**Data**

Map<Int, Boolean>StandPositions

Int NumKitsToRequest //Initially 0

Kit CurrentlyHeldKit

//Shared data with Stand

List<MyKit> MyKits

Stand stand

Conveyor conveyor

Camera camera

GUIKitRobot guiKitRobot

//Prevent collisions (not for v0)

//Shared with PartsRobot.

Semaphore AccessKit

Semaphore Animation

**Messages**

//From Conveyor

HereIsKit(Kit k){

MyKit = k;

k.KS = PickedUp;

NumKitsToRequest--;

}

//From Stand

NeedKit(int location){

StandPositions.put(location, true);

NumKitsToRequest++;

}

//From Stand

MoveKitToInspectionArea(Kit k){

k.KS = MarkedForInspection;

}

//From Camera

KitPassedInspection(){

MyKits.get(0).KS = Inspected;

}

//From GUI

placeKitOnConveyorDone(){

Animation.release();}

placeKitInInspectionAreaDone(){

Animation.release();}

placeKitOnStandDone(){

Animation.release();}**Scheduler**

If Ǝ k in MyKits ϵ

k.KS = PickedUp PlaceMyKitOnStand()

If Ǝ k in MyKits ϵ

k.KS = MarkedForInspection

call PlaceKitInInspectionArea(k)

If Ǝ k in MyKits ϵ

k.KS = Inspected

call ShipKit(k)

if(NumKitsToRequest > 0)

call RequestKit()

**Actions**

RequestKit(){

Conveyor.NeedKit();

}

PlaceMyKitOnStand(){

//Find int i in //StandPositions.keys() ϵ //StandPositions.get(int) = true

Animation.acquire();

guiKitRobot.placeKitOnStand(i);

StandPositions.put(i, false);

Stand.HereIsKit(MyKit, i);

}

PlaceKitInInspectionArea(Kit k){

Animation.acquire();

guiKitRobot.placeKitInInspectionArea(k);

k.KS = AwaitingInspection;

camera.InspectKit(k);

}

ShipKit(Kit k){

Animation.acquire();

guiKitRobot.placeKitOnConveyor();

conveyor.TakeKitAway(k);

stand.ShippedKit();

myKits.remove(k);

}