**Data**

Map<Int, Boolean>StandPositions

Int NumKitsToRequest //Initially 0

Kit MyKit

//Shared data with Stand

List<Kit> KitsOnStand

Stand stand

Conveyor conveyor

Camera camera

//Prevent collisions (not for v0)

//Shared with KitRobot.

Semaphore AccessKit

**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(){

KitsOnStand.get(0).KS = Inspected;

}**Scheduler**

if(MyKit != null)

if(MyKit.KS = PickedUp) PlaceMyKitOnStand()

If Ǝ k in KitsOnStand ϵ

k.KS = MarkedForInspection

call PlaceKitInInspectionArea(k)

If Ǝ k in KitsOnStand 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

DoPlaceKitOnStand(i);

StandPositions.add(i, false);

Stand.HereIsKit(MyKit, i);

}

PlaceKitInInspectionArea(Kit k){

DoPlaceKitInInspectionArea(k);

k.KS = AwaitingInspection;

camera.InspectKit(k);

}

ShipKit(Kit k){

DoPlaceKitOnConveyor();

conveyor.TakeKitAway(k);

stand.ShippedKit();

}

DoPlaceKitOnStand(int location){

//Animation}

DoPlaceKitInInspectionArea(Kit k){//Animation}

DoPlaceKitOnConveyor(){

//Animation}