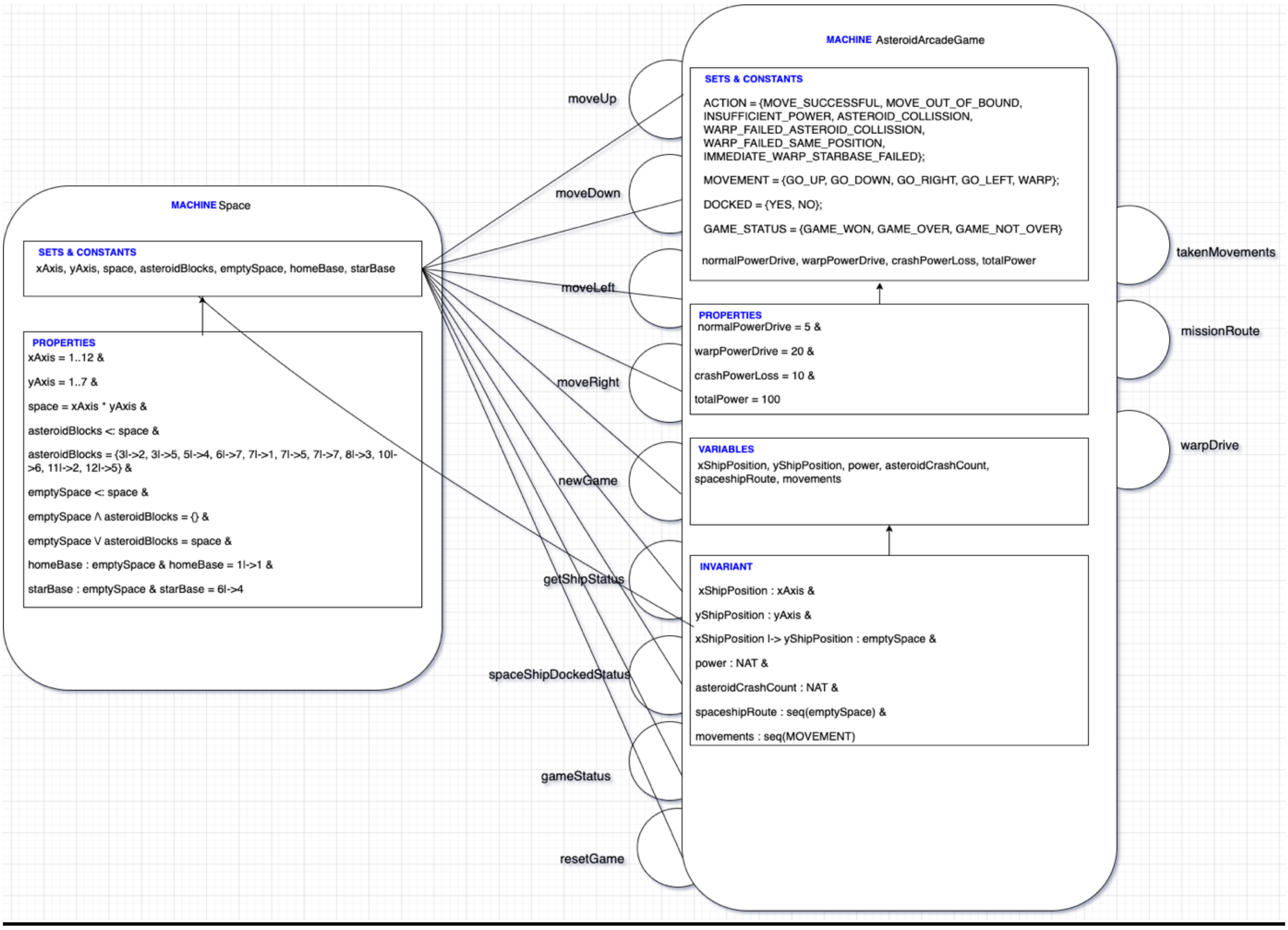
Name – Vathila De Silva
IIT No – 20200765
UoW No – w1833511



MACHINE Space

SETS & CONSTANTS

xAxis, yAxis, space, asteroidBlocks, emptySpace, homeBase, starBase

PROPERTIES

```
xAxis = 1..12 &
yAxis = 1..7 &
space = xAxis * yAxis &
asteroidBlocks <: space &
asteroidBlocks = {3l->2, 3l->5, 5l->4, 6l->7, 7l->1, 7l->5, 7l->7, 8l->3, 10l->6, 11l->2, 12l->5} &
emptySpace <: space &
emptySpace /\ asteroidBlocks = {} &
emptySpace \/ asteroidBlocks = space &
homeBase : emptySpace & homeBase = 1l->1 &
starBase : emptySpace & starBase = 6l->4
```

MACHINE AsteroidArcadeGame

SETS & CONSTANTS

```
ACTION = {MOVE_SUCCESSFUL, MOVE_OUT_OF_BOUND,
INSUFFICIENT_POWER, ASTEROID_COLLISSION,
WARP_FAILED_ASTEROID_COLLISSION,
WARP_FAILED_SAME_POSITION,
IMMEDIATE_WARP_STARBASE_FAILED};

MOVEMENT = {GO_UP, GO_DOWN, GO_RIGHT, GO_LEFT, WARP};

DOCKED = {YES, NO};

GAME_STATUS = {GAME_WON, GAME_OVER, GAME_NOT_OVER}

normalPowerDrive, warpPowerDrive, crashPowerLoss, totalPower
```

PROPERTIES

```
normalPowerDrive = 5 &
warpPowerDrive = 20 &
crashPowerLoss = 10 &
totalPower = 100
```

VARIABLES

xShipPosition, yShipPosition, power, asteroidCrashCount, spaceshipRoute, movements

INVARIANT

```
xShipPosition : xAxis &
yShipPosition : yAxis &
xShipPosition l-> yShipPosition : emptySpace &
power : NAT &
asteroidCrashCount : NAT &
spaceshipRoute : seq(emptySpace) &
movements : seq(MOVEMENT)
```

Explanation of the state invariants in the system

The asteroid game is made up of 2 machines.

1. Space – For definition of space and its boundaries
2. AsteroidArcadeGame – For definition of the asteroid game, its states, operations. AsteroidArcadeGame SEES Space

These machines consist of 4 enumerated SETS,

```
ACTION = {MOVE_SUCCESSFUL, MOVE_OUT_OF_BOUND,
INSUFFICIENT_POWER, ASTEROID_COLLISSION,
WARP_FAILED_ASTEROID_COLLISSION, WARP_FAILED_SAME_POSITION,
IMMEDIATE_WARP_STARBASE_FAILED}
```

to define the alert messages that are displayed,

```
GAME_STATUS = {GAME_WON, GAME_OVER, GAME_NOT_OVER}
```

to define the status of the game,

```
MOVEMENT = {GO_UP, GO_DOWN, GO_RIGHT, GO_LEFT, WARP};
```

to define the movement of the spaceship,

```
DOCKED = {YES, NO};
```

to define the status if the spaceship is docked.

The below are the state invariant explanations.

```
xShipPosition : xAxis & yShipPosition : yAxis & xShipPosition
|-> yShipPosition : emptySpace &
```

The movement of the spaceship is restricted to the boundaries of the space and it can only travel through areas that are void of any obstacles.

```
power : NAT & asteroidCrashCount : NAT &
```

The ship always has a minimum amount of power, which can never be below zero. The number of collisions has to be a non-zero full number.

```
spaceshipRoute : seq(emptySpace) & movements : seq(MOVEMENT)
```

The space path that has been visited must be limited within the boundaries of the space.