**Thermometer**

**CodedQActor** thermometer **context** ctxParkingservice **className** "thermometerSimulator"

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**import** it.unibo.kactor.MsgUtil

**import** kotlinx.coroutines.delay

**import** it.unibo.kactor.ActorBasic

**import** it.unibo.kactor.ApplMessage

**import** alice.tuprolog.Term

**import** alice.tuprolog.Struct

**import** java.util.Timer

**import** kotlin.random.Random

**class** thermometerSimulator (name : String ) : ActorBasic( name ) {

**var** thermo = 5

@kotlinx.coroutines.ObsoleteCoroutinesApi

@kotlinx.coroutines.ExperimentalCoroutinesApi

**override** **suspend** **fun** actorBody(msg: ApplMessage) {

**if**( msg.msgId() == "startThermometer") startThermo()

}

@kotlinx.coroutines.ObsoleteCoroutinesApi

@kotlinx.coroutines.ExperimentalCoroutinesApi

**suspend** **fun** startThermo(){

**while**(**true**){

**for**( i **in** 4..40){

thermo = i

**var** time = Random.nextLong(1000,5000)

println(thermo)

delay(time)

**val** m4 = MsgUtil.buildEvent(name, "thermometer", "temperature($thermo)")

emit(m4)

}

}

}

}

**Client Simulator**

*\*\*Dispatch perché senno non vanno in coda*

**QActor** clientsimulator **context** ctxParkingservice{

[# var n=4 #]

**State** s0 **initial** {

**if**[# n>0 #]{

//emit pickup : pickup(0000)

**forward** outdoor **-m** pickup : pickup(pickup)

[# n = n-1 #]

**updateResource**[#"$n"#]

**delay** 4000

}

}**Goto** s0

}

**Bozza per testare il thermometerSimulator**

**QActor** parkingmanager **context** ctxParkingservice{

[#

lateinit var thermometerActor : ActorBasic

var T = "0"

#]

**State** s0 **initial**{

[# thermometerActor = sysUtil.getActor("thermometer")!! #]

**forward** thermometer **-m** startThermometer : startThermometer(ok)

}

**Transition** t0 **whenEvent** thermometer -> handleTemperature

**State** handleTemperature{

**onMsg**( thermometer : termperature(D) ){

[# T = payloadArg(0) #]

//emit temperature : temperature($T) //for application level

}

**updateResource** [# "$T" #] //after this we can use $T in qak, out of [##]

**println**("temperature=$T")

}**Transition** t0 **whenEvent** thermometer -> handleTemperature

}