Pseudo Code for ConveyorAgent

Data:

```
public enum ConveyorStatus {ON, OFF}; // shows status of conveyor

public enum MachineStatus {FREE, BUSY, STALLED};

public enum ConveyorSensorIn {ON, OFF, CHANGE};

public enum ConveyorSensorOut {ON, OFF,STALLED};
public ConveyorStatus status;
public MachineStatus machinestatus;
public ConveyorSensorIn sensor1;
public ConveyorSensorOut sensor2;
private WorkStation station;
private ConveyorFamily family;
private ConveyorFamily backCF;
String name;
private List<Part> parts = new ArrayList<Part>(); // list of all parts
private Transducer transducer;
private int ConveyorNumber;

public ConveyorAgent(String name, Transducer transducer, int ConveyorNumber)
{
          super();
          this.name = name;
          this.transducer = transducer;
          this.transducer.register(this, TChannel.SENSOR);
          this.ConveyorNumber = ConveyorNumber;
          // this.family = family;
          //this.factorystatus = Status.NOTHINGTODO;
          this.machinestatus = MachineStatus.FREE;
          this.status = ConveyorStatus.OFF;
          this.sensor1 = ConveyorSensorIn.OFF;
          this.sensor2 = ConveyorSensorOut.OFF;
}
```

Messages:

```
//machine received part
public void msgHereIsNewPart(Part part)

  {
            System.out.println("Received Part from previous conveyor");
            this.parts.add(part);
            System.out.println("number of parts on conveyor: " + parts.size());
            stateChanged();
  }

  //machine received part
  public void msgPartReceived(Part p)
  {
            System.out.println("Machine received part from Conveyor");
            parts.remove(p);
            System.out.println("number of parts on conveyor: " + parts.size());
  }

  //when machine finished part, change state of machine
  public void msgImFree()
  {
            System.out.println("Machine is done with part, hes now free");
            this.machinestatus = MachineStatus.FREE;
            stateChanged();
  }

  public void msgMachineFree()
  {
            System.out.println("Just sent part to another conveyor, machine now free");
            this.machinestatus = MachineStatus.FREE;
            stateChanged();
  }

  // left sensor pressed
  public void leftSensorPushed()
   {
            // print("SensorIn Pressed");
            sensor1 = ConveyorSensorIn.ON;
            backCF.msgConveyorPartReceived(family);
            backCF.msgConveyorStopping();
            stateChanged();
  }
```

```
        // right sensor pressed
        public void rightSensorPushed()
        {
                // print("SensorOut Pressed");
                sensor2 = ConveyorSensorOut.ON;
                stateChanged();
        }

        public void leftSensorReleased()
        {
                // print("SensorIn released");
                sensor1 = ConveyorSensorIn.OFF;
                stateChanged();
        }

        public void rightSensorReleased()
        {
                // print("SensorOut released");
                sensor2 = ConveyorSensorOut.OFF;
                stateChanged();
        }
```

Scheduler:

```
        public boolean pickAndExecuteAnAction()
         {

                // when the in sensor is off
                if (sensor1 == ConveyorSensorIn.OFF)
                {
                        TellOtherCFImReady();
                        return true;
                }
                // when the entry sensor is pressed and the out sensor is not and the
                // conveyor is off
                if ((sensor1 == ConveyorSensorIn.ON) && (status == ConveyorStatus.OFF) &&
((sensor2 == ConveyorSensorOut.OFF) || (sensor2 == ConveyorSensorOut.STALLED)))
                {
                        StartConveyor();
                        return true;
                }
                // when the out sensor isvg on and the machine is busy
                if ((sensor2 == ConveyorSensorOut.ON) && (machinestatus ==
        MachineStatus.BUSY))
                {
                        StallConveyor();
                        return true;
```

```
                }
                // when the out sensor is on and the machine is free
                if (((sensor2 == ConveyorSensorOut.ON) && (machinestatus ==
MachineStatus.FREE)))
                {
                        GivePartToMachine();
                        return true;
                }
                return false;
        }
```

Actions:

```
        // conveyor is turned on
        private void StartConveyor() {
                System.out.println("conveyor is on and part can go to this conveyor");

                Object[] args = { ConveyorNumber };
                transducer.fireEvent(TChannel.CONVEYOR, TEvent.CONVEYOR_DO_START, args);
                this.status = ConveyorStatus.ON;
                //stateChanged();

        }

        // stalls conveyor
        private void StallConveyor() {
                System.out.println("Stalling conveyor");
                Object[] args = { ConveyorNumber };
                transducer.fireEvent(TChannel.CONVEYOR, TEvent.CONVEYOR_DO_STOP, args);
                this.machinestatus = MachineStatus.STALLED;
                this.status = ConveyorStatus.OFF;
                //stateChanged();

        }

        // sends msg to previous conveyor to say i'm ready
        private void TellOtherCFImReady()
        {
                System.out.println("notify previous conveyor family that i'm ready");
                //this.factorystatus = Status.READY;
                this.sensor1 = ConveyorSensorIn.CHANGE;
                backCF.msgConveyorReady(backCF);
                //stateChanged();
        }

        // sending part to another conveyor
        private void GivePartToMachine() {
```

```
            System.out.println("conveyor giving part to machine");
            //this.factorystatus = Status.NOTHINGTODO;
            this.machinestatus = MachineStatus.BUSY;
            System.out.println("MACHINE STATUS IS: " + machinestatus);
            this.sensor2 = ConveyorSensorOut.STALLED;
            station.msgHereIsPart(parts.get(0));

            if(status == ConveyorStatus.OFF)
                    this.StartConveyor();
            //stateChanged();
    }
    //for transducer
    public void eventFired(TChannel channel, TEvent event, Object[] args)
    {
            if ((ConveyorNumber * 2 + 1) == (Integer) args[0]
                            || ConveyorNumber * 2 == (Integer) args[0]) {
                    if (channel == TChannel.SENSOR
                                    && event == TEvent.SENSOR_GUI_PRESSED) {
                            if ((Integer) args[0] % 2 == 0) {
                                    this.leftSensorPushed();

                            } else if ((Integer) args[0] % 2 == 1) {
                                    this.rightSensorPushed();
                            }
                    } else if (channel == TChannel.SENSOR
                                    && event == TEvent.SENSOR_GUI_RELEASED) {
                            if ((Integer) args[0] % 2 == 0) {
                                    this.leftSensorReleased();
                                    //backCF.msgConveyorReady(family);

                            } else if ((Integer) args[0] % 2 == 1) {
                                    this.rightSensorReleased();
                            }
                    }

            }
    }
```