//Popup pseudo-code

Data:

ConveyorFamily cf, cf2;

// top and bottom workstation

WorkStation top, bot;

List<Glass> glasses;

List<Glass> waitingGlasses;

List<Glass> doneGlasses;

String name;

boolean raise = false;

boolean sensorOccupied = false;

PopupState popupState = PopupState.NULL;

WorkStationState wState = WorkStationState.EMPTY;

enum PopupState {

NULL, WORKING\_ON\_GLASS, RAISED, GLASS\_ARRIVED, SENDING\_GLASS\_TO\_SENSOR, SENSOR\_EMPTY, SENSOR\_OCCUPIED

}

enum WorkStationState {

NULL, TOP\_WORKSTATION\_OCCUPIED, BOTH\_WORKSTATION\_OCCUPIED, BOT\_WORKSTATION\_OCCUPIED, EMPTY

}

Messages:

msgGlassDone(WorkStation work, Glass glass) {

if (work.getName() == "Top"

&& wState == WorkStationState.BOTH\_WORKSTATION\_OCCUPIED) {

wState = WorkStationState.BOT\_WORKSTATION\_OCCUPIED;

} else if (work.getName() == "Top"

&& wState == WorkStationState.TOP\_WORKSTATION\_OCCUPIED) {

wState = WorkStationState.EMPTY;

} else if (work.getName() == "Bot"

&& wState == WorkStationState.BOTH\_WORKSTATION\_OCCUPIED) {

wState = WorkStationState.TOP\_WORKSTATION\_OCCUPIED;

} else if (work.getName() == "Bot"

&& wState == WorkStationState.BOT\_WORKSTATION\_OCCUPIED) {

wState = WorkStationState.EMPTY;

}

doneGlasses.add(glass);

glasses.remove(0);//move the first one in the list which is the one first being processed

stateChanged();

}

msgCanISendGlass(Sensor sensor, Glass glass) {

waitingGlasses.add(glass);

stateChanged();

}

msgIAmOccupied(Sensor sensor) {

sensorOccupied = true;

popupState = PopupState.SENSOR\_OCCUPIED;

stateChanged();

}

msgIAmEmpty(Sensor sensor) {

sensorOccupied = false;

popupState = PopupState.SENSOR\_EMPTY;

stateChanged();

}

msgHereIsGlass(Sensor sensor, Glass glass) {

glasses.add(glass);

popupState = PopupState.GLASS\_ARRIVED;

stateChanged();

}

Scheduler:

if (popupState == PopupState.GLASS\_ARRIVED

&& glasses.get(glasses.size() - 1).recipe.getNeedWashing()) {

glassArrived();

return true;

}

if (popupState == PopupState.GLASS\_ARRIVED

&& !glasses.get(glasses.size() - 1).recipe.getNeedWashing()) {

passToNextConveyorFamily();

return true;

}

if (popupState == PopupState.WORKING\_ON\_GLASS && raise

&& glasses.size() > 0) {

DoRaisePopup();

return true;

}

if (popupState == PopupState.SENDING\_GLASS\_TO\_SENSOR) {

notifyNextFamily();

return true;

}

if (popupState == PopupState.SENSOR\_EMPTY) {

// next conveyor family is empty in first sensor

passGlassToNextFamily();

return true;

}

if (popupState == PopupState.SENSOR\_OCCUPIED) {

popupState = PopupState.NULL;// it can do sth else as long as it is waiting for sensor and it is possible

return true;

}

if (waitingGlasses.size() > 0 && popupState == PopupState.RAISED

&& wState != WorkStationState.BOTH\_WORKSTATION\_OCCUPIED) {

// when popup is raised, and there is another glass coming in!!!

doLowerPopup();

return true;

}

if (waitingGlasses.size() > 0 && popupState != PopupState.RAISED

|| waitingGlasses.size() > 0 && popupState == PopupState.RAISED

&& wState == WorkStationState.BOTH\_WORKSTATION\_OCCUPIED) {//when popup is lowered or popup raised and both workstation is occupied

checkStationState();

return true;

}

Methods:

passToNextConveyorFamily() {

popupState = PopupState.SENDING\_GLASS\_TO\_SENSOR;

if (raise)

// fire the event in the code

stateChanged();

}

doLowerPopup() {

popupState = PopupState.WORKING\_ON\_GLASS;

// fire event in the code here

……….

// I checked in the scheduler, if there bot one is empty, just send the

// sensor the msgIAmEmpty()

cf.sensor2.msgIAmEmpty();

top.msgIAmLowered();

bot.msgIAmLowered();

raise = false;

stateChanged();

}

checkStationState() {

if (wState != WorkStationState.BOTH\_WORKSTATION\_OCCUPIED) {

cf.sensor2.msgIAmEmpty();

glasses.add(waitingGlasses.remove(0));

} else {

cf.sensor2.msgIAmOccupied();// tell sensor2 to wait

waitingGlasses.remove(0);

}

stateChanged();

}

passGlassToNextFamily() {

if (doneGlasses.size() > 0)

cf2.sensor1.msgHereIsGlass(this, doneGlasses.remove(0));

else if (glasses.size() > 0)

cf2.sensor1.msgHereIsGlass(this, glasses.remove(0));

popupState = PopupState.NULL;

stateChanged();

}

glassArrived() {

// if(glass.recipe.XXX)

popupState = PopupState.WORKING\_ON\_GLASS;

if (wState != WorkStationState.BOTH\_WORKSTATION\_OCCUPIED)

{

raise = true;

}

// fire event here

stateChanged();

}

setName(String name) {

this.name = name;

}

DoRaisePopup() {

if (wState == WorkStationState.EMPTY) {

top.msgHereIsGlass(this, glasses.get(glasses.size() - 1));

wState = WorkStationState.TOP\_WORKSTATION\_OCCUPIED;

} else if (wState == WorkStationState.TOP\_WORKSTATION\_OCCUPIED) {

bot.msgHereIsGlass(this, glasses.get(glasses.size() - 1));

wState = WorkStationState.BOTH\_WORKSTATION\_OCCUPIED;

} else if (wState == WorkStationState.BOT\_WORKSTATION\_OCCUPIED) {

bot.msgHereIsGlass(this, glasses.get(glasses.size() - 1));

wState = WorkStationState.BOTH\_WORKSTATION\_OCCUPIED;

}

cf.sensor2.msgIAmOccupied();

popupState = PopupState.RAISED;

top.msgIAmRaised();

bot.msgIAmRaised();

stateChanged();

}

notifyNextFamily() {

if (raise) {

raise = false;

// move down the popup when it is still up

}

cf2.sensor1.msgCanISendGlass();

stateChanged();

}

setTopWorkStation(WorkStation work) {

top = work;

}

setBotpWorkStation(WorkStation work) {

bot = work;

}