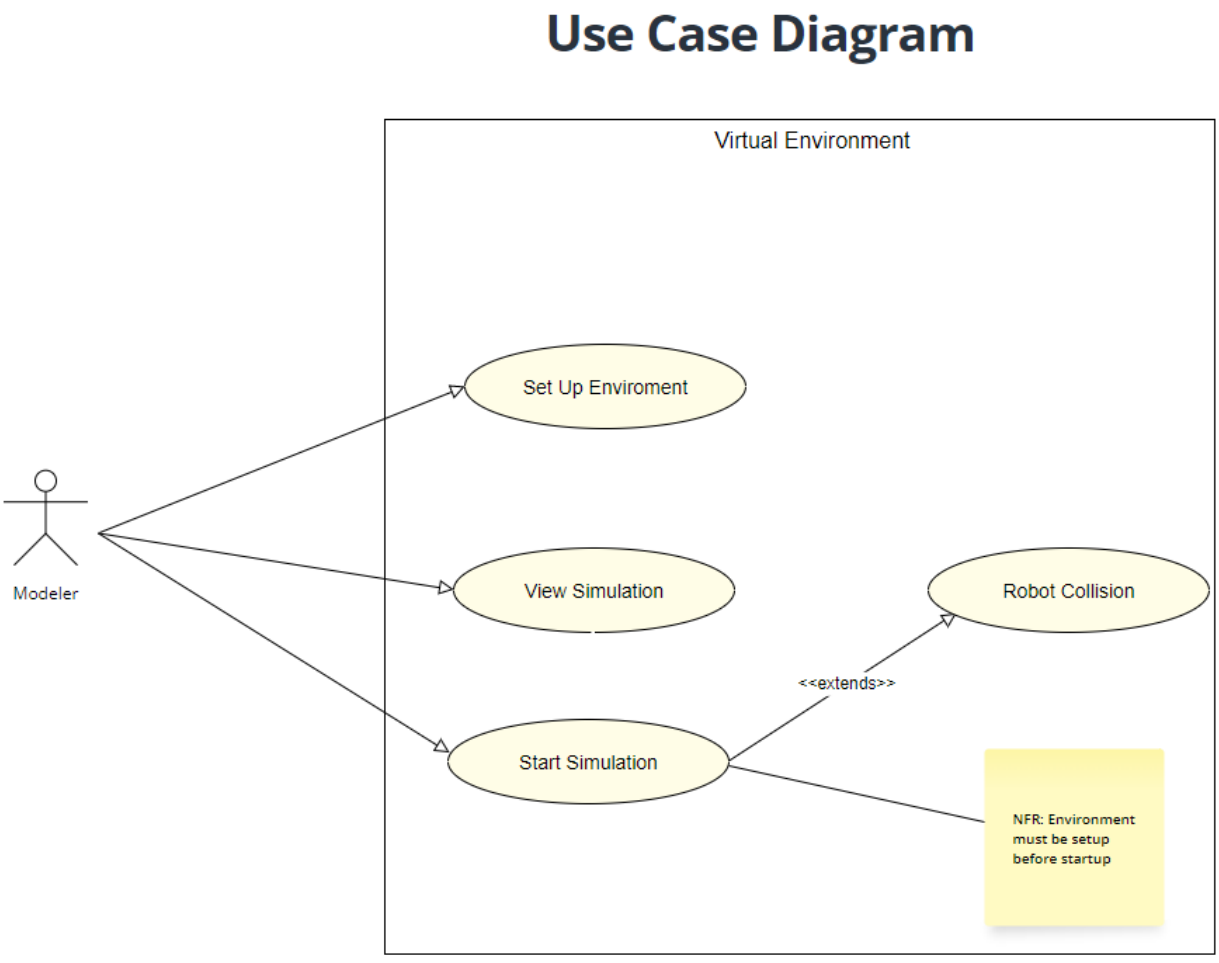
Ibrahim Alsaffarini – 86362

Sultan Al Nuaimi – 86011

Youssef Elmadany – 87790

Table of Contents:

*Updated Use Case Diagram:



1. Fully Dressed Use Cases:

Use case: Setup_Environment
ID: Base 1
Brief Description: User is prompted to enter the initial values to setup the environment
Primary Actors: Modeler
Secondary Actor: None
PreConditions: The program executable should be opened
Main Flow: 1.Modeler starts the program 2.System asks for number of robots and exits 3.Modeler Inputs environment grid size, number of robots, and number of exits. 4.System initializes Grid, random Robot starting locations, 5.For every exit 5.1.System asks for exit location on the grid 5.2.Modeler inputs exit location 5.3.System initializes the exit location 6.System generates success message
Extensions: 5.2: Modeler Inputs invalid exit location 1. System pops error message and prompts the user to re-enter exit location 5.2a: out of bounds of the environment size. 1. Modeler Inputs new location within the environment size 5.2b: Modeler Inputs exit location that already exists in the environment. 1. Modeler inputs new, unique location.
Postcondition:

Environment setup and ready for simulation

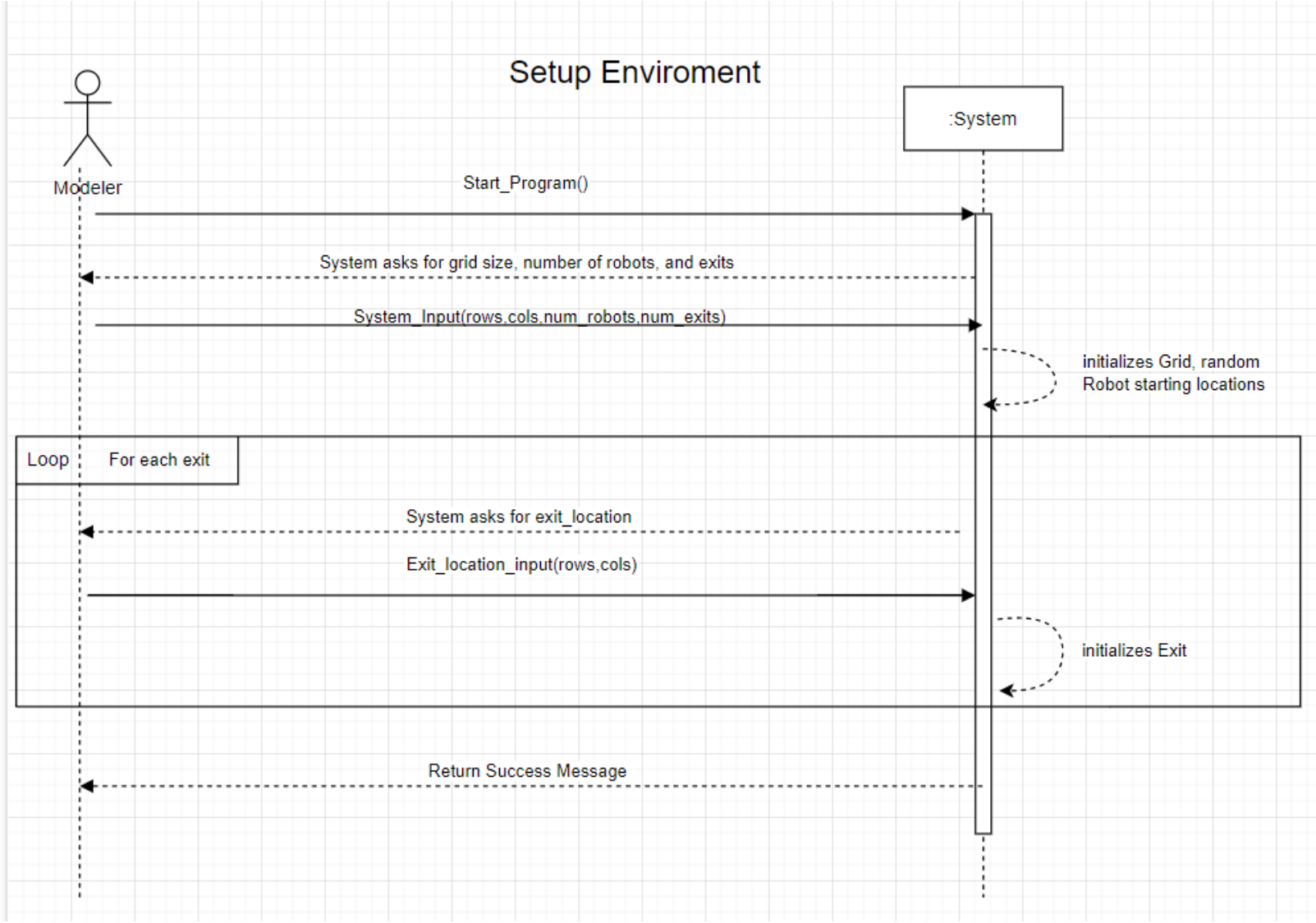
Use case: Start Simulation
ID: Base 2
Brief Description: Starts the simulation by placing the exits and and running the simulation until all robots find their nearest exit.
Primary Actors: Modeler
Secondary Actor: None
PreConditions: Environment must be setup before starting simulation
Main Flow: 1.Modeler starts the simulation 2.For each robot: 2.1System calculates nearest exit for each robot 3.While robots are still in the simulation 3.1 For each robot in 1 Cycle: 3.1.1 Set direction() 3.1.2 Check collision() Extension point: Collision, Extension: Stop Collision 3.1.3 Set speed() 3.1.4 Move robot() 3.2 If robot reached exit remove robot from simulation 4.System ends the simulation
Extensions: 1.stop Collision
Postcondition: Simulation runned successfully!

Extension Use case: stop Collision
ID: Extend 2.1
Brief Description: The system checks if there exists a robot in front of current robot path
Primary Actors: System
Secondary Actor: none
PreConditions: Simulation must be running
Main Flow: 1.System checks if there is a robot in front of the current robots path 2.if there exists robot in the current robot movement path 2.1System checks distance between the current robot and the obstacle robot 2.2System decreases speed according to the distance
Postcondition: Collision checked successful,the current robot speed updated

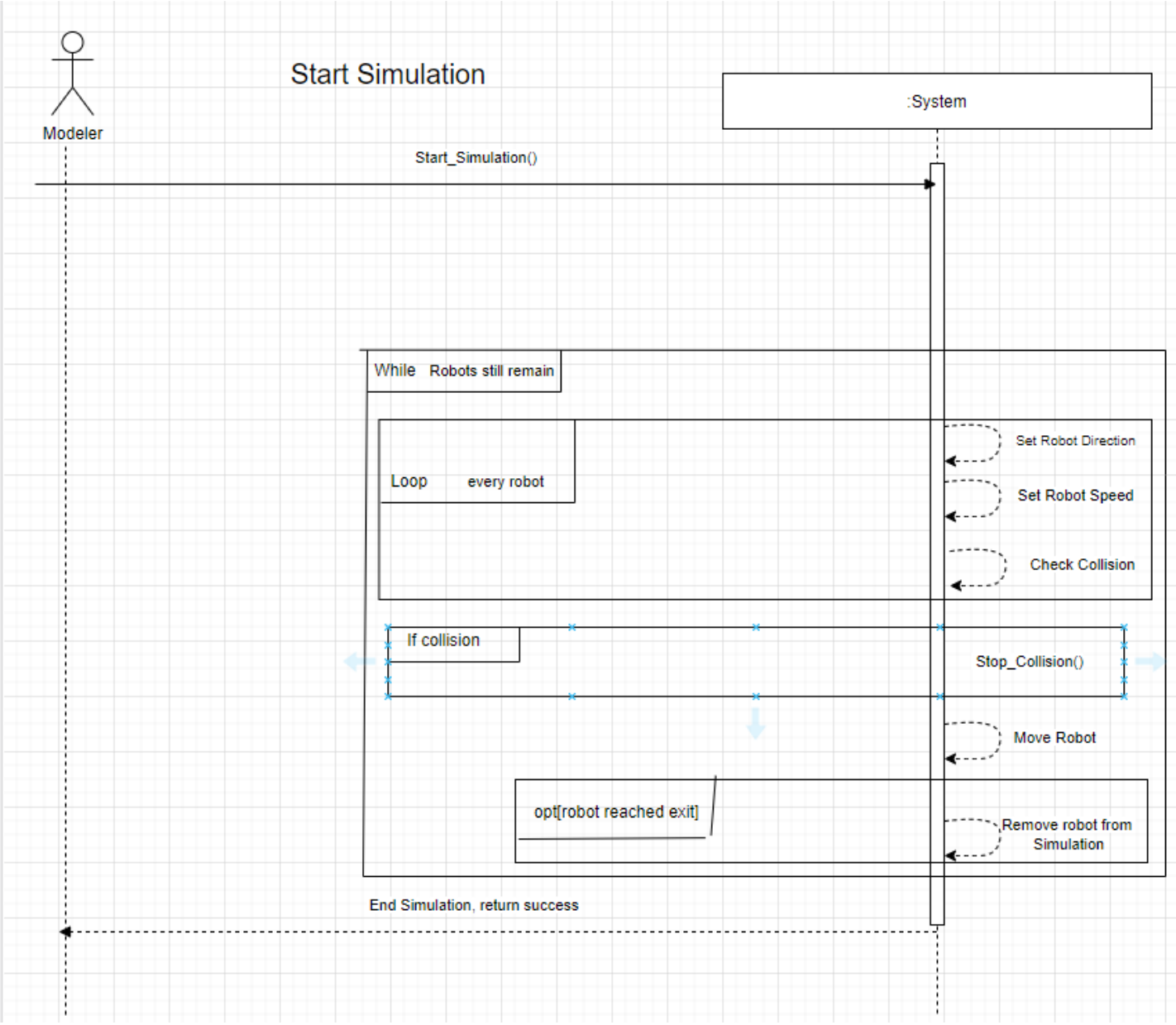
2 - System Sequence Diagram:

Note: clearer images found on pdf file.

Setup Environment:

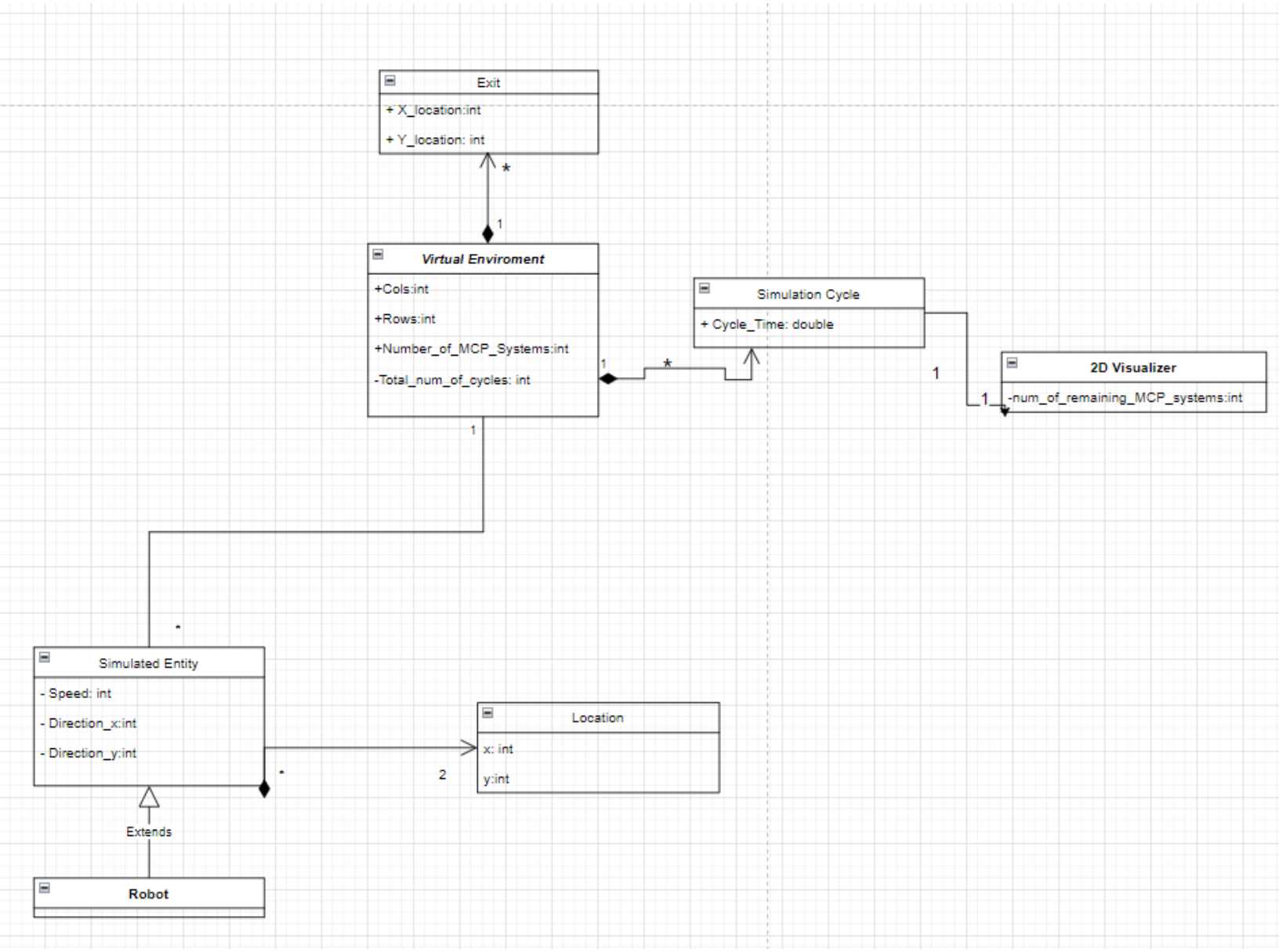


Start Simulation:



3 - Operation Contracts:

New Domain Model:



Setup SSD:

Contract 1: Start Program
Operation: StartProgram()
Cross References: Setup Environment
PreCondition: none
PostCondition: - System starts and requests environment Info - Instance of virtual environment v created

Contract 2: System Input
Operation: System_input(rows,cols,num_of_robots, num_exits)
Cross References: Setup Environment
PreCondition: Program started
PostCondition: - v.rows = rows - v.cols = cols - v.Number_of_MCP_Systems = num_of_robots - An N number of Robot entities are created where N = num_of_robots - An N number of Exit entities are created where N = num_exits

Contract 3: Exit Locations
Operation: Exit_Location_input(rows,cols)
Cross References: Setup Environment
PreCondition: Exits instance created
PostCondition: - Exit Instance S attributes are modified where - S.X_location = cols - S.Y_location = rows

Start SSD:

Contract 1: Start_Simulation
Operation: Start_Simulation ()
Cross References: Start Simulation
PreCondition: Environment setted up
PostCondition: - A Simulation Cycle instance S created . - S.Cycle_time is populated with the total run time - All robots exited the simulation - Success message printed - Program terminated