**Camera Agent**

**Data:**

Kit kit;

enum kitStatus = {NOT\_READY,DONE, MESSAGED};

kitStatus kitState = kitStatus.NOT\_READY;

boolean kitDone;

List<MyNest> nests;

class MyNest {

NestAgent nest;

PartType type;

List<GUIPart> guiParts;

enum status = {NOT\_READY, READY, PHOTOGRAPHED };

status state;

MyNest(NestAgent nest, PartType type) {

this.nest = nest;

this.type = type;

this.state = status.NOT\_READY;}

}

**Messages:**

InspectKit(Kit kit) {

this.kit = kit;

kitStatus kitState = kitStatus.NOT\_READY;

stateChanged();}

IAmFull(NestAgent nest) {

MyNest nest = new MyNest(nest, nest.type);

nests.add(nest);

stateChanged();}

TakePictureNestDone(List<GUIPart> parts, NestAgent nest) {

for(MyNest nest :nests) {

if(nest.nest = nest) {

nest.guiParts = parts;

state = status.PHOTOGRAPHED;

return; }

}

stateChanged();

}

TakePictureKitDone(boolean done) {

kitDone =done;

kitState = kitStatus.DONE;

stateChanged();}

**Scheduler:**

if there exists MyNest nest in nests

if(nest.state = status.NOT\_READY)

takePictureOfNest(nest);

if there exists MyNest nest in nests

if(nest.state = status.PHOTOGRAPHED)

tellPartsRobot(nest);

if(kit != null && kitState = kitStatus.NOT\_READY)

takePictureOfKit(kit);

if(kit!=null && kitState == kitStatus.DONE)

tellKitRobot();

**Actions:**

takePictureOfNest(MyNest nest) {  
 GUICamera.takePicture(nest);

nest.state = status.READY;

stateChanged(); }

tellPartsRobot(nest) {

List<GUIPart> goodParts = new ArrayList();

for(guiPart part: nest.guiParts) {

if(part.isGood())

goodParts.add(part);

}

PartsRobotAgent.msgHereAreGoodParts(Map<nest.nest, goodParts>);

nests.remove(nest);

stateChanged(); }

takePictureOfKit() {

GUICamera.takePicture(kit);

stateChanged(); }

tellKitRobot() {

KitRobotAgent.msgKitPassedInspection(done);

kitState = kitStatus.MESSAGED;

stateChanged(); }