# CREST

## A Continuous, REactive SysTems DSL

Stefan Klikovits, Université de Genéve

@stklik

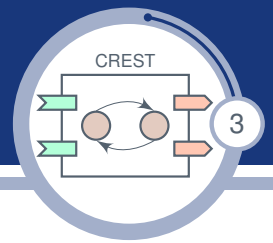
2018 CUSO Winter School

# Growing plants

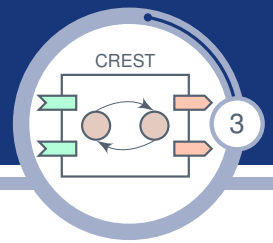
# Growing plants

(c) Debra Roby
https://www.flickr.com/photos/darinhercules/

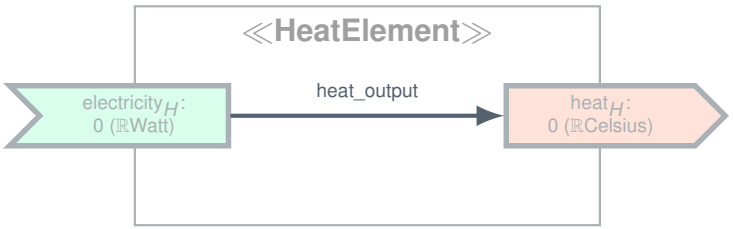
# The Language



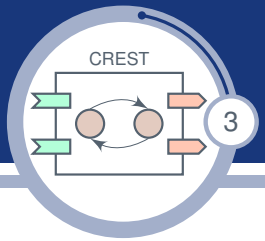

# The Language

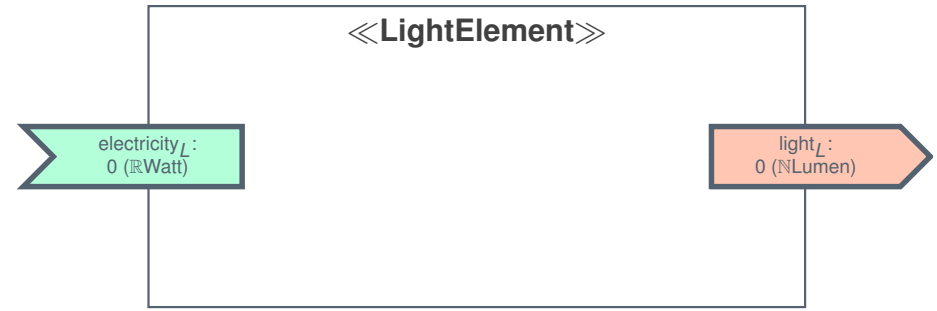


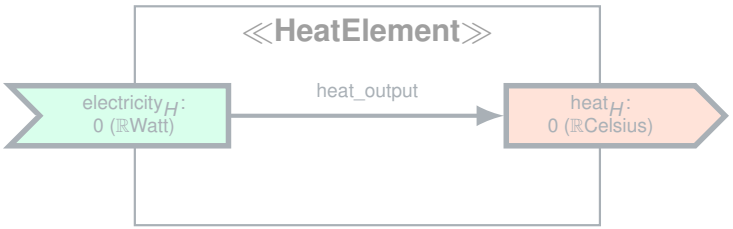

# The Language



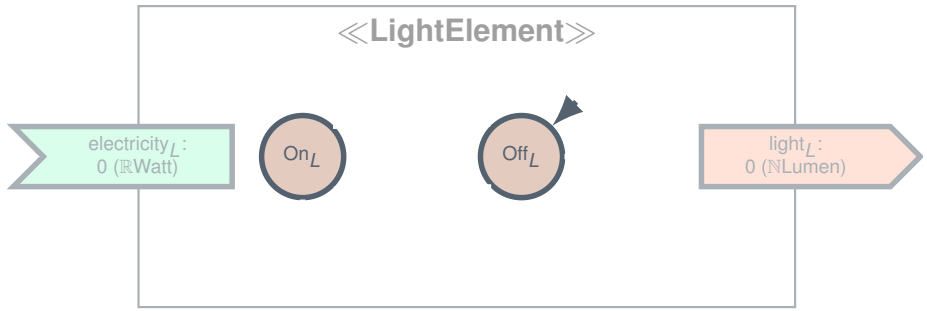
# The Language

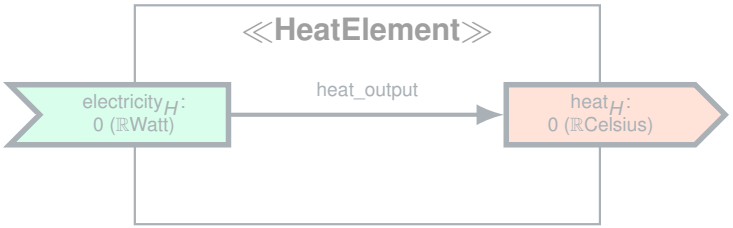


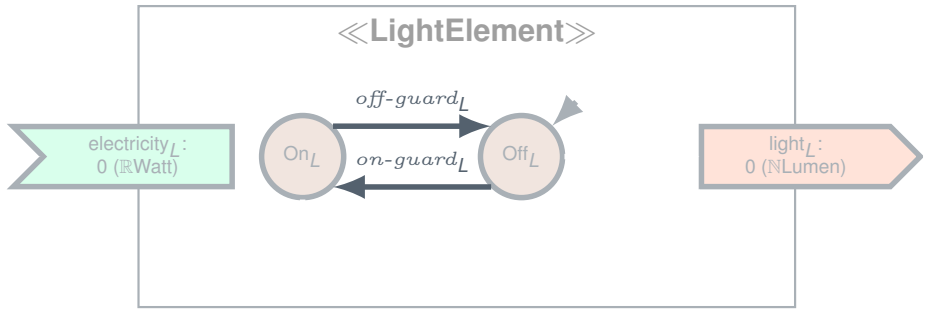



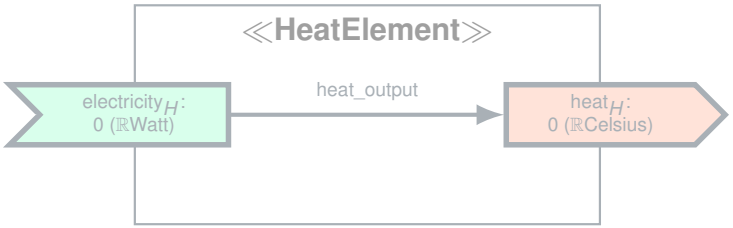
# The Language



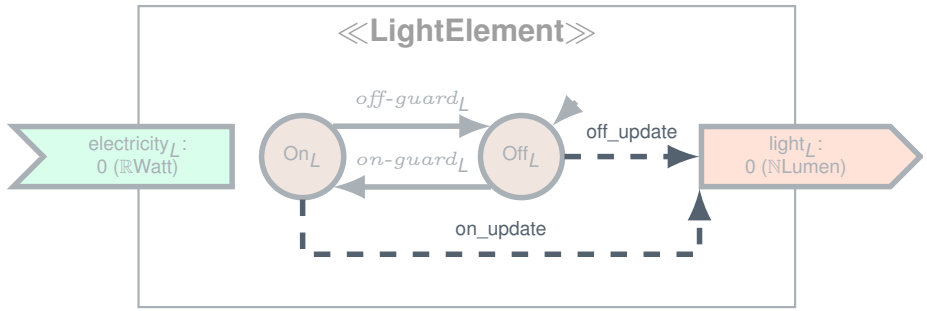



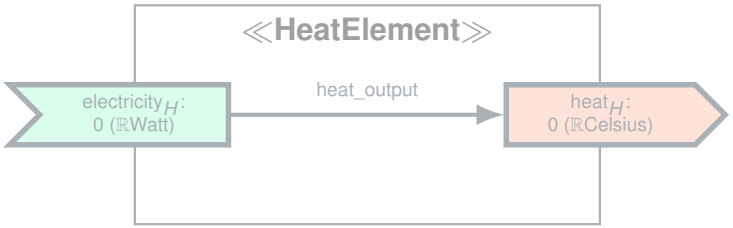
# The Language



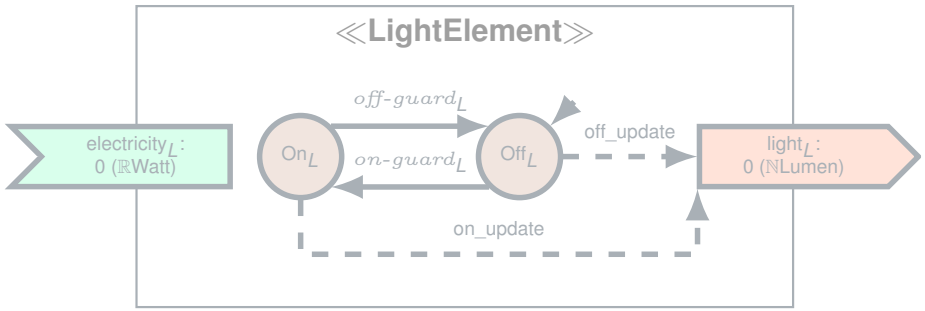
# The Language

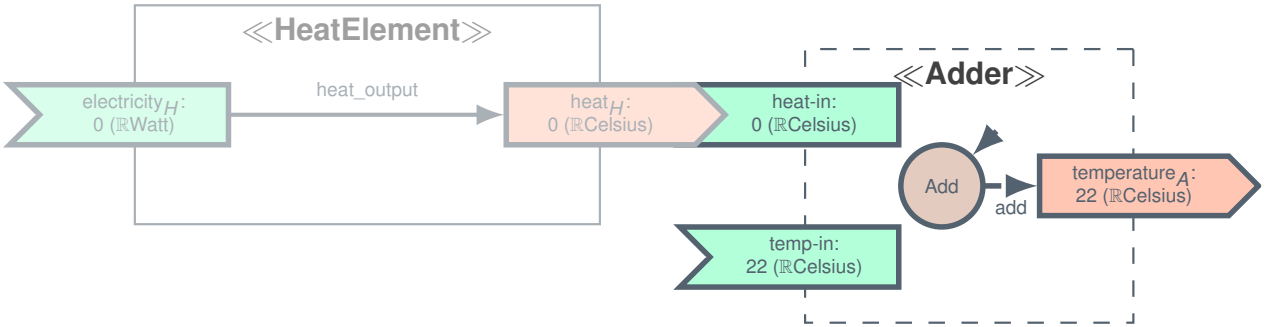


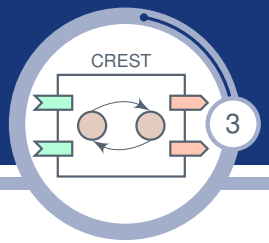
# The Language

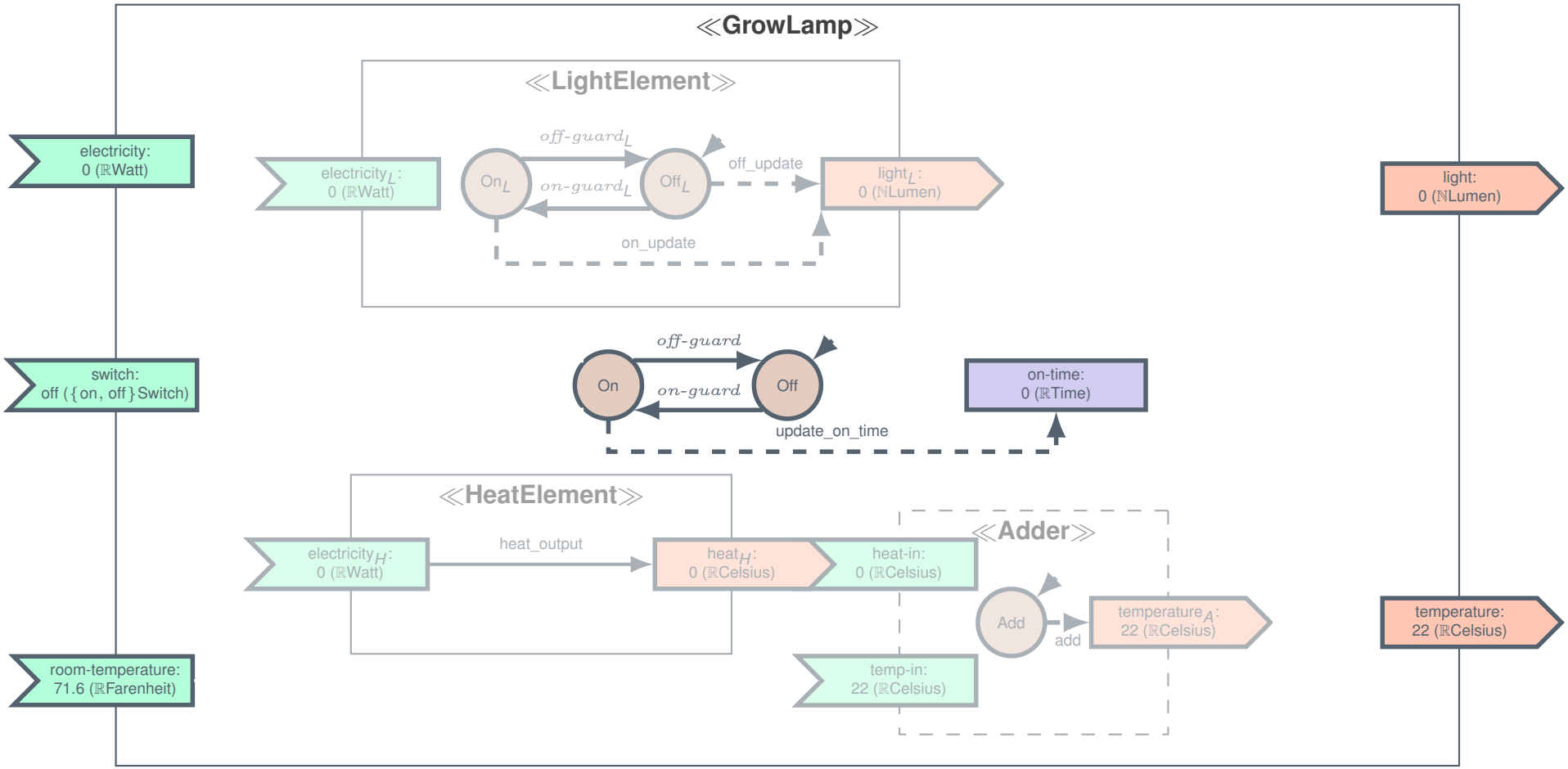




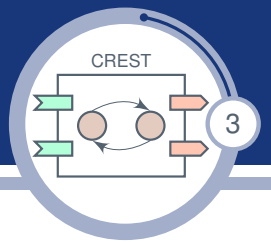



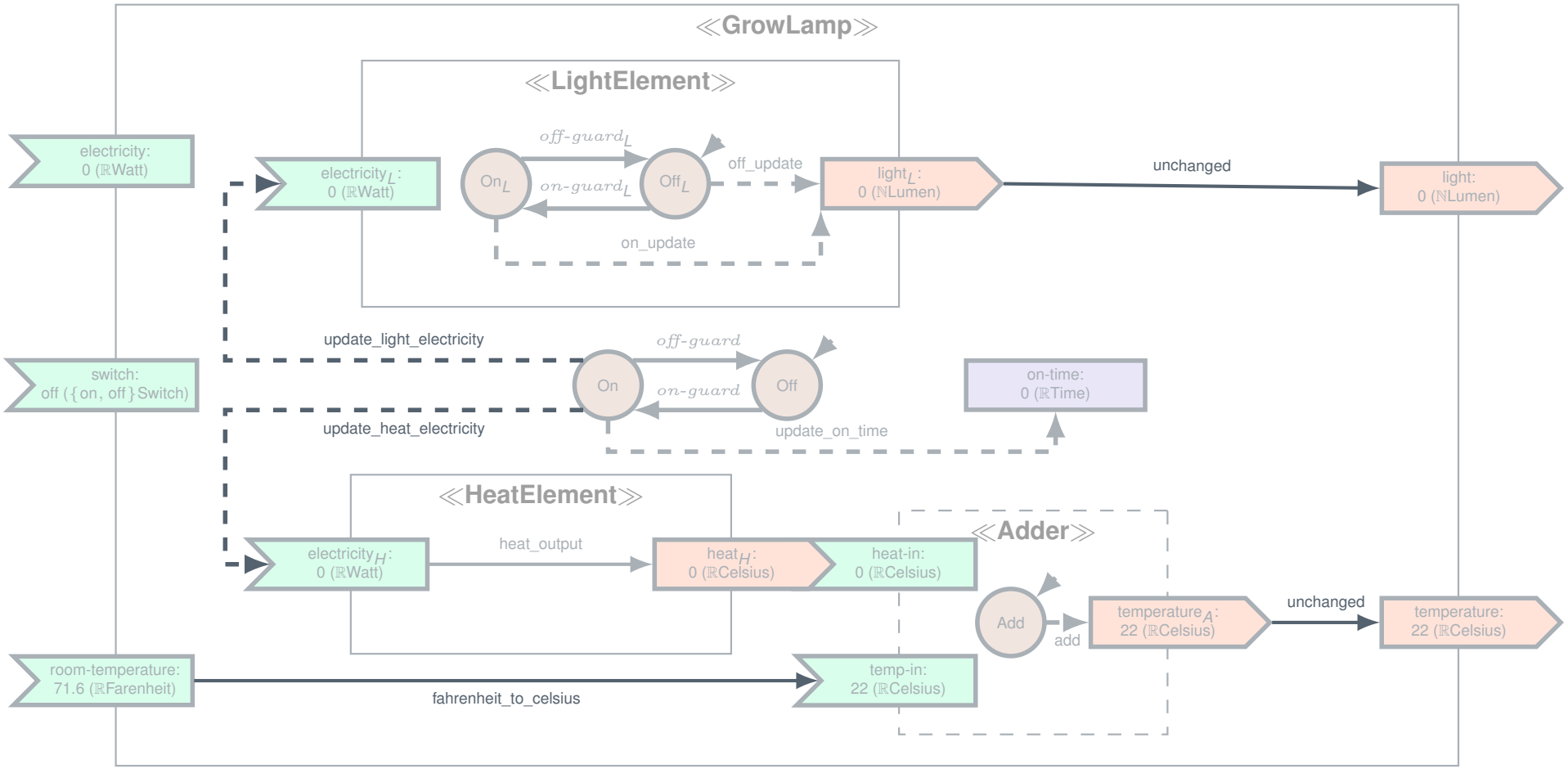
# The Language



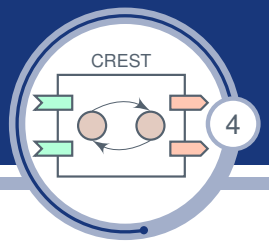



# The Language



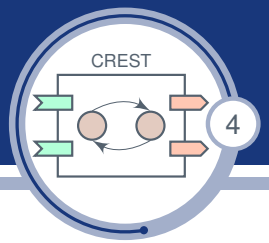



# Language Realization



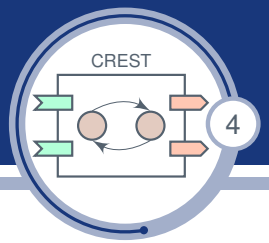

```
class LightElement(Entity):
```

# Language Realization



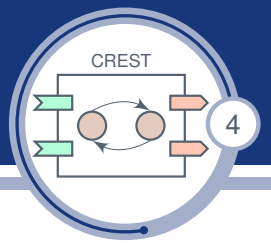

```
class LightElement(Entity):

  electricity = Input(resource=Watt, value=0)
  light       = Output(resource=Lumen, value=0)
```

# Language Realization



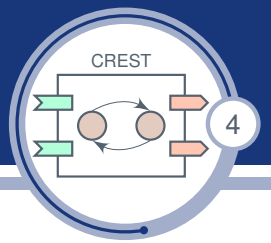

```
class LightElement(Entity):

  electricity = Input(resource=Watt, value=0)
  light       = Output(resource=Lumen, value=0)

  on = current = State()
  off = State()
```

# Language Realization



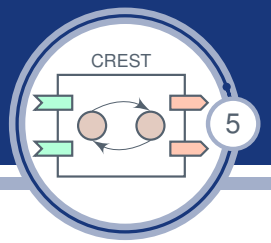

```
class LightElement(Entity):

  electricity = Input(resource=Watt, value=0)
  light       = Output(resource=Lumen, value=0)

  on = current = State()
  off = State()

  off_to_on = Transition(source=off, target=on,
        guard=(lambda self: self.switch.value == "on" and
                            self.electricity.value >= 100)
  )
```

# Language Realization




```
class LightElement(Entity):

  electricity = Input(resource=Watt, value=0)
  light       = Output(resource=Lumen, value=0)

  on = current = State()
  off = State()

  off_to_on = Transition(source=off, target=on,
        guard=(lambda self: self.switch.value == "on" and
                            self.electricity.value >= 100)
  )

  @update(state=on, target=light)
  def set_light_on(self, dt=0):
      return 800
```

# Why not «formalism-XYZ»?

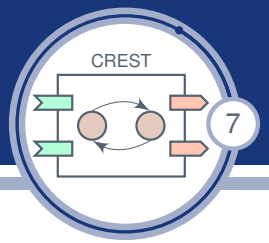



- powerful, but complex
- too generic
- feature workarounds
- architecture and behaviour

# One DSL to rule them all?

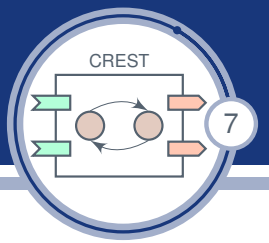


# CREST Formalization




*Domains*, *Units*

# CREST Formalization

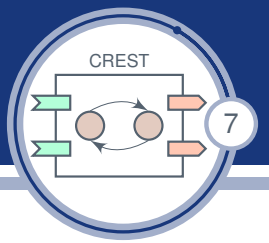



$$Domains \times Units = Types$$

$$\text{values} = \{\langle v, unit \rangle \mid v \in domain, \langle domain, unit \rangle \in Types\}$$

where

# CREST Formalization



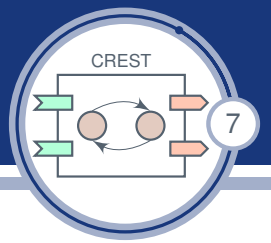

$$Domains \times Units = Types$$

$$\text{values} = \{\langle v, unit \rangle \mid v \in domain, \langle domain, unit \rangle \in Types\}$$

where

...

# CREST Formalization




$$Domains \times Units = Types$$

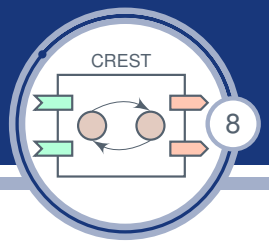
$$\text{values} = \{\langle v, unit\rangle \mid v \in domain, \langle domain, unit\rangle \in Types\}$$

where

...

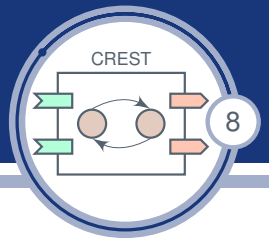
it's in the paper

# How to design a simulator

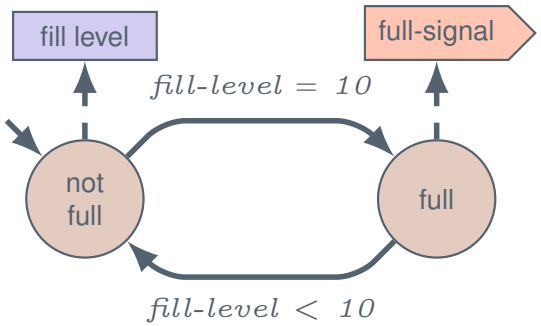



- advance time in constant steps?
- transition time calculation
- generate CREST diagrams

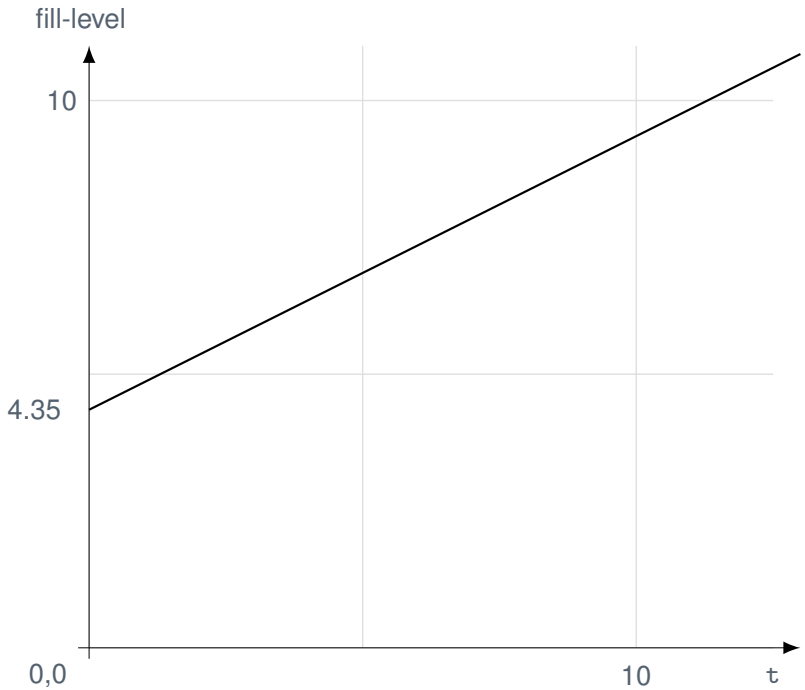
# How to design a simulator



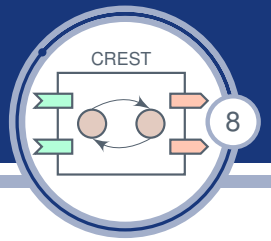

- advance time in constant steps?
- transition time calculation
- generate CREST diagrams



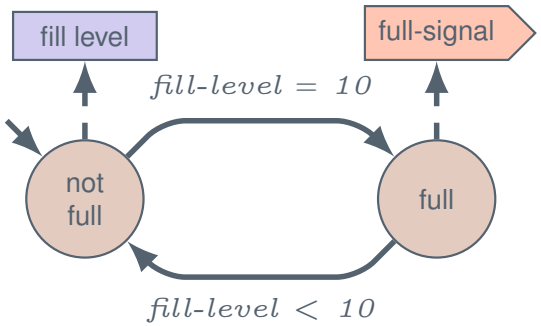

*flow-rate: 0.5 per time step*



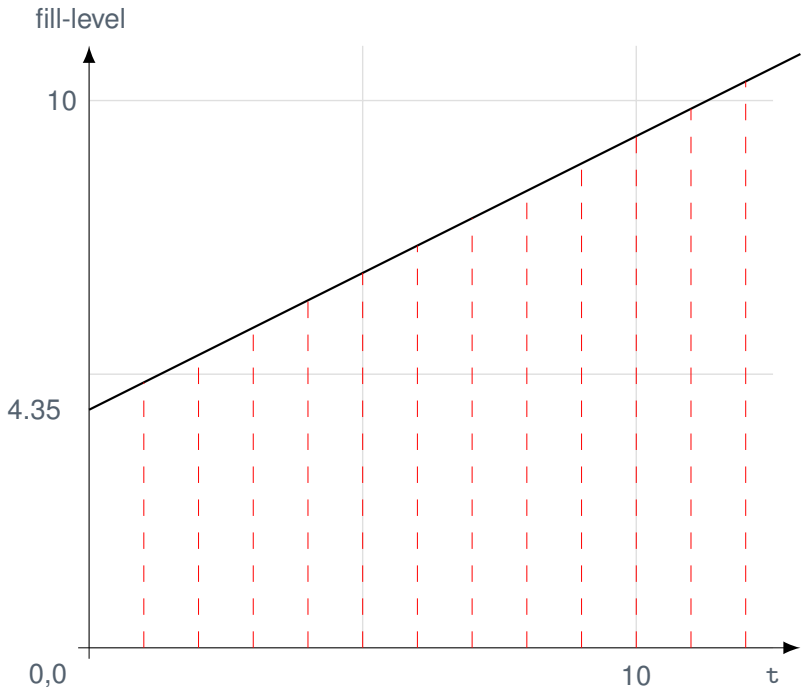
# How to design a simulator



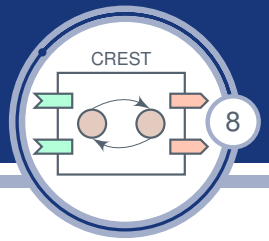

- advance time in constant steps?
- transition time calculation
- generate CREST diagrams



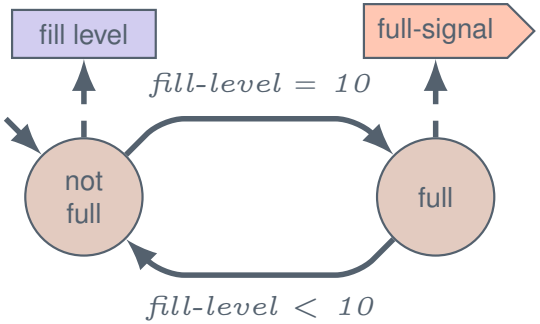

*flow-rate: 0.5 per time step*



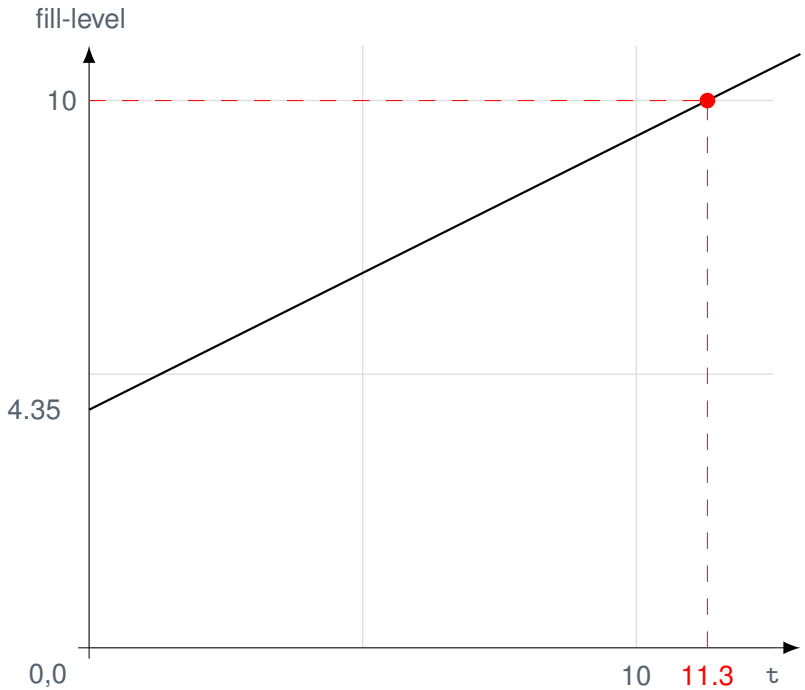
# How to design a simulator



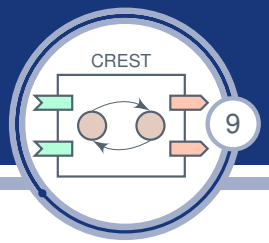

- advance time in constant steps?
- transition time calculation
- generate CREST diagrams



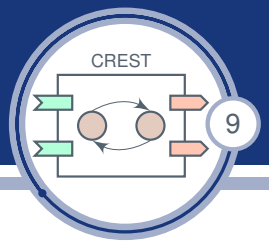

*flow-rate: 0.5 per time step*



# Advanced

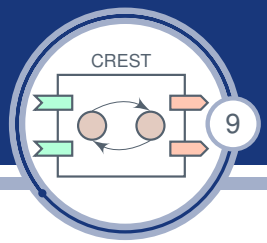


# Advanced



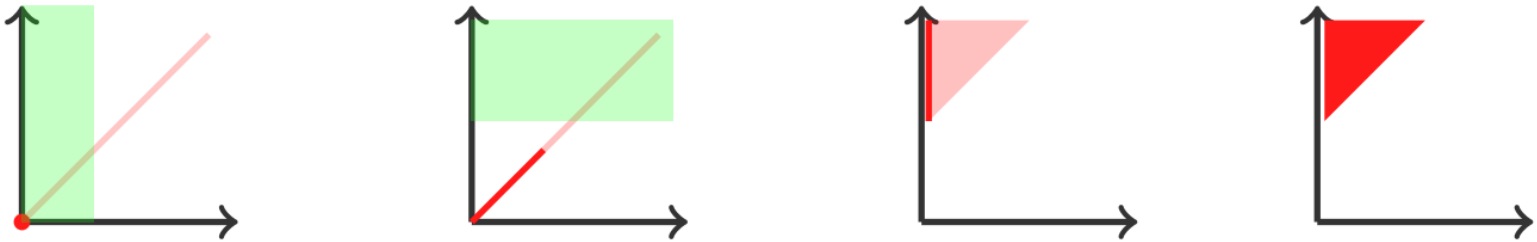

## Verification

# Advanced



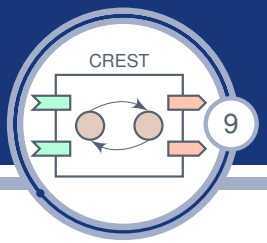

## Verification

- state space exploration
- zone/region-based verification



http://www-i2.informatik.rwth-aachen.de/ sri/Slides/sri-zonebased.pdf

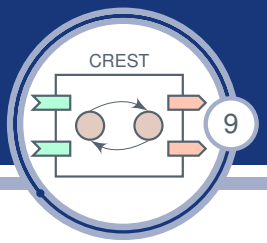
# Advanced




## Verification

- state space exploration
- zone/region-based verification
- requirements language

# Advanced

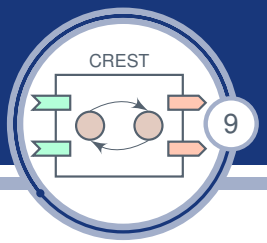



## Verification

- state space exploration
- zone/region-based verification
- requirements language

## Controller synthesis

# Advanced




## Verification

- state space exploration
- zone/region-based verification
- requirements language

## Controller synthesis

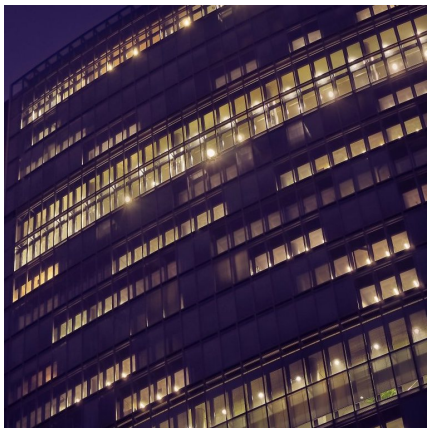
- planning + optimization + simulation
- changing parameters
- changing (sub-)systems

# Applications

plant growing

home automation

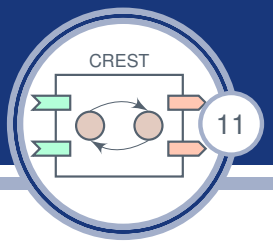
job scheduling



office automation

https://cs.wikipedia.org/wiki/Roomba#/media/File:IRobot_Roomba_780.jpg

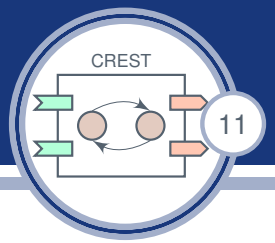
# Conclusion




CREST

- architecture & behaviour
- continuous & reactive
- formal basis

# Conclusion




CREST

- architecture & behaviour
- continuous & reactive
- formal basis

`https://mybinder.org/v2/gh/stklik/CREST/binder`

# CREST

## A Continuous, REactive SysTems DSL

Stefan Klikovits, Université de Genéve

@stklik

2018 CUSO Winter School