HEADS
Heterogenous and Distributed
Services for the Future Computing Continuum

The Room X4 – Handling risks by concurrent watchdogs

HEADS
Heterogenous and Distributed
Services for the Future Computing Continuum

Recap of X3

The Room X3 – implications of a challenging environment

- X3 guards the expected inputs with timers
- X3 guards the expected results of output with clever observation and recovery
- X3 has modifications that are due to the challenging environment
  - but to test X3 it is a lot easier to execute the simulated version!
- X3 in reality would have to
  - manipulate thermometers e.g. by removing batteries
  - manipulate switches e.g. by physically altering them

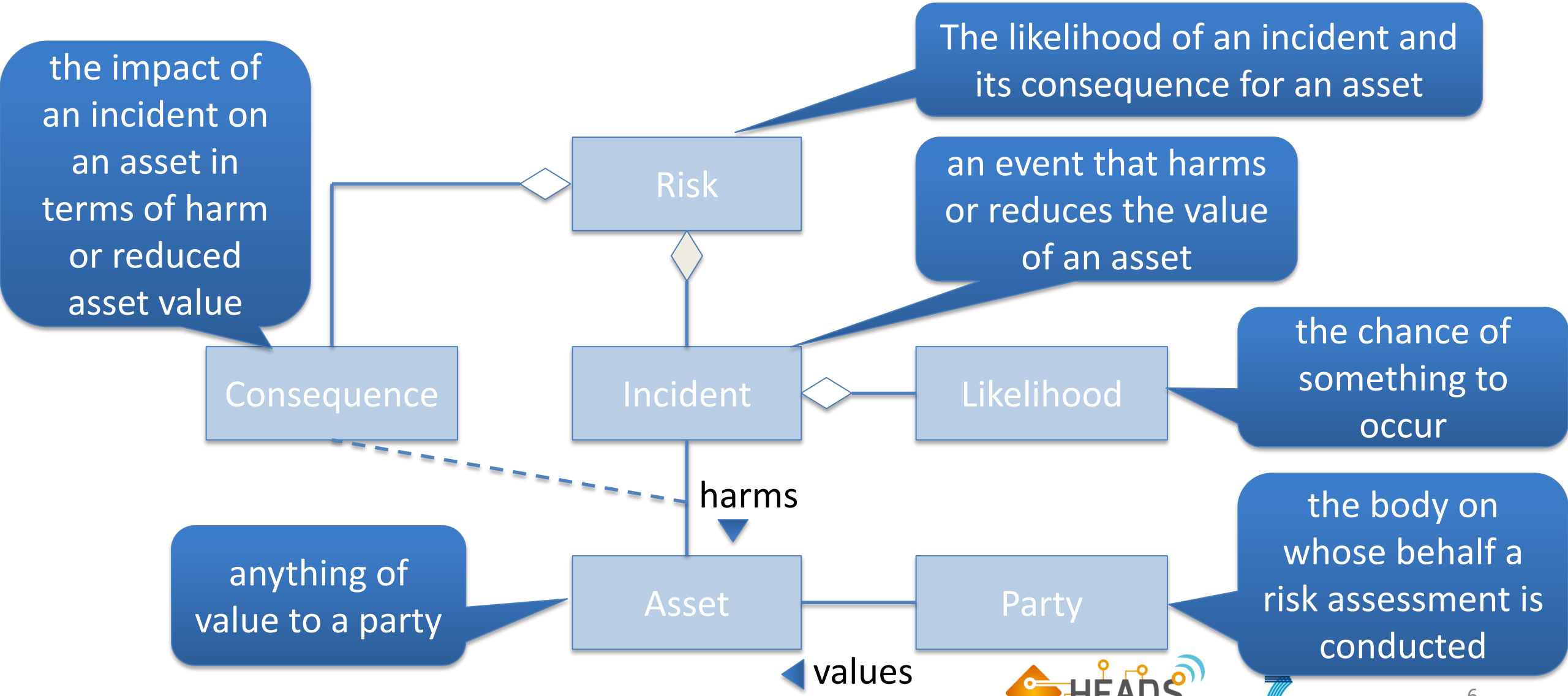
HEADS
Heterogenous and Distributed
Services for the Future Computing Continuum

Small Risk Analysis

Risk Analysis

- A Risk Analysis can be performed in other to establish what risks should be protected against
- We present a few slides indicating what could happen to a Room – the analysis presented is not complete
- The risk analysis ends up with some suggestions for treatments

What is Risk?

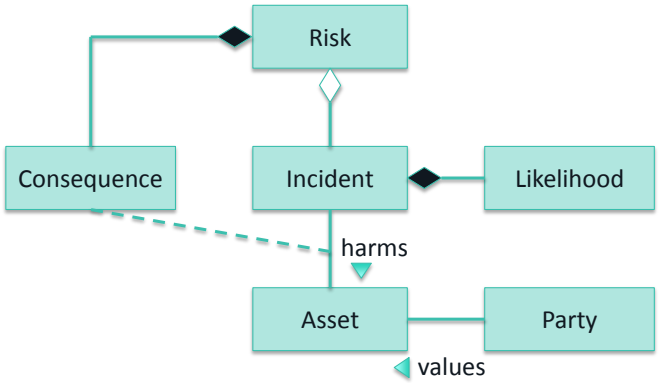

the impact of an incident on an asset in terms of harm or reduced asset value
The likelihood of an incident and its consequence for an asset
Risk
an event that harms or reduces the value of an asset
Consequence
Incident
Likelihood
the chance of something to occur
harms
anything of value to a party
Asset
Party
the body on whose behalf a risk assessment is conducted
values

Assets – what we want to protect and maintain

Asset	Explanation
Comfort temperature	That the temperature in the room is within the range intended by its user
The Room is not physically jeopardized	There should be no fire due to overheating, or freeze due to no heating

Consequence scale for The Room physically jeopardized

Consequence value	Definition
Insignificant	Unpleasant temperature for less than 2 hours
Minor	Unpleasant temperature for 2-25 hours
Moderate	Unhealthy temperature for more than 2 hours
Major	Freezing pipes
Critical	Heater constantly on and burning



Assets and Incidents and Vulnerability

Affecting (Asset)	Event (Incident)	Exploiting What? (Vulnerability)
The Room is not physically jeopardized	Heater switch fails to turn on	Unreliable cheap switches
The Room is not physically jeopardized	Heater switch fails to turn off	Unreliable cheap switches
Comfort temperature	Thermometers show wrong temperatures	Unreliable cheap thermometers

Likelihood and Consequence of Incidents from Non-Malicious

Threat	Incident	Likelihood	Consequence
The Room will freeze	Heater switch fails to turn on	Unlikely	Major
The Room will burn	Heater switch fails to turn off	Unlikely	Critical
Inadequate temperature	Thermometers show wrong temperatures	Certain	Insignificant

Risk Levels (threats indicated inside cells)

Consequence	Likelihood: Rare	Unlikely	Possible	Likely	Certain
Critical	Turns heater on	fails to turn off			
Major		Turns heater off, fails to turn on			
Moderate				Uses web i/f	
Minor					
Insignificant					show wrong

Treatment of Incidents from Non-Malicious Threats

Incident	Risk level	Treatment
Heater switch fails to turn on	Medium	Introduce temperature warning
Heater switch fails to turn off	High	Introduce temperature warning; Physical emergency switch on heater
Thermometers show wrong temperatures	Medium	Add more redundant thermometers

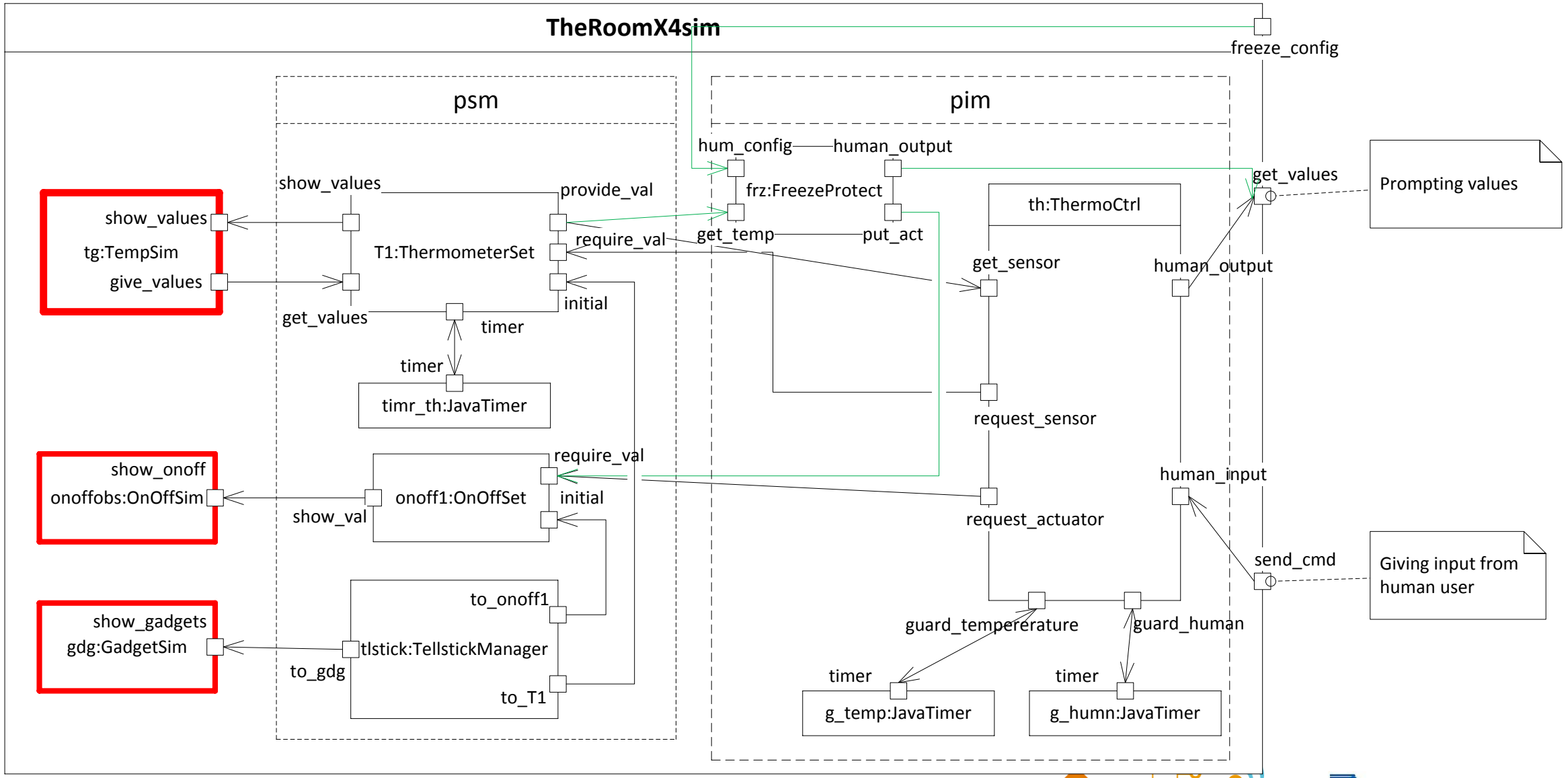
HEADS
Heterogenous and Distributed
Services for the Future Computing Continuum

The Room X4 – monitoring low temperatures

Watchdog on freezing temperatures

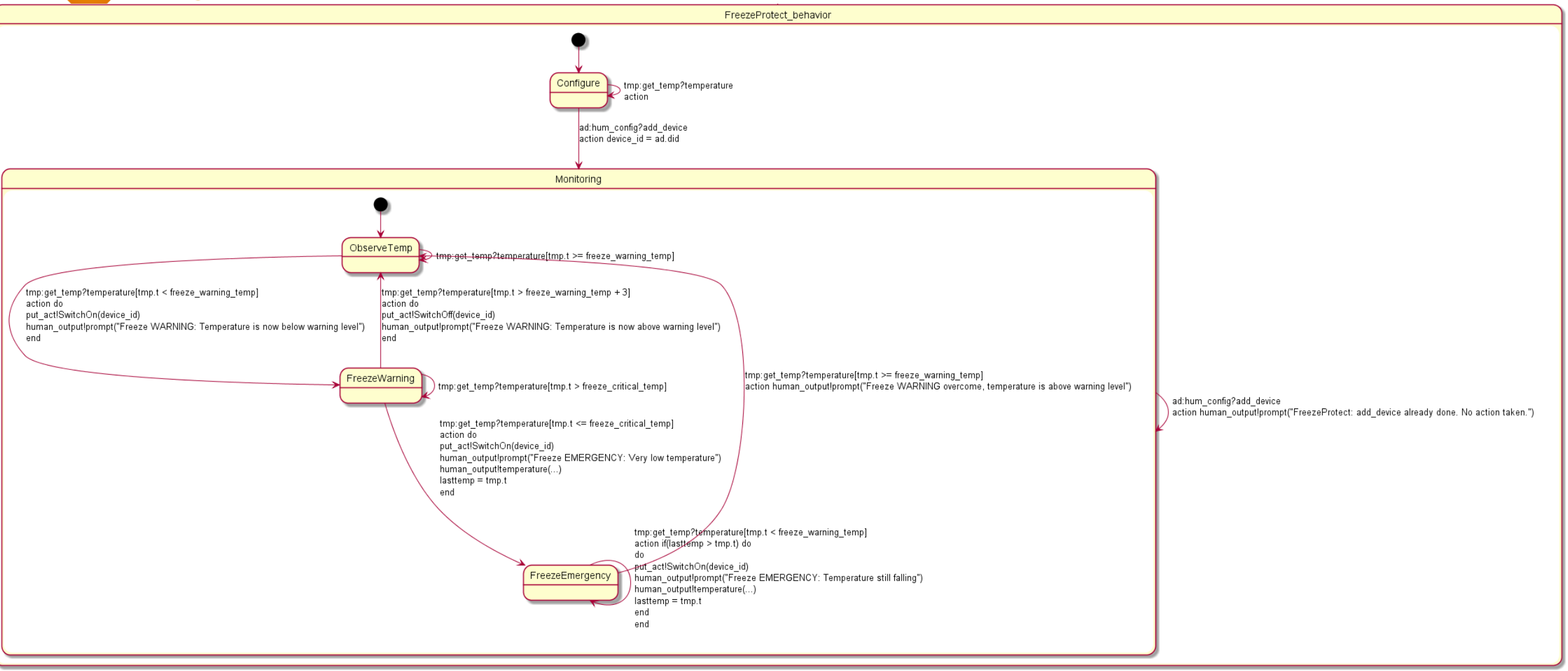
- Our logical watchdog is a process that runs in parallel with our main service
- It monitors the well-being of the Room
  - whether the Room's temperature is falling towards freezing
- When the temperature is alarming,
  - warnings are issued to the Room's owner (or user)
    - in the simulated version, this is merely a prompt onto the human output, but in a future real version, it could be an SMS to the user's cellphone
  - repeated message is sent to Switch to turn on

The Room X4 architecture



Technical issue

- Connectors – the colored connectors are added
  - Multiple connectors from the same sending port
    - An output message is cloned and sent on all the connectors
  - Multiple connectors into the same receiving port
    - This is no problem as the messages will be sorted on arrival time


FreezeProtect_behavior
Configure
tmp:get_temp?temperature
action
ad:hum_config?add_device
action device_id = ad.did
Monitoring
ObserveTemp
tmp:get_temp?temperature[tmp.t >= freeze_warning_temp]
tmp:get_temp?temperature[tmp.t < freeze_warning_temp]
action do
put_act!SwitchOn(device_id)
human_output!prompt("Freeze WARNING: Temperature is now below warning level")
end
tmp:get_temp?temperature[tmp.t > freeze_warning_temp + 3]
action do
put_act!SwitchOff(device_id)
human_output!prompt("Freeze WARNING: Temperature is now above warning level")
end
FreezeWarning
tmp:get_temp?temperature[tmp.t > freeze_critical_temp]
tmp:get_temp?temperature[tmp.t >= freeze_warning_temp]
action human_output!prompt("Freeze WARNING overcome, temperature is above warning level")
ad:hum_config?add_device
action human_output!prompt("FreezeProtect: add_device already done. No action taken.")
tmp:get_temp?temperature[tmp.t <= freeze_critical_temp]
action do
put_act!SwitchOn(device_id)
human_output!prompt("Freeze EMERGENCY: Very low temperature")
human_output!temperature(...)
lasttemp = tmp.t
end
FreezeEmergency
tmp:get_temp?temperature[tmp.t < freeze_warning_temp]
action if(lasttemp > tmp.t) do
do
put_act!SwitchOn(device_id)
human_output!prompt("Freeze EMERGENCY: Temperature still falling")
human_output!temperature(...)
lasttemp = tmp.t
end
end

FreezeProtect – notice that guards are tested in order

Tested first

```
state FreezeWarning {
     transition -> ObserveTemp
     event tmp:get_temp?temperature
     guard tmp.t>freeze_warning_temp+3 // We give a little delta here with 3
     action do
          put_act!SwitchOff(device_id)
          human_output!prompt("Freeze WARNING: Temperature is now above warning level")
     end
     transition -> FreezeWarning
     event tmp:get_temp?temperature
     guard tmp.t>freeze_critical_temp
          // Just discard, we are still in Warning range
     transition -> FreezeEmergency
     event tmp:get_temp?temperature
     guard tmp.t<=freeze_critical_temp
     action do
          put_act!SwitchOn(device_id)
          human_output!prompt("Freeze EMERGENCY: Very low temperature")
          human_output!temperature(tmp.id, tmp.txt, tmp.t)
          lasttemp = tmp.t
     end
}
```

On separation of concerns

- The idea is that the Watchdog (FreezeProtection) should be independent from the behavior of the main service, the Thermostat
- Clearly, independence is not total as:
  - Both FreezeProtection and ThermoCtrl will actuate the switch
    - and therefore they both need to configure the switch
      - In X4 we do this independently through two human interfaces
        - » This ensures independence, but introduces the risk of human error on trivial matters such as the id of the switch

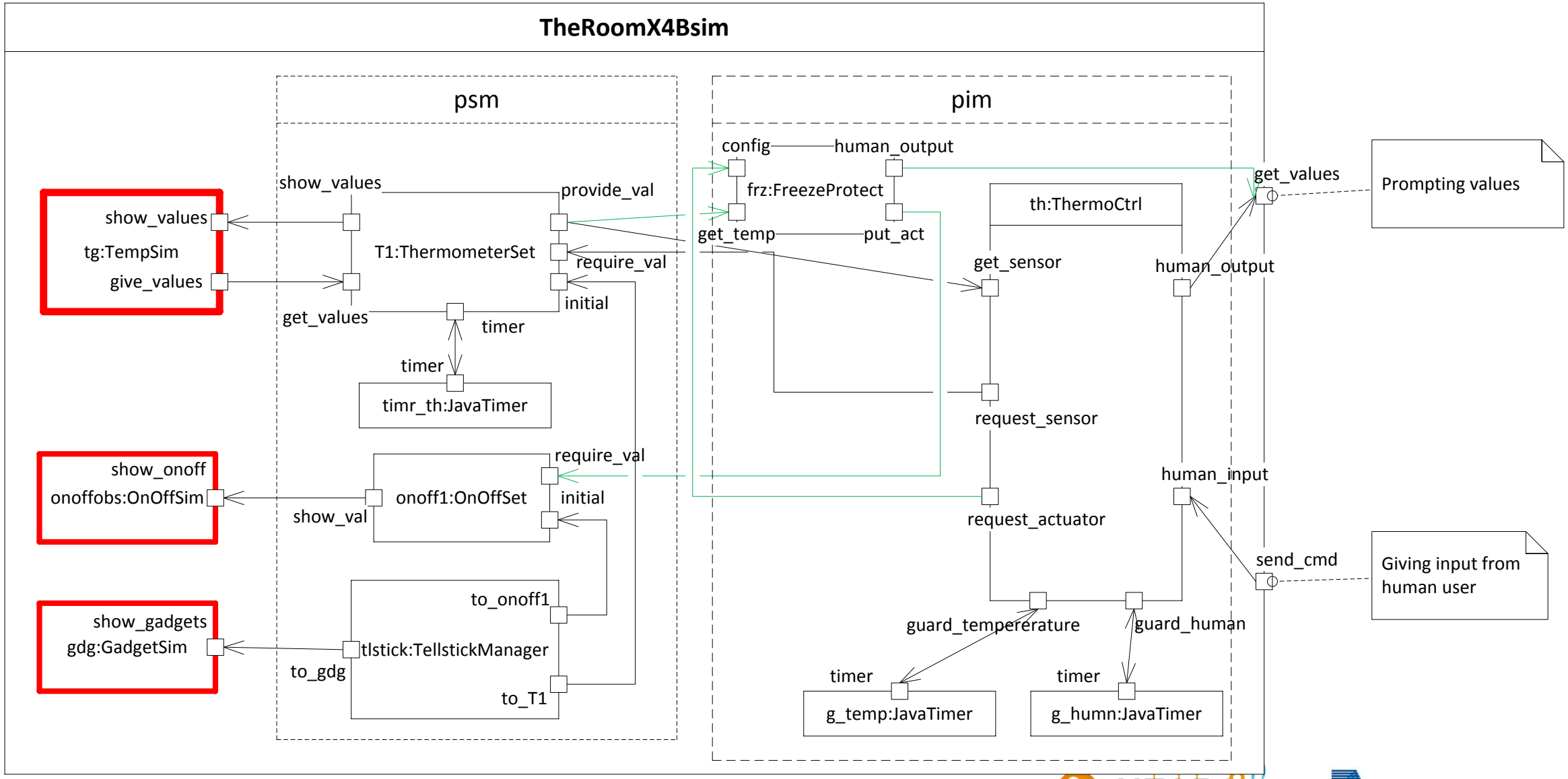
HEADS
Heterogenous and Distributed
Services for the Future Computing Continuum

The Room X4B – intercepting messages

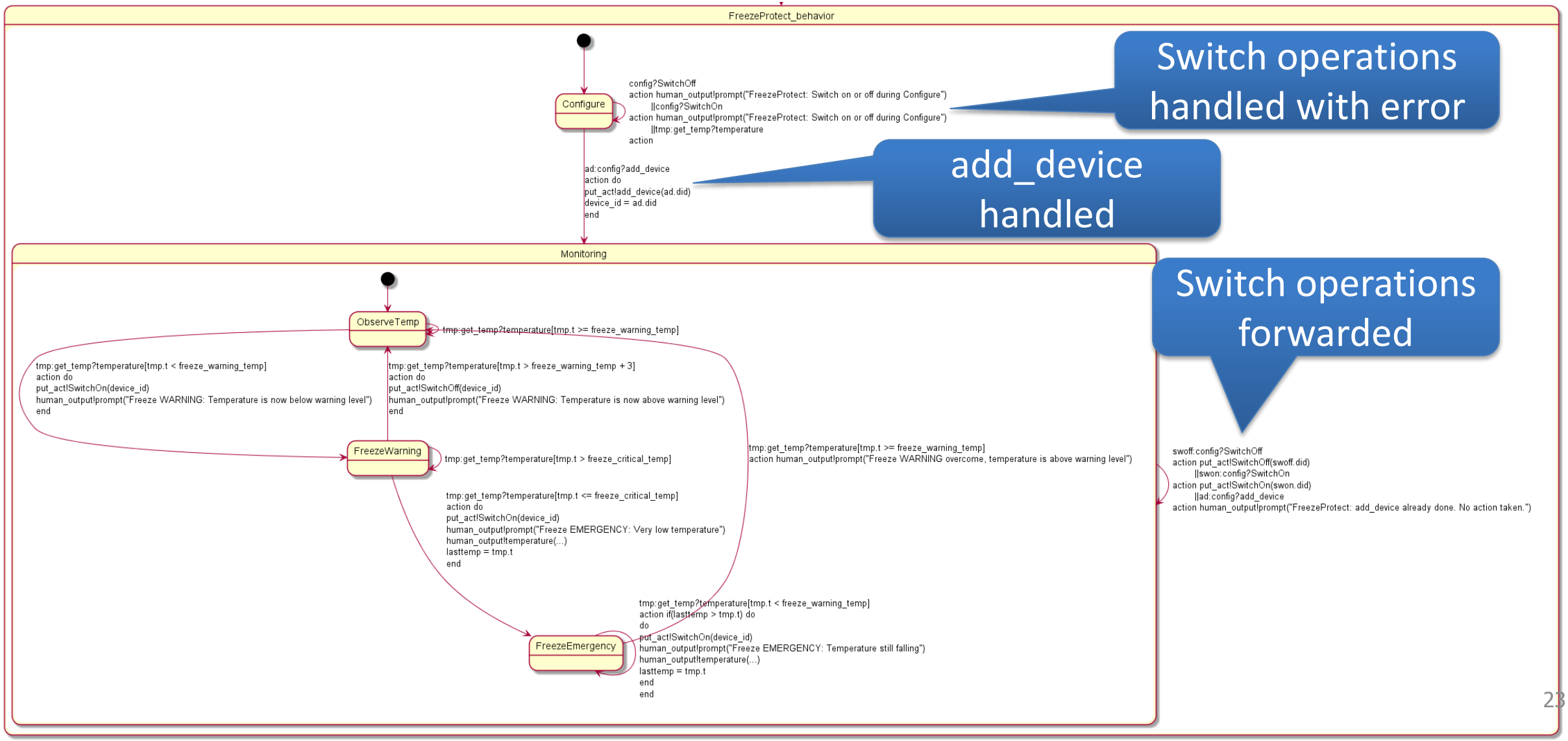
Intercepting Watchdog

- The problem with X4A was that the FreezeProtect was so independent from the main thermostat service that human error could occur
- We could change the architecture slightly and achieve better integration (and thus slightly less independence) eliminating the possibility of human error
  - Notice that ThermoCtrl no longer directly actuate switch; all communication from ThermoCtrl to onoffset goes through FreezeProtect

The Room X4B – more integrated



FreezeProtect now handles SwitchOn and SwitchOff



FreezeProtect details in ThingML (1)

```
statechart FreezeProtect_behavior init Configure {
    state Configure {
    transition -> Monitoring
    event ad:config?add_device
    action do
        put_act!add_device(ad.did)
        device_id = ad.did
    end
    transition -> Configure
    event tmp:get_temp?temperature
    action do end // Do nothing, just discard
    transition -> Configure
    event config?SwitchOn
    event config?SwitchOff
    action
        human_output!prompt("FreezeProtect: Switch on or off during Configure")
    }
```

add_device handled

Switch operations handled with error

FreezeProtect details in ThingML (2)

```
composite state Monitoring init ObserveTemp keeps history {
    state ObserveTemp ...
    state FreezeWarning ...
    state FreezeEmergency ...

    transition -> Monitoring
    event ad:config?add_device
    action
        human_output!prompt("FreezeProtect: add_device already done. No action taken.")
    transition -> Monitoring
    event swon:config?SwitchOn
    action do
        put_act!SwitchOn(swon.did)
    end
    transition -> Monitoring
    event swoff:config?SwitchOff
    action do
        put_act!SwitchOff(swoff.did)
    end
}
```

Switch operations forwarded

The Room X4B – intercepting Watchdog summary

- FreezeProtect works in parallel with the Thermostat
  - There may be rooms where protection against freezing is not an issue, and the watchdog can be omitted easily
  - It would be possible to include this monitoring within the main service, as we already has shown by monitoring unexpected temperature movement, but separate concurrent watchdogs are
    - better separation of concern
    - may even be deployed as truly separate devices
- Interception of messages may be useful for integration

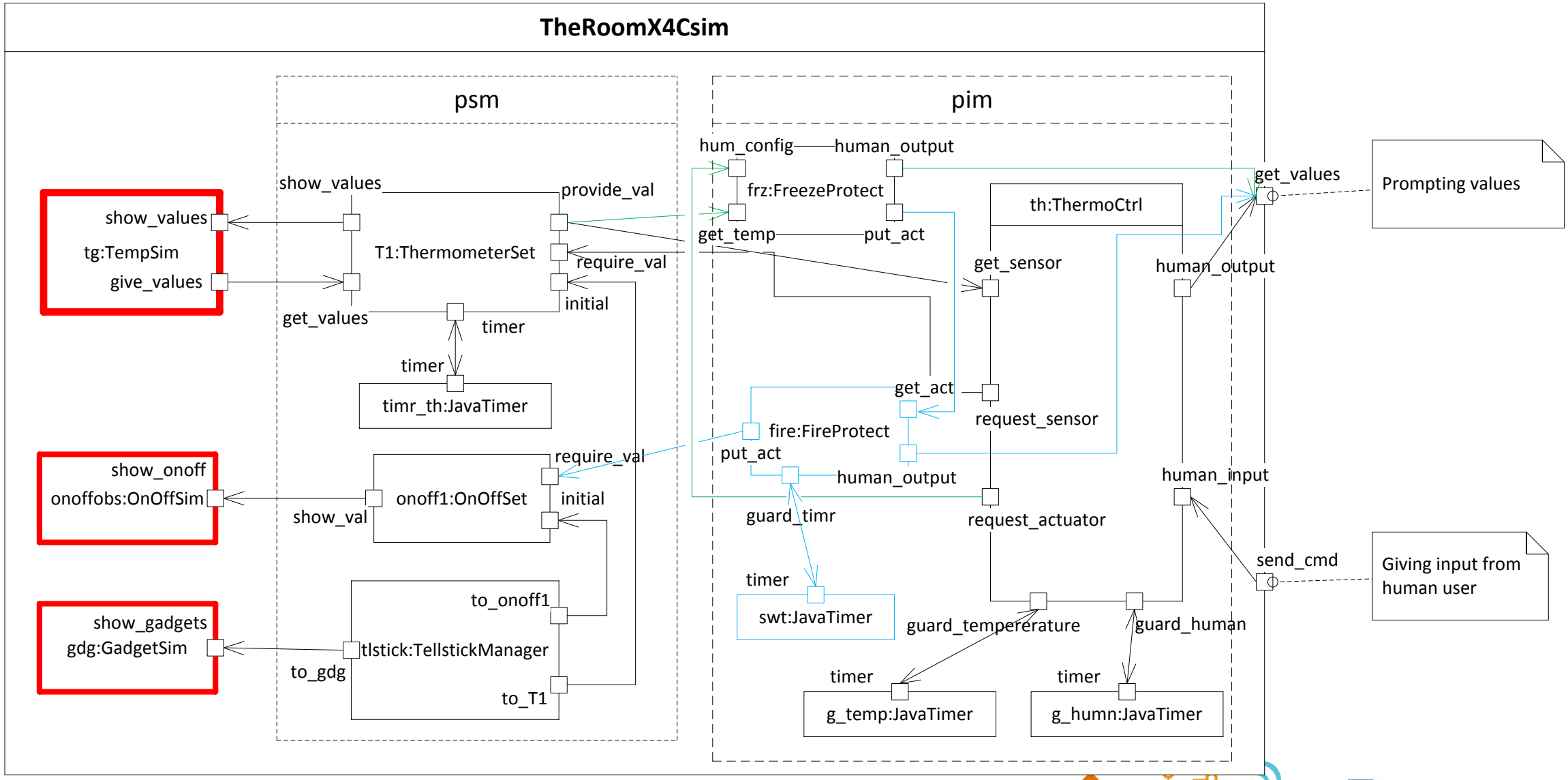
HEADS
Heterogenous and Distributed
Services for the Future Computing Continuum

The Room X4C – monitoring heater activity

More risk treatment: Fire protection

- Introducing FireProtect to prevent the switch to be on for so long that the chance of fire is too high
- We let FireProtect be inserted in the chain between FreezeProtect and the switch controller in the PSM
- FireProtect makes sure that the heater will rest every once in a while

The Room X4C – Fire protection



Technical issues

- Connectors – the colored connectors are added
  - FireProtection intercepts any output to the Switch
    - now coming from the FreezeProtect watchdog
- Timer duration – a PSM problem with PIM consequences
  - It turns out that when we want to handle long durations, our X3 timer is not sufficient as it will need wider range
    - We change from 2 byte integers to 4 byte integers

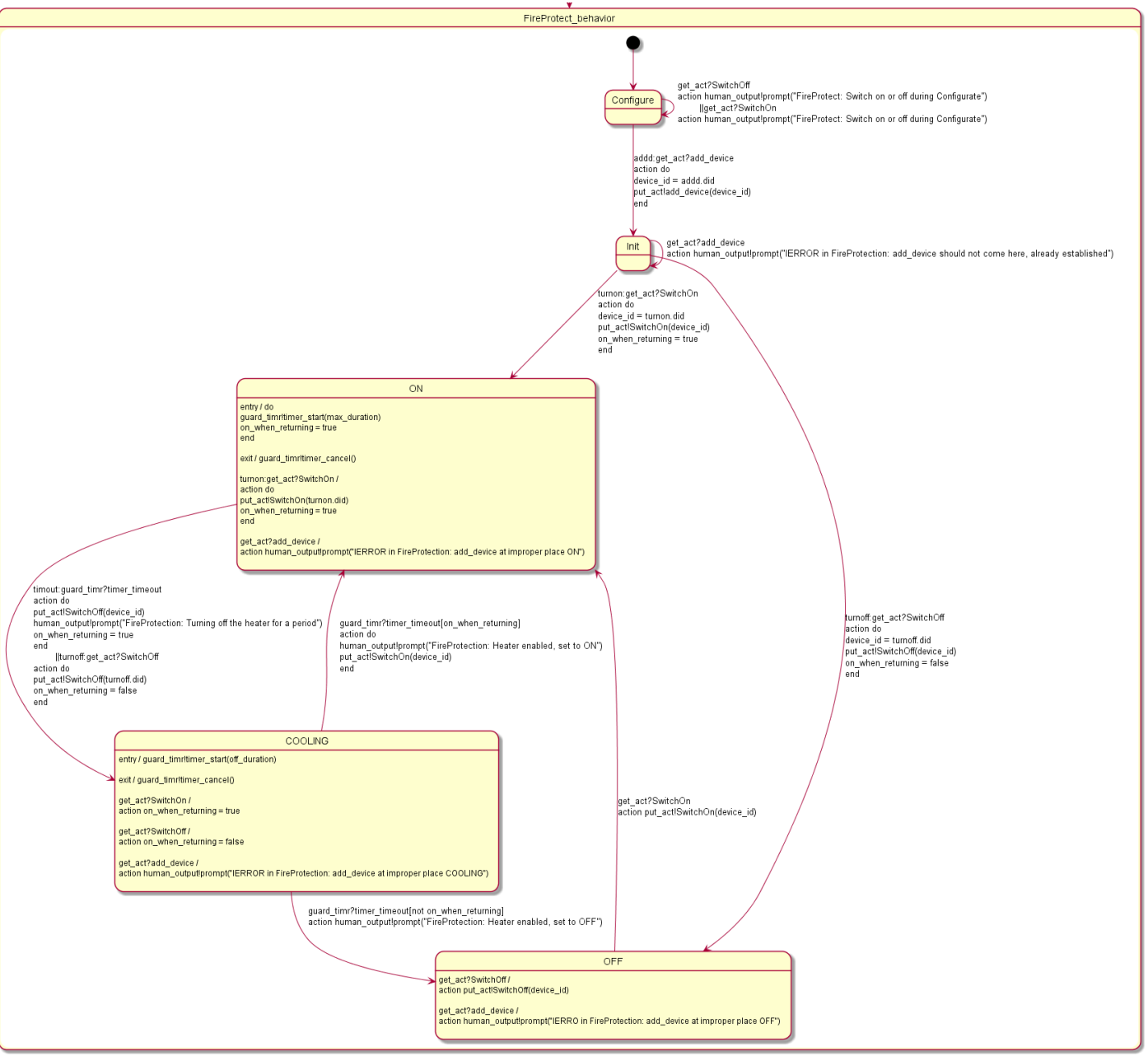
The improved timer


```
thing fragment TimerMsgs {
// Start the Timer
message timer_start(delay : Long);
// Cancel the Timer
message timer_cancel()@debug "false";
// Notification that the timer has expired
message timer_timeout();
}
```

Intricate problems with the fire protection

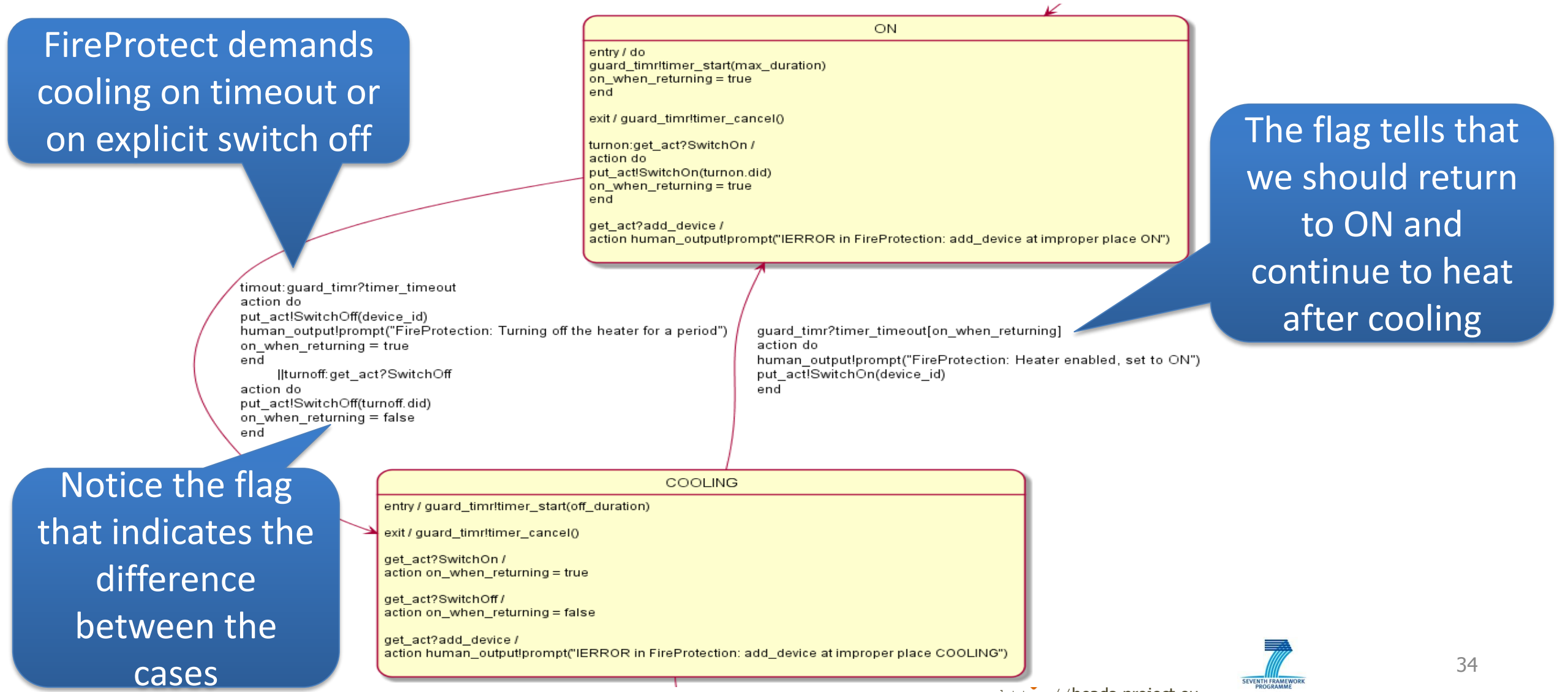
- Switching on or off can come from the main thermostat or from the freeze protection at any time
- We must force the cooling to last at least *off_duration*
- It should not be possible to trick the cooling by pretending to turn off, and then immediately turn on again
  - We decide that whenever the heater is turned off, regardless of whether it is due to having been on for *max_duration*, it shall cool off for at least *off_duration*

FireProtect – overview



- ON
  - the heater is on and we are waiting for a break
    - either by the timeout
    - or by an explicit turn off
- COOLING
  - the heater is turned off
  - waiting a given time regardless
- OFF
  - The heater is off and there should be no danger of fire

FireProtect – from ON to COOLING – and back again



Internal transitions – all inside one state

```
state ON { // Invariant: The heater is ON but should not be ON longer than max_duration
on entry do
      guard_timr!timer_start(max_duration)
      on_when_returning = true
end
on exit guard_timr!timer_cancel()

      internal event turnon:get_act?SwitchOn
      action do
            put_act!SwitchOn(turnon.did)
            on_when_returning = true
      end
      transition -> COOLING
      event turnoff:get_act?SwitchOff
      action do
            put_act!SwitchOff(turnoff.did)
            on_when_returning = false
      end
      transition -> COOLING
      event timout:guard_timr?timer_timeout
      action do
            // turn the heater off for fire protection
            put_act!SwitchOff(device_id)
            human_output!prompt("FireProtection: Turning off the heater for a period")
            on_when_returning = true
      end
      internal event get_act?add_device
      action human_output!prompt("IERROR in FireProtection: add_device at improper place ON")
}
```

The timer is controlled on entry and exit

This is an internal transition that does not go outside the state ON and therefore the timer is not reset (which would make it possible to prolong the heating indefinitely)

Notice the flag that indicates where to return after the cooling

Summary of X4C with FireProtect

- It is intricate to make sure that we ensure cooling
- Internal transitions
  - stay in the state and do not execute entry and exit clauses
  - useful for us in ON and COOLING since we do not want resetting of the timers
- Flag used to indicate future state
  - Flags should not be used instead of states
  - but here we use a flag to hold the value of a future state
    - The flag is then checked in the transition guard

Summary of The Room X4 mitigating risks

- A small risk analysis showed that there may be risks of
  - freezing if the heater never goes on
  - fire if the heater is always on
- We introduce independent, concurrent watchdogs on the actuation of the heater (separation of concerns)
  - FreezeProtect takes input from Thermostat, and delivers switching commands to FireProtect, and error messages to user
  - FireProtect takes input from FreezeProtect, and delivers switching commands to actuator, and error messages to user

Consortium

http://heads-project.eu