**GantryAgent**

**Data:**

List<List<Bins>> binList;

List<PartType> requestedParts;

**Messages:**

HereIsBinConfig(List<Bin> bins) {

binList.add(bins);

stateChanged();}

INeedParts(PartType type) {

requestedParts.add(type);

stateChanged();}

receiveBinDone(bin) {

bin.state = OVER\_FEEDER;

stateChanged();}

dropBinDone(bin) {

bin.state = EMPTY;

stateChanged(); }

removeBinDone() {

binList.remove(bin);

stateChanged();}

**Scheduler:**

if there exists a "requested" in requestedParts

for(List<Bin> bin : binList)

if(bin.type == requested && bin.state == FULL)

moveToFeeder(bin);

if there exists a "requested" in requestedParts

for(List<Bin> bin : binList)

if(bin.type == requested && bin.state == OVER\_FEEDER)

fillFeeder(bin);

if there exists a "requested" in requestedParts

for(List<Bin> bin : binList)

if(bin.type == requested && bin.state == EMPTY)

discardBin(bin);

**Actions:**

moveToFeeder(bin) {

bin.state = MOVING;

GUIGantry.receiveBin(bin);

stateChanged();}

fillFeeder(bin) {

FeederAgent.msgHereAreParts(bin.part);

bin.state = FILLING\_FEEDER;

GUIGantry.dropBin(bin, bin.feeder);

stateChanged();}

discardBin(bin) {

bin.state = DISCARDING;

GUIGangry.removeBin(bin);

stateChanged();}

**FeederAgent**

**Data:**

List<PartType> requestList;

List<Part> currentParts

**Messages:**

INeedPart(PartType type) {

requestList.add(type)

stateChanged(); }

HereAreParts(Part p) {

p.status == IN\_FEEDER;

currentParts.add(p);

stateChanged(); }

givePartToDiverterDone(part) {

part.status ==END\_DIVERTER;

stateChanged(); }

givePartToLaneDone(part) {

stateChanged(); }

**Scheduler:**

if there exists requestedType in requestList

if(requestedType.status == PENDING)

getParts(requestedType);

if there exists part in currentParts

if(part.status == IN\_FEEDER)

giveToDiverter(part);

if there exists part in currentParts

if(part.status == END\_DIVERTER)

giveToLane(part);

**Actions:**

getParts(requestedType) {

GantryAgent.msgINeedParts(requestedType)

stateChanged(); }

giveToDiverter(part) {

GUIDiverter.face(part.laneOrientation);

GUIFeeder.givePartToDiverter(part);

part.state = IN\_DIVERTER;

stateChanged(); }

giveToLane(part) {

LaneAgent.msgHereIsPart(part);

GUIDiverter.givePartToLane(part);

currentParts.remove(part);

stateChanged();

**LaneAgent**

**Data:**

List<PartType> requestList;

List<Part> currentParts

**Messages:**

INeedPart(Part p) {

requestList.add(type)

stateChanged(); }

HereIsPart(Part p) {

p.status == IN\_LANE;

currentParts.add(p);

stateChanged(); }  
receivePartDone(part) {

stateChanged(); }

givePartToNestDone(part) {

currentParts.remove(part);

stateChanged(); }

**Scheduler:**

if there exists requestedType in requestList

if(requestedType.status == PENDING)

getParts(requestedType);

if there exists part in currentParts

if(part.status == IN\_LANE)

giveToNest(part);

**Actions:**

getParts(requestedType) {

FeederAgent.msgINeedParts(requestedType)

stateChanged(); }

giveToNest(part) {

GUILane.givePartToNest(part)

NestAgent.msgHereIsPart(part);

stateChanged(); }

**NestAgent**

**Data:**

List<PartType> requestList;

List<Part> currentParts

int count = 0;

int full = 9;

boolean takingParts = false;

**Messages:**

HereIsPartType(PartType type) {

GUINest.purge();

requestList.clear();

requestList.add(type);

stateChanged();}

HereIsPart(Part p) {

p.status = IN\_NEST;

currentParts.add(p);

stateChanged(); }

TakingPart(Part p) {

GUINest.givePartToPartsRobot(p);

currentParts.remove(p);

stateChanged();

DoneTakingParts() {

takingParts = false;

stateChanged();

receivePartDone()

givePartToPartsRobotDone()

purgingDone()

**Scheduler:**

if there exists requestedType in requestList

if(requestedType.status == PENDING && count < full)

getParts(requestedType);

if there exists part in currentParts

if(part.status == IN\_NEST)

moveToPosition(part);

if(count == full) {

nestFull(); }

if(takingParts ==true) {

updateParts(); }

**Actions:**

getParts(requestedType) {

LaneAgent.msgINeedPart(requestedType);

stateChanged();}

moveToPosition(part) {

GUINest.receivePart(part);

part.status = IN\_NEST\_POSTION;

stateChanged();

nestFull() {

CameraAgent.msgIAmFull(requestedType);

takingParts = true;

stateChanged();

updateParts() {

GUINest.updatePartsList(); }