Høgskolen i Østfold

EXAMINATION

Course code:	**Course:**
ITI43515	Modeling Cyber-Physical Systems
Date:	**Duration:**
3. May 2016	From 9 a.m. to 13 p.m.
Permitted sources:	**Lecturer:**
All written aiding tools	Professor Øystein Haugen

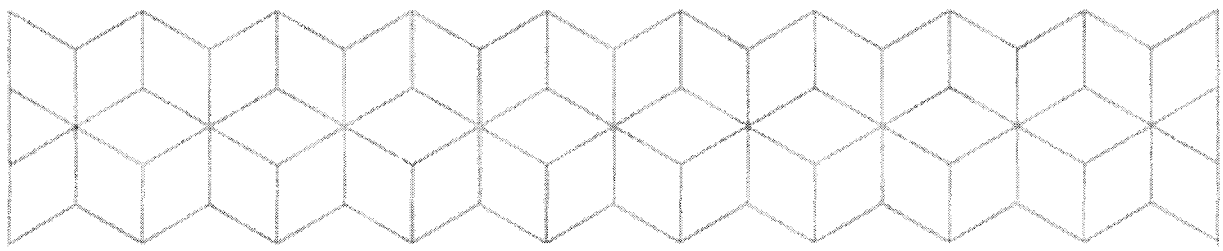
The examination:

The examination papers consist of **8** pages including this page. Please check that the examination papers are complete before you start answering the questions.

Pages 2-4 describe the exam.
Pages 5-8 are Annexes giving relevant details from the course.

Date of announcement of the examination results: 30. May 2016

The examination results are available on the Studentweb no later than two workdays after the announcement of the examination results www.hiof.no/studentweb



Exam Modeling Cyber-Physical Systems Spring 2016

Context

The context is The Room – our experimental context consisting of a controlling computer equipped with a Tellstick Duo which communicates wirelessly on 433.92 MHz to a set of sensors and actuators. The sensors are mostly thermometers and the actuators are mainly power switches that can switch on or off.

The Room V5 has two inside thermometers and one on-off actuator connected to a heater. The following assumptions holds for The Room V5:

- The room will get warmer when the heater is on, and eventually go above comfort temperature
- The room will get colder when the heater is off, and eventually go below comfort temperature

The assumptions are not always realistic, but for our purpose they make the temperature control a little easier.

Our starting point is The Room V4 where the control of the temperature is done through a couple of rule state machines, one controlling the upper boundary of the comfort temperature range and the other controlling the lower boundary of the comfort temperature range.

Exercise 1 Model modifications (50%)

a. Composite structure (UML) or configuration (ThingML)

Figure 1 contains a composite structure of the desired system. The Rules do not control the heater directly, but send their recommendation to the controller CTRL. The controller decides whether the heater should be on or off depending on the recommendations. Your task is to modify the ThingML configuration to match the composite structure.

What changes are needed to the ThingML configuration of The Room V4? (given in Annex 1 The Room V4 configuration in ThingML)

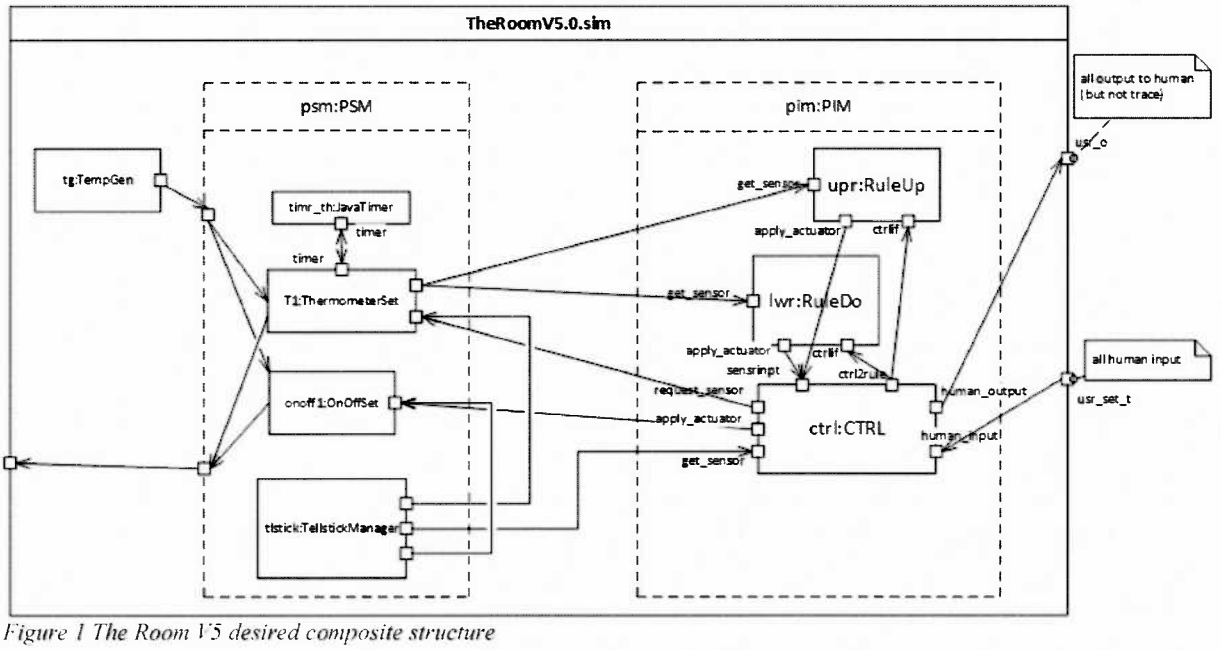


Figure 1 The Room V5 desired composite structure

b. The Room V5 behavior

Modify the system to only send a new heater actuation if it is different from the last actuation.

The desired behavior of The Room V5 is now given in Figure 2.

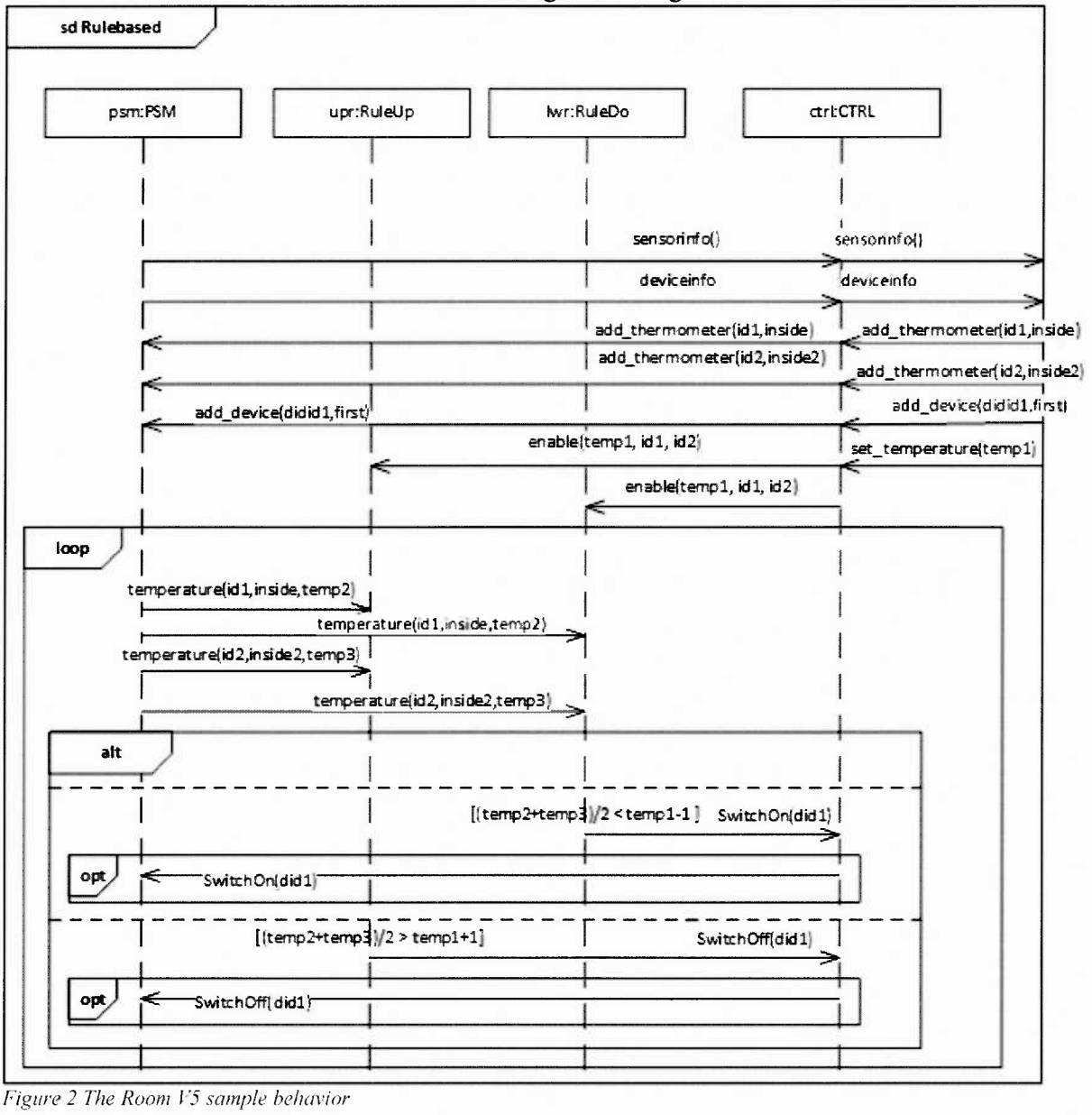


Figure 2 The Room V5 sample behavior

For the *CTRL* state machine, you shall do the following:
Make the state Thermostat into a composite state and introducing three simple internal states in it: "ExpectIncrease" and "ExpectDecrease" and "Initial". The increase and decrease relate to the expected changes of temperature in the room.

The Thermostat state of The Room V4 is given in Annex 2 CTRL state Thermostat from The Room V4.

You should write down the state Thermostat for The Room V5 in ThingML, and explain in natural language any additional changes that you found desirable to do in CTRL.
Explain in natural language any changes you would need in V5 relative to V4 for RuleUp and RuleDo (given in Annex 3 RuleUp from The Room V4 (RuleDo is symmetrical)).

Exercise 2 Risk analysis of The Room as a Freezer (25%)

We assume that our Room V5 is used to control a Freezer used for storing food. Thus the heater is replaced by a cooling device and the actuating is changed accordingly.

Perform a risk analysis of the Freezer on behalf its owner focusing on the following ingredients

- The assets of the Freezer
- Non-malicious incidents (do not consider malicious threats)
- Vulnerabilities
- Treatments

Exercise 3 Variability modeling of The Room (25%)

a. VSpec model

Make a *BVR model* that describes a Room product line with the following properties:

- Always has a controlling device with a Tellstick
- One or more sensors
    - Each sensor is either a thermometer or a hygrometer or both
- One or more actuators
    - Each actuator is either an on-off switch or a dimmer

b. Resolution model

From this BVR model define *one resolution that describes the V5 Room* in Exercise 1.

c. Variants

With a limitation of 3 sensors and 2 actuators, *how many possible rooms* does the BVR model define? Explain how you find this number.

---------- End of Exam ----------

Annex 1 The Room V4 configuration in ThingML

```
import "psm_sim.thingml"
import "pim.thingml"
import "io.thingml"
import "javatimer.thingml"

configuration CPS {
	instance tlstick:TellstickManager
	instance T1:ThermometerSet
	instance onoff1:OnOffSet
	instance ctrl:CTRL
	instance upr:RuleUp
	instance lwr:RuleDo
	instance myself:Human
	instance timer : TimerJava
	// SIMULATION
	instance tg:TempGen

	// PIM
	connector ctrl.request_sensor => T1.require_val
	connector ctrl.ctrl2rule => lwr.ctrlif
	connector ctrl.ctrl2rule => upr.ctrlif
	connector ctrl.apply_actuator => onoff1.require_val

	connector upr.apply_actuator => onoff1.require_val
	connector lwr.apply_actuator => onoff1.require_val


	// PSM
	connector tlstick.to_T1 => T1.require_val
	connector tlstick.to_pim => ctrl.get_sensor
	connector tlstick.to_onoff1 => onoff1.require_val

	connector T1.provide_val => lwr.get_sensor
	connector T1.provide_val => upr.get_sensor
	connector T1.timer => timer.timer

	// Human
	connector myself.send_cmd => ctrl.human_input
	connector ctrl.human_output => myself.get_values

	// SIMULATION
	connector tg.give_values => T1.simulate_val
}
```

Annex 2 CTRL state Thermostat from The Room V4

```
state Thermostat {
      on entry ctrl2rule!enable(tmrature, thermo_id1,thermo_id2, switch_id)
      on exit ctrl2rule!disable()   // freeze the rules

      // transitions of state Thermostat
      transition -> On
      event swon:human_input?SwitchOn
      action do
            apply_actuator!SwitchOn(swon.did)
      end
      transition -> Off
      event swoff:human_input?SwitchOff
      action do
            apply_actuator!SwitchOff(swoff.did)
      end
      transition -> Thermostat
      event set_temp:human_input?set_temperature
      action do
            tmrature = set_temp.t
            ctrl2rule!enable(tmrature, thermo_id1, thermo_id2, switch_id)
      end
      transition -> Thermostat
      event get_sensor?sensorinfo
      event get_sensor?deviceinfo
      event get_sensor?temperature
      event human_input?add_thermometer
      event human_input?add_device
      action do
            print("IERROR: Unexpected signal in Thermostat. Discarded\n")
      end
}
```

Annex 3 RuleUp from The Room V4 (RuleDo is symmetrical)

```
thing RuleUp includes GeneralMsg, TemperatureMsg, OnOffMsg
@debug "false"
{
	provided port get_sensor {
		receives temperature
	}
	required port apply_actuator{
		sends SwitchOn, SwitchOff
	}
	provided port ctrlif {
		receives enable, disable
	}

	// Properties
	property sid1:Integer = 0	// identifier of the first thermometer
	property trm1:Double = -101	// temperature of first thermometer
	property sid2:Integer = 0	// identifier of the second thermometer
	property trm2:Double = -102	// temperature of second thermometer
	property switch_id:Integer = 0	// The actuator switch identifier
	property desiredtemp:Double = 20	// the desired temperature

	// Behavior definition
	statechart Rule_behavior init Disabled {
		state Disabled {
			transition -> Enabled
			event enabling:ctrlif?enable
			action do
				// Any initialization for the rule can be done here
				sid1 = enabling.sid1
				sid2 = enabling.sid2
				switch_id = enabling.swid
				desiredtemp = enabling.desiredtemp
			end

			transition -> Disabled
			event disabling:ctrlif?disable
			event tmps:get_sensor?temperature
			action do
				// just discard, do nothting
			end
		}

		state Enabled {
			// Transition when having received all values
			transition -> Enabled
			event receivesensor: get_sensor?temperature
			action do
				// Record sensor values
				if(sid1==receivesensor.id) trm1=receivesensor.t
				if(sid2==receivesensor.id) trm2=receivesensor.t


				// and then eventual actuation happens
				// if some condition then actuate
				// This rule checks the upper boundary of comfort
				// range against the average of temperatures
				if (((trm1+trm2)/2) > desiredtemp+1)
				 apply_actuator!SwitchOff(switch_id)
			end
			transition -> Disabled
```

```
                event disabling:ctrlif?disable
                action do
                        // any clean up can be done here
                        trm1 = -201
                        trm2 = -202
                end
                transition -> Enabled
                event reenabling:ctrlif?enable
                action do
                        // reinitialize
                        sid1 = reenabling.sid1
                        sid2 = reenabling.sid2
                        switch_id = reenabling.swid
                        desiredtemp = reenabling.desiredtemp
                end
        }
    }
}
```