Simulator
«Java Class»

+ timer : Timer
+ sensorStatusMap: HashMap<String, Object>
+ district: JSONObject
+ simulationStatus: boolean
+ simDuration: long
+ b: Block
+ block: JSONObject

+ Simulator (sensorStatusMap, district)
+ run ()
+ shutdownTask ()
+ new Block (sensorStatusMap, this)
+ b.createBlock (block): void
+ getSimulationStatus(): return boolean
+ getSensorValue (sensorID): return Object

Block
«Java Class»

+ sensorStatusMap: HashMap<String, Object>
+ simulator: Simulator
+ blockName:String
+ host: String
+ b: Building
+ building: JSONObject

+ Block(sensorStatusMap, simulator)
+ createBlock(block)
+ new Building (sensorStatusMap, simulator)
+ b.createBuilding(blockName, host, building): void

Building
«Java Class»

+ sensorStatusMap: HashMap<String, Object>
+ simulator: Simulator
+ buildingType: String
+ buildingName: String
+ f: Floor
+ floor: JSONObject

+ Building(sensorStatusMap, simulator)
+ createBuildiing (blockName, host, building)
+ new Floor (sensorStatusMap, simulator)
+ f.createBuildingFloor (blockName, buildingName, host, floor): void
+ f.createFloor (blockName, buildingName, host, floor): void

Simulator **1** — Build — **1..*** Block

Block **1** — Contains — **1..*** Building

Floor
«Java Class»

+ sensorStatusMap: HashMap<String, Object>
+ simulator: Simulator
+ floorNumber: long
+ d: Division
+ division: JSONObject
+ s: Space
+ space: JSONObject

+ Floor (sensorStatusMap, simulator)
+ createBuildiingFloor (blockName, buildingName, host, floor)
+ createFloor (blockName, buildingName, host, floor)
+ new Division (sensorStatusMap, simulator)
+ d.createDivision(blockName, buildingName, floorNumber, host, division): void
+ new Space (sensorStatusMap, simulator)
+ s.createSpace(blockName, buildingName, floorNumber, host, space): void

Space
«Java Class»

+ sensorStatusMap: HashMap<String, Object>
+ simulator: Simulator
+ d: Division
+ division: JSONObject

+ Space (sensorStatusMap, simulator)
+ createSpace (blockName, buildingname, floorNumber, host, space)
+ new Division (sensorStatusMap, simulator)
+ d.createDivision (blockName, buildingName, floorNumber, host, division): void

Building **1** — Contains — **1..*** Floor

Space **0..*** — Contains — **1** Floor

Space **1** — Contains

Floor **1** — Contains

Contains

1..*

Division
«Java Class»

+ sensorStatusMap: HashMap
<String, Object>
+ simulator: Simulator
+ sensor: JSONObject
+ sensorType: String
+ sensorModel: String
+ actuator: JSONObject
+ topicString: String

+ Division(sensorStatusMap,
simulator)
+ createDivision (blockName,
buildingName, floorNumber,
host, division)
+ new SensorTempExt (host,
blockName, buildingName,
sensor, sensorStatusMap,
simulator): start()
+ new SensorTempInt (host,
blockName, buildingName,
sensor, sensorStatusMap,
simulator): start()
+ new SensorA (host,
blockName, buildingName,
sensor, sensorStatusMap,
simulator): start()
+ new Consumer (host,
topicString, simulator): start()

1..*

1   Contains   0..*

Consumer
«Java Class»

+ kafkaProperties: Properties
+ topicString: String
+ simulator: Simulator
+ host: String

+ Consumer (host, topicString,
simulator)
+ consumer.subscribe
(topicString)
+simulator.getSimulationStatus()
+ consumer.close()

1

Contains

0..*

Sensor
«Java Interface»

+ setSensorStatus(): void

1   Calls   1..*

Producer
«Java Class»

+ props: Properties
+ producer: KafkaProducer
+ host: String

+ Producer(host)
+ configureTopic (topicString):
void
+ publishMessage (topicString,
message): void
+ producer.send(): void
+ closeProducer (): void
+ producer.close()

SensorTempExt
«Java Class»

+ sensorStatusMap: HashMap
+ simulator: Simulator
+ producer: Producer
+ sensorID: String
+ topicString: String
+ model_parameters: JSONObject
+ message: String

+ SensorTempExt (host, blockName, buildingName, sensor, sensorStatusMap, simulator)
+ new Producer (host)
+ producer.configTopic (topicString): void
+ new ModelA (model_parameters, this, simulator): start()
+ run ()
+simulator.getSimulationStatus(): return boolean
+ producer.publishMessage (topicString, message): void
+ setSensorStatus (double temperature): void

SensorTempInt
«Java Class»

+ sensorStatusMap: HashMap
+ simulator: Simulator
+ producer: Producer
+ sensorID: String
+ topicString: String
+ model_parameters: JSONObject
+ message: String

+ SensorTempInt (host, blockName, buildingName, sensor, sensorStatusMap, simulator)
+ new Producer (host)
+ producer.configTopic (topicString): void
+ new ModelB (model_parameters, this, simulator): start()
+ run ()
+simulator.getSimulationStatus(): return boolean
+ producer.publishMessage (topicString, message): void
+ setSensorStatus (double temperature): void

SensorA
«Java Class»

+ sensorStatusMap: HashMap
+ simulator: Simulator
+ producer: Producer
+ sensorID: String
+ topicString: String
+ message: String

+ SensorA (host, blockName, buildingName, sensor, sensorStatusMap, simulator)
+ new Producer (host)
+ producer.configTopic (topicString): void
+ new ModelRandom (this, simulator): start()
+ run ()
+simulator.getSimulationStatus(): return boolean
+ producer.publishMessage (topicString, message): void
+ setSensorStatus (double temperature): void

SensorModel
«Java Interface»

ModelA
«Java Class»

+ simulator: Simulator
+ tempMin: double
+ tempMax: double
+ localTime: LocalDateTime
+ sensor: Sensor
+ sleep_time: long

+ ModelA (model_parameters, sensor, simulator)
+ run ()
+simulator.getSimulationStatus(): return boolean
+ sensor.setSensorStatus (temperature): void

ModelB
«Java Class»

+ simulator: Simulator
+ tempBase: double
+ maxTempDeviation: double
+ random: Random
+ sensor: Sensor
+ sleepTime: long

+ ModelB (model_parameters, sensor, simulator)
+ new Random ()
+ run ()
+simulator.getSimulationStatus(): return boolean
+ sensor.setSensorStatus (temperature): void

ModelRandom
«Java Class»

+ simulator: Simulator
+ sensor: Sensor
+ random: Random
+ sleepTime: long

+ ModelRandom (sensor, simulator)
+ new Random ()
+ run()
+simulator.getSimulationStatus(): return boolean
+ sensor.setSensorStatus (temperature): void