Goals to Operational Specifications: Getting it right!

Dalal Alrajeh
Department of Computing
Imperial College London

Nazareno Aguirre
Department of Computer Science
Universidad Nacional de Río Cuarto

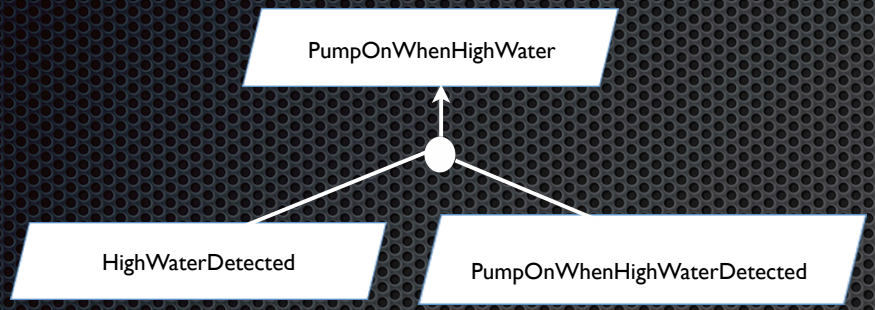
Best of RESG Research 2014
December 4th 2014

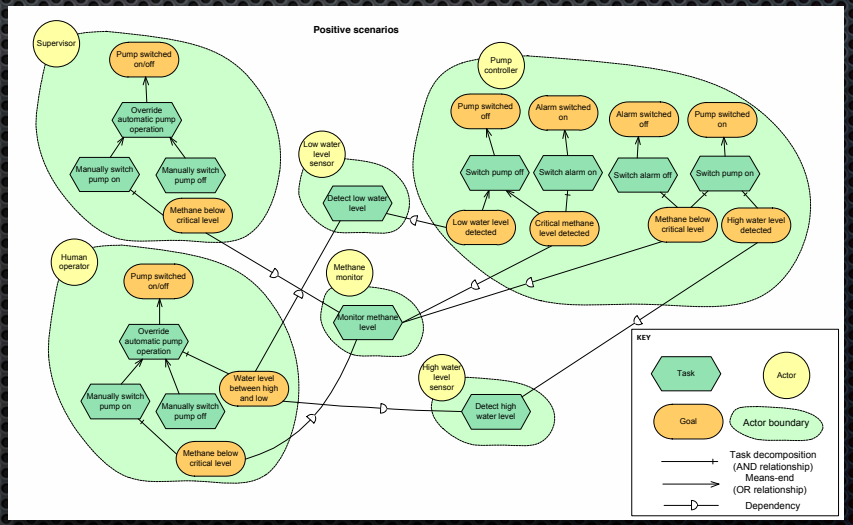
Detecting Errors Early

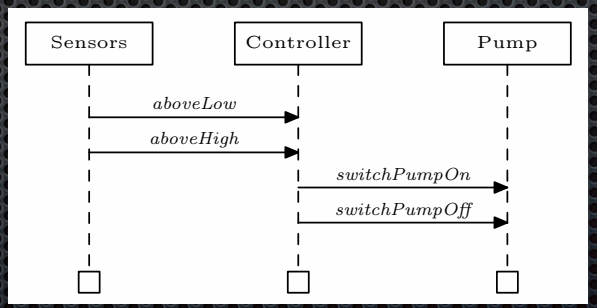
Detecting Errors Early

Software Requirements

Software Requirements

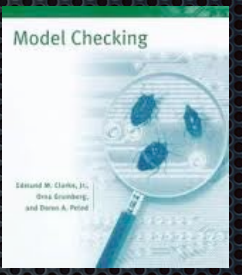


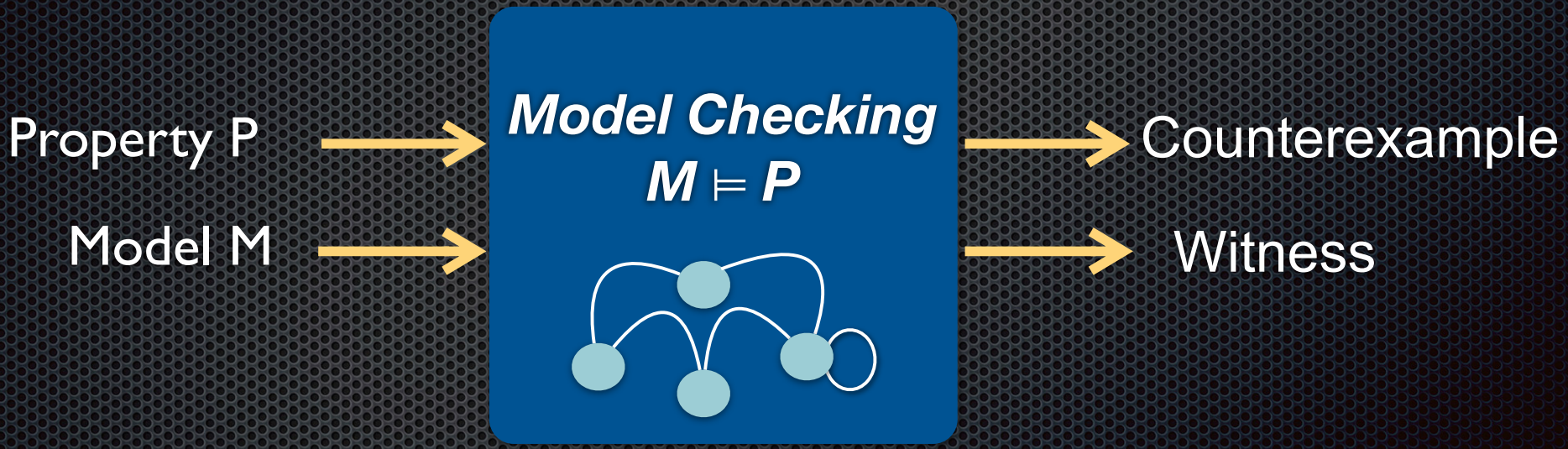




Model Checking

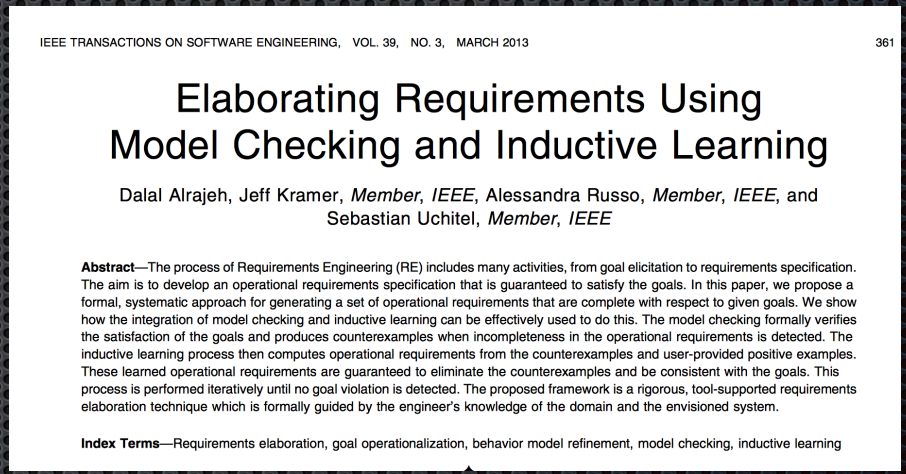
An automatic technique for verifying properties of finite state systems.





Extending Model Checking

Elaborating Requirements using ***model checking*** *and* ***inductive learning***
TSE'13



Elaborating Requirements Using Model Checking and Inductive Learning

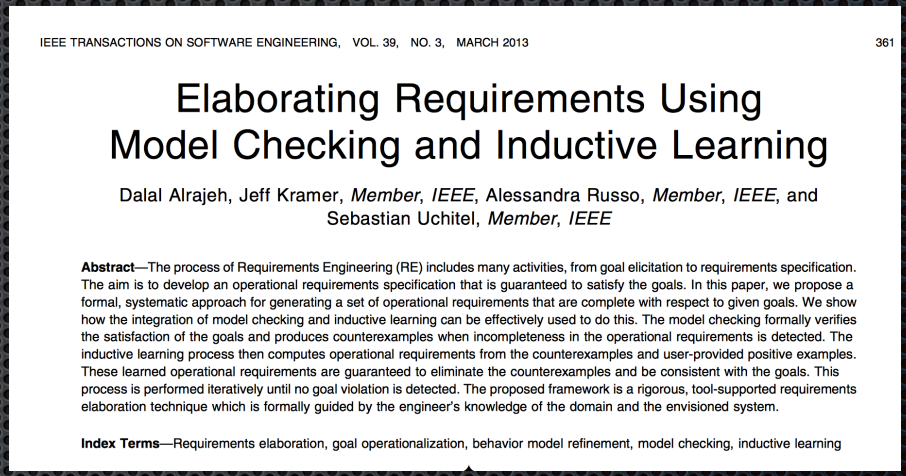
Dalal Alrajeh, Jeff Kramer, *Member, IEEE*, Alessandra Russo, *Member, IEEE*, and Sebastian Uchitel, *Member, IEEE*

Abstract—The process of Requirements Engineering (RE) includes many activities, from goal elicitation to requirements specification. The aim is to develop an operational requirements specification that is guaranteed to satisfy the goals. In this paper, we propose a formal, systematic approach for generating a set of operational requirements that are complete with respect to given goals. We show how the integration of model checking and inductive learning can be effectively used to do this. The model checking formally verifies the satisfaction of the goals and produces counterexamples when incompleteness in the operational requirements is detected. The inductive learning process then computes operational requirements from the counterexamples and user-provided positive examples. These learned operational requirements are guaranteed to eliminate the counterexamples and be consistent with the goals. This process is performed iteratively until no goal violation is detected. The proposed framework is a rigorous, tool-supported requirements elaboration technique which is formally guided by the engineer's knowledge of the domain and the envisioned system.

Index Terms—Requirements elaboration, goal operationalization, behavior model refinement, model checking, inductive learning

Extending Model Checking

Elaborating Requirements using ***model checking*** *and* ***inductive learning***
TSE'13



IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 39, NO. 3, MARCH 2013 361

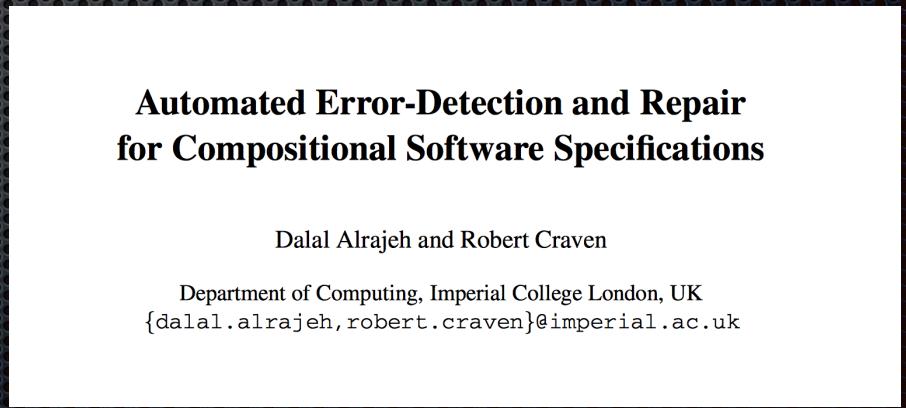
Elaborating Requirements Using Model Checking and Inductive Learning

Dalal Alrajeh, Jeff Kramer, *Member, IEEE*, Alessandra Russo, *Member, IEEE*, and Sebastian Uchitel, *Member, IEEE*

Abstract—The process of Requirements Engineering (RE) includes many activities, from goal elicitation to requirements specification. The aim is to develop an operational requirements specification that is guaranteed to satisfy the goals. In this paper, we propose a formal, systematic approach for generating a set of operational requirements that are complete with respect to given goals. We show how the integration of model checking and inductive learning can be effectively used to do this. The model checking formally verifies the satisfaction of the goals and produces counterexamples when incompleteness in the operational requirements is detected. The inductive learning process then computes operational requirements from the counterexamples and user-provided positive examples. These learned operational requirements are guaranteed to eliminate the counterexamples and be consistent with the goals. This process is performed iteratively until no goal violation is detected. The proposed framework is a rigorous, tool-supported requirements elaboration technique which is formally guided by the engineer's knowledge of the domain and the envisioned system.

Index Terms—Requirements elaboration, goal operationalization, behavior model refinement, model checking, inductive learning

Repairing compositional specifications using ***model checking*** *and* ***theory revision***
SEFM'14



Automated Error-Detection and Repair for Compositional Software Specifications

Dalal Alrajeh and Robert Craven

Department of Computing, Imperial College London, UK
{dalal.alrajeh,robert.craven}@imperial.ac.uk

Automated Goal Operationalisation Based on Interpolation and SAT Solving *

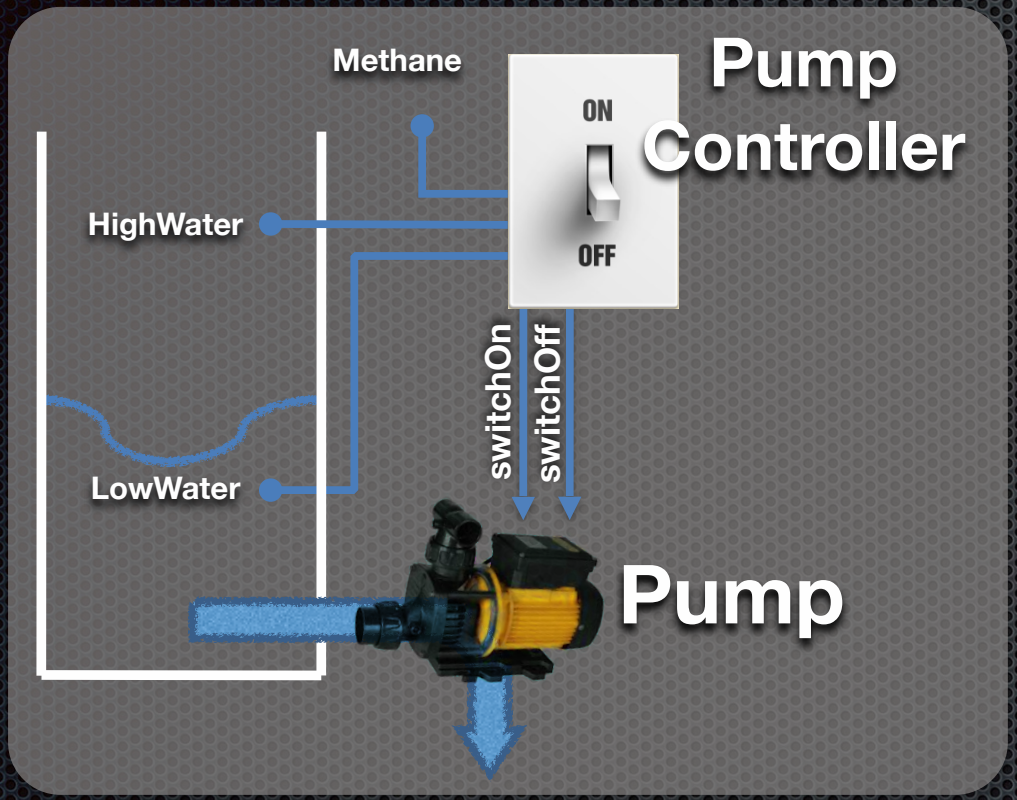
Renzo Degiovanni* Dalal Alrajeh† Nazareno Aguirre* Sebastian Uchitel†‡

*Departamento de Computación, Universidad Nacional de Río Cuarto and CONICET, Argentina
†Department of Computing, Imperial College London, UK
‡Departamento de Computación, Universidad de Buenos Aires and CONICET, Argentina
{rdegiovanni, naguirre}@dc.exa.unrc.edu.ar, {dalal.alrajeh04, s.uchitel}@imperial.ac.uk

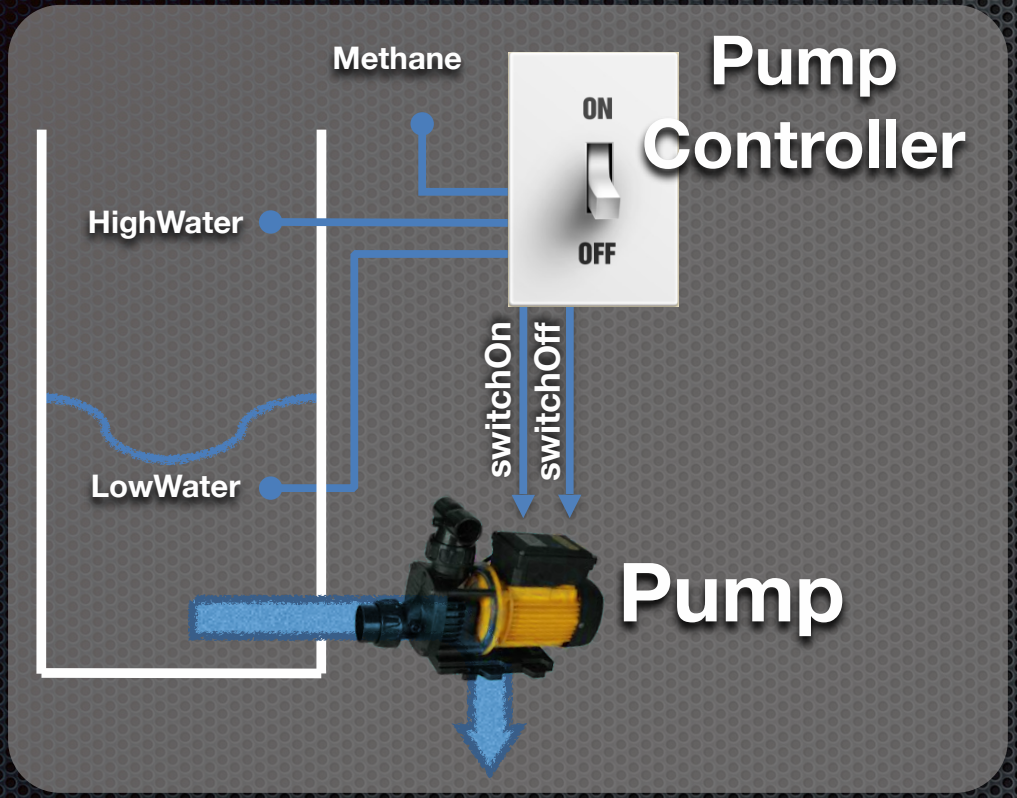
ICSE 2014

Motivating Example

Motivating Example



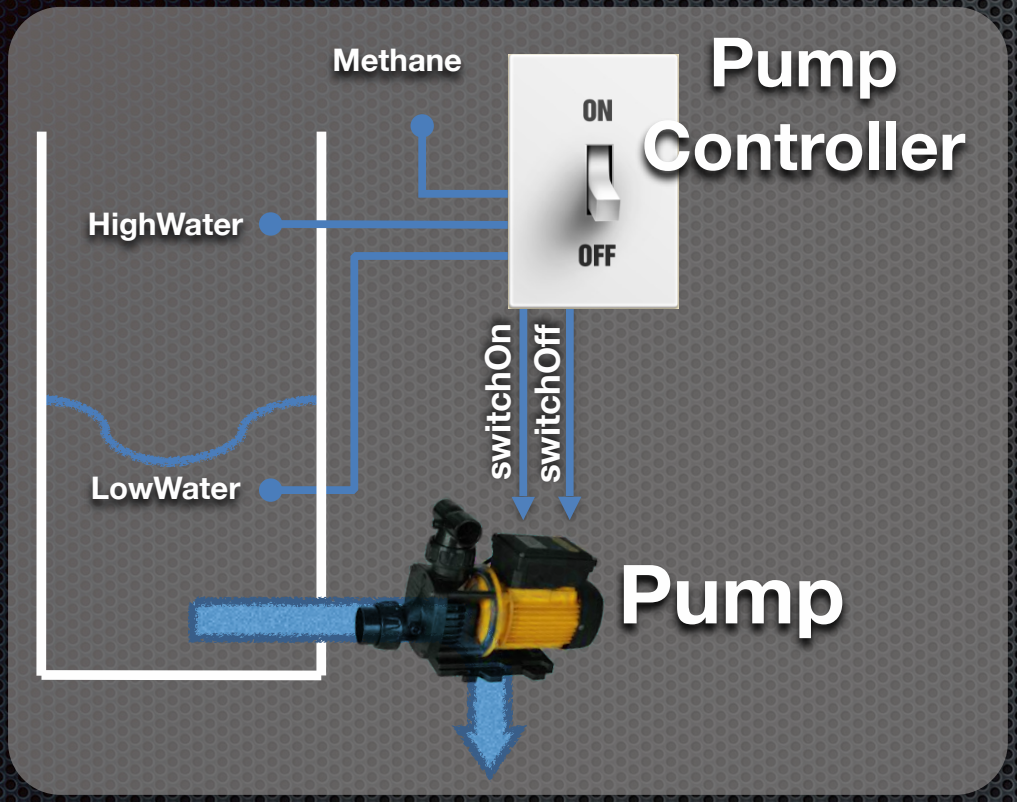
Motivating Example



switchOn
Pre: Pump=Off
Trig: ——
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off

Motivating Example



switchOn
Pre: Pump=Off
Trig: ——
Post: Pump=On

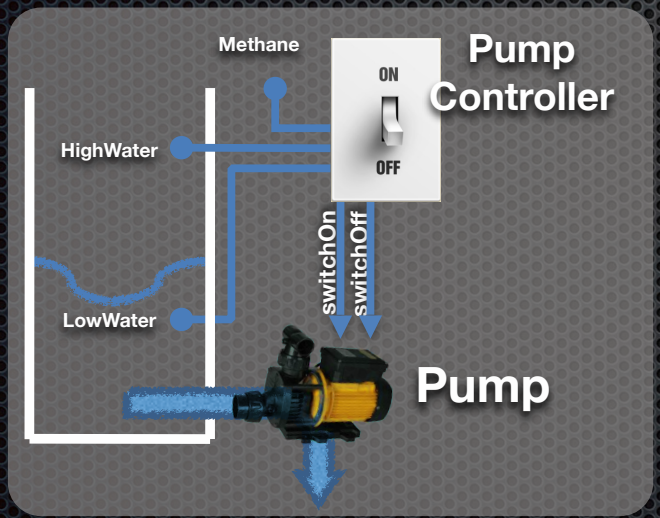
switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off

If the water level is high and there is no methane, then the pump should be on.

Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: ——
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off



If the water level is high and there is no methane, then the pump should be on.

Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: ——
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off

cannot add controlled operations

If the water level is high and there is no methane, then the pump should be on.

Goal Operationalisation

permission

switchOn

Pre: Pump=Off **and ???**

Trig: — —

Post: Pump=On

switchOff

Pre: Pump=Off **and ???**

Trig: — —

Post: Pump=Off

Agent's abilities remain **static**

cannot add controlled operations

If the water level is high and there is no methane, then the pump should be on.

Goal Operationalisation

permission

switchOn

Pre: Pump=Off **and ???**

Trig: **???**

Post: Pump=On

obligation

switchOff

Pre: Pump=Off **and ???**

Trig: **???**

Post: Pump=Off

cannot add controlled operations

If the water level is high and there is no methane, then the pump should be on.

Goal Operationalisation

permission

switchOn

Pre: Pump=Off **and ???**

Trig: **???**

Post: Pump=On

obligation

switchOff

Pre: Pump=Off **and ???**

Trig: **???**

Post: Pump=Off

Agent's abilities remain **static**

Correct operationalisation

If the water level is high and there is no methane, then the pump should be on.

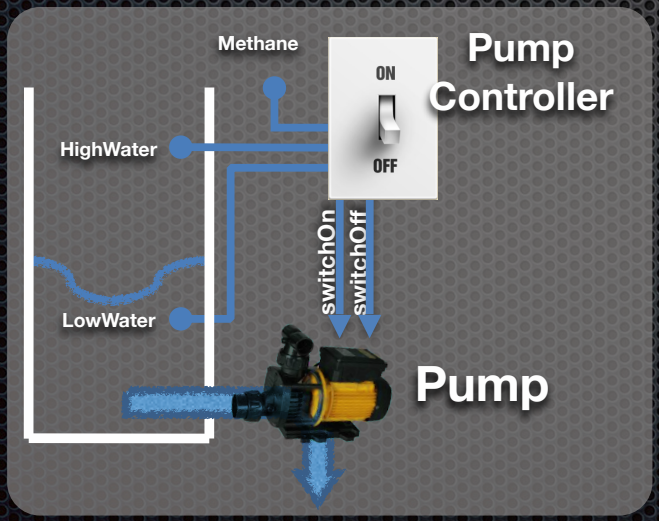
cannot add controlled operations

Goal Operationalisation

(iterative/interactive)

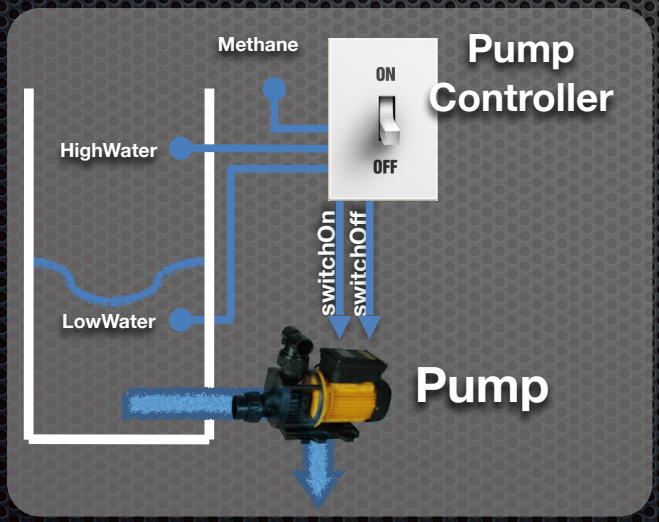
Goal Operationalisation

(iterative/interactive)



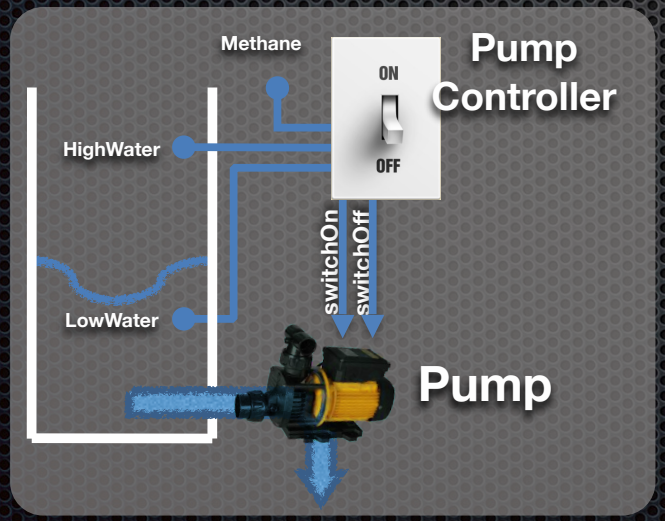
Goal Operationalisation

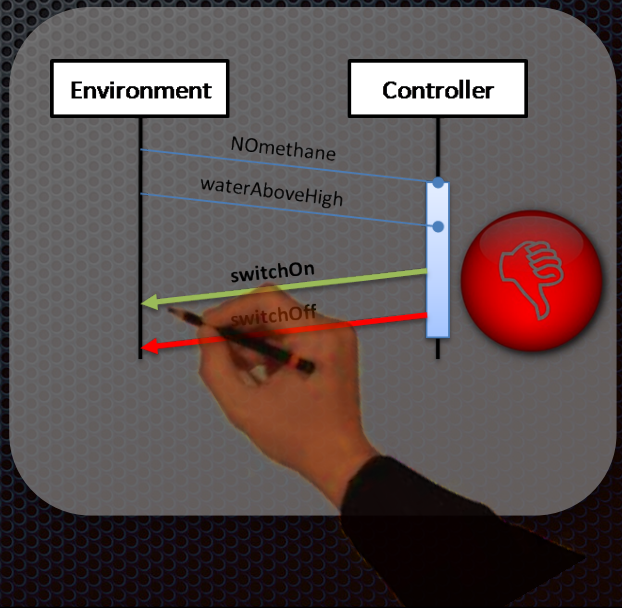
(iterative/interactive)



Goal Operationalisation

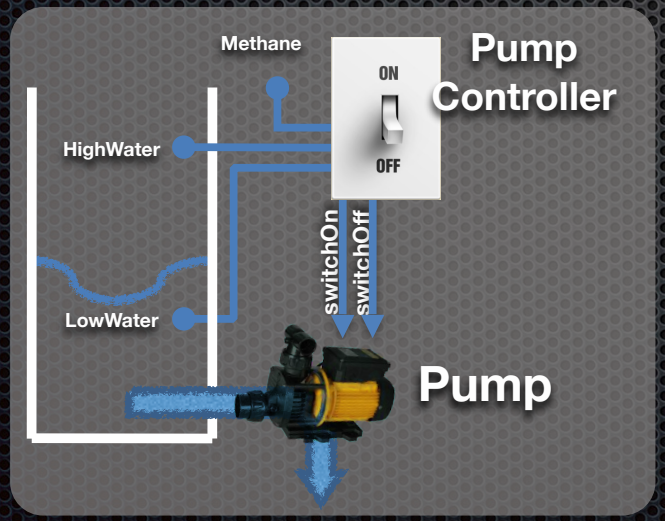
(iterative/interactive)



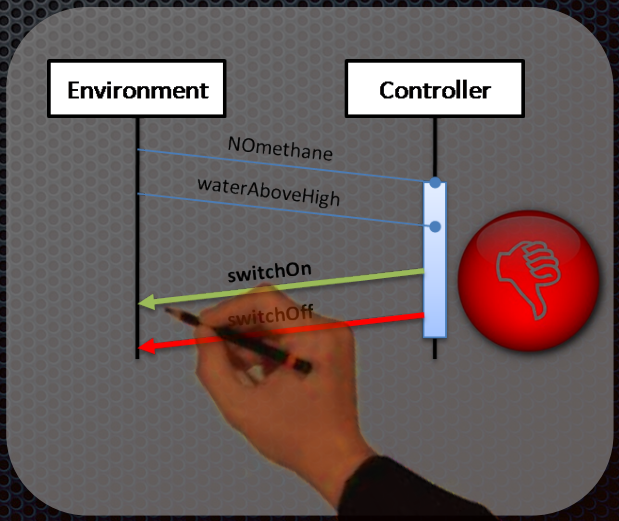


Goal Operationalisation

(iterative/interactive)



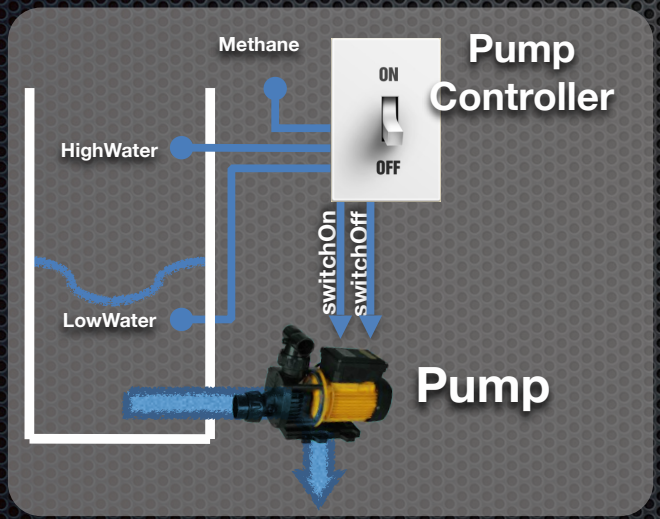
Operation Model Redesign



Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: ——
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off

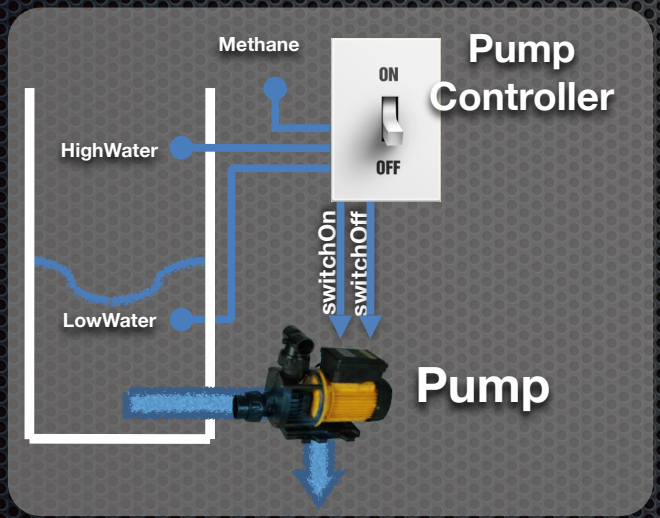


If the water level is high and there is no methane, then the pump should be on.

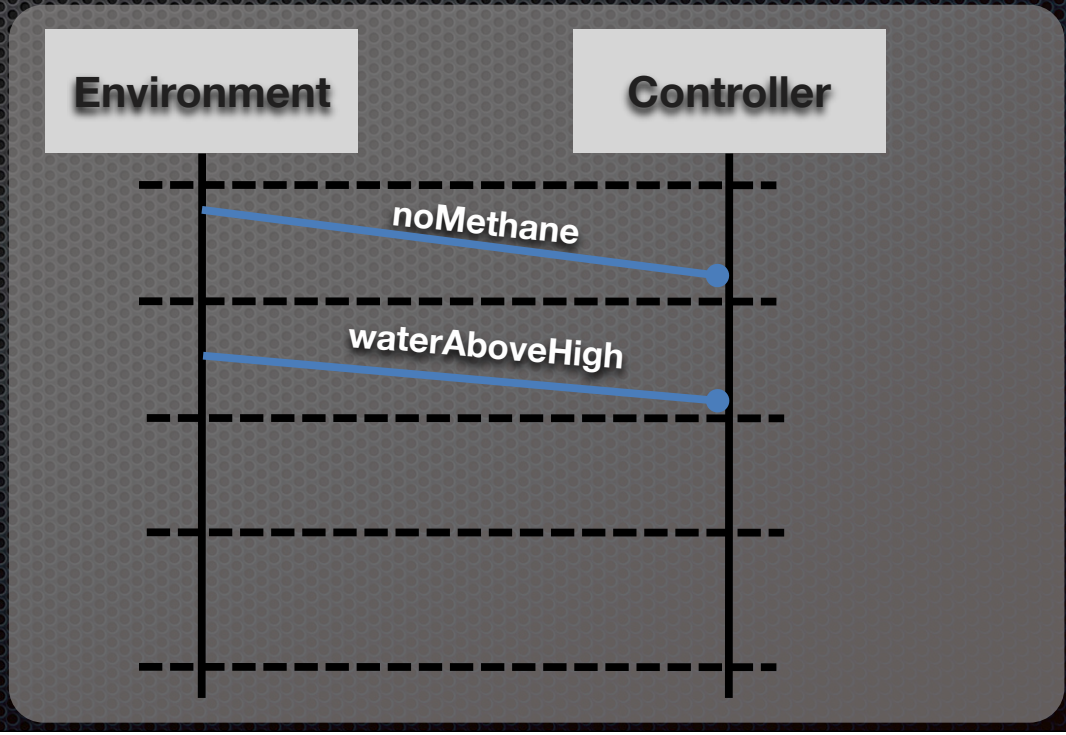
Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: — —
Post: Pump=On

switchOff
Pre: Pump=On
Trig: — —
Post: Pump=Off



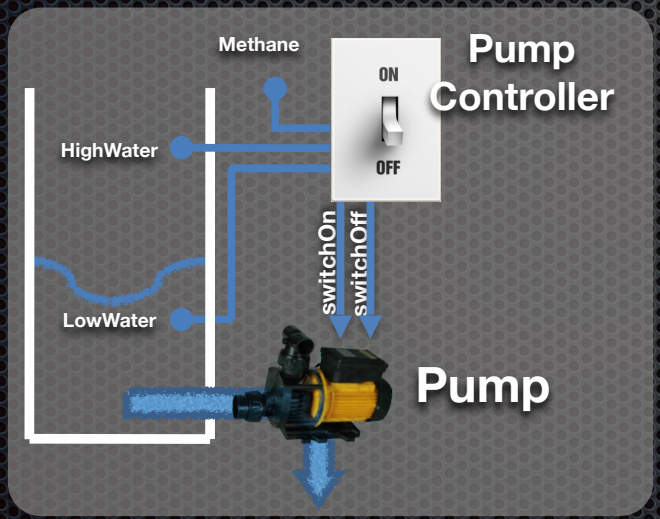
If the water level is high and there is no methane, then the pump should be on.



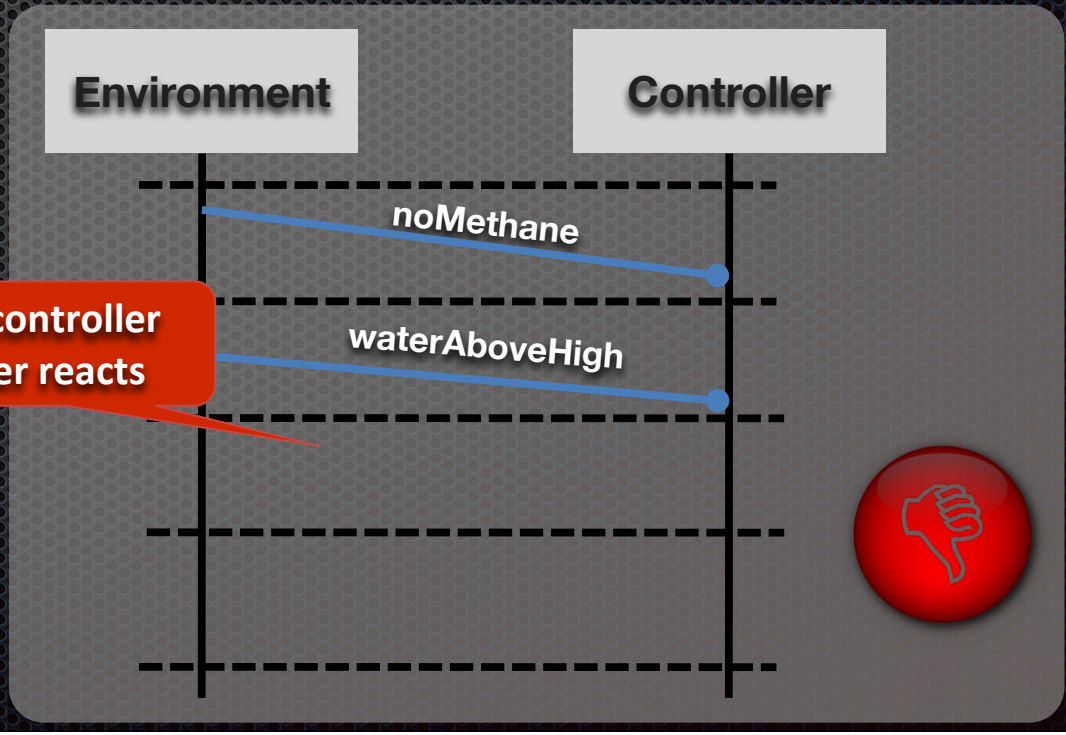
Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: — —
Post: Pump=On

switchOff
Pre: Pump=On
Trig: — —
Post: Pump=Off



If the water level is high and there is no methane, then the pump should be on.

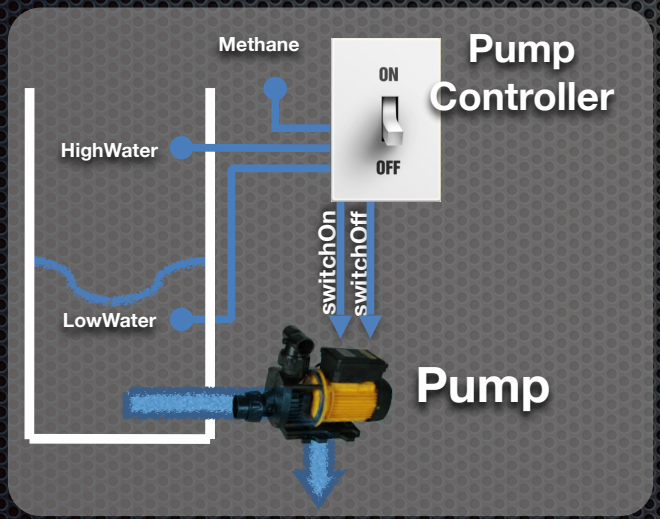


Goal Operationalisation

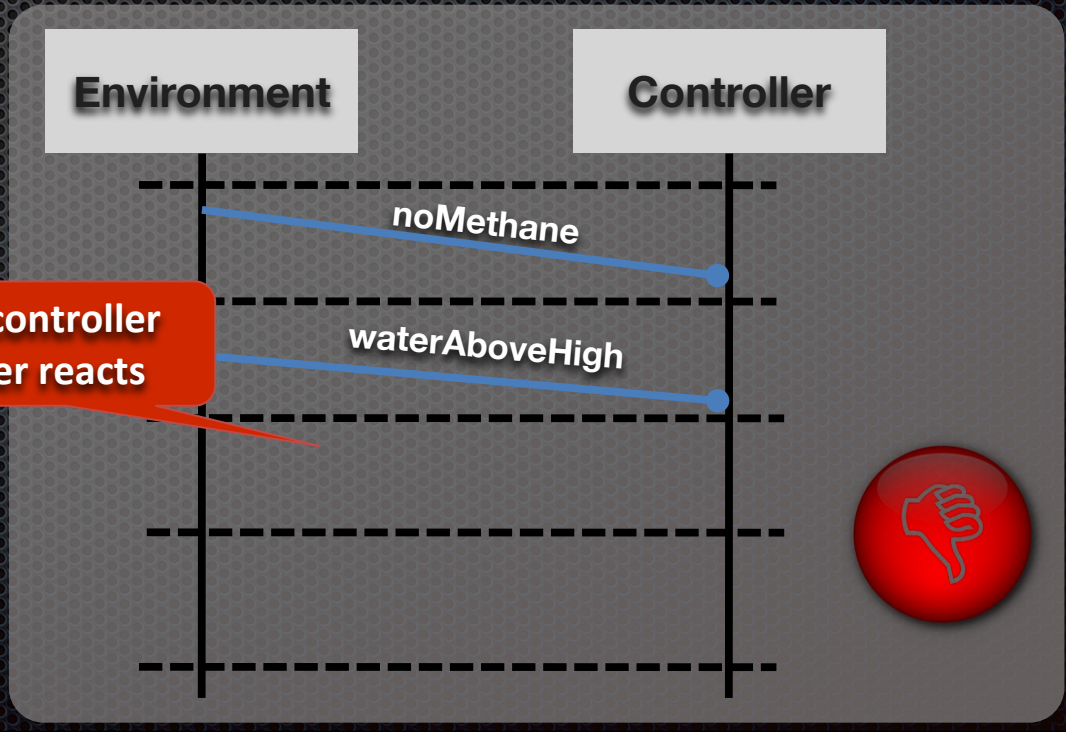
Trigger this operation

switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off



If the water level is high and there is no methane, then the pump should be on.

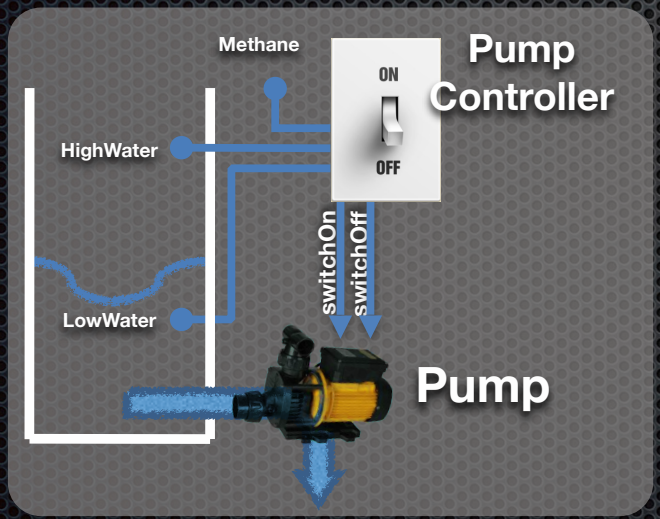


Goal Operationalisation

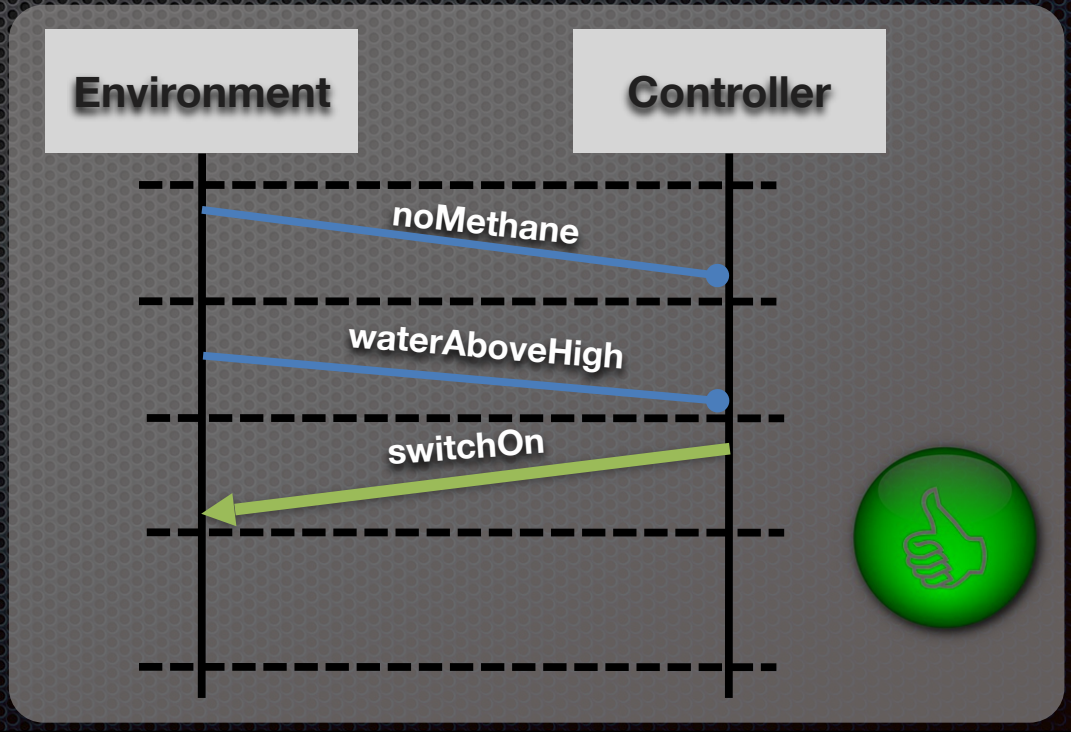
Trigger this operation

switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off



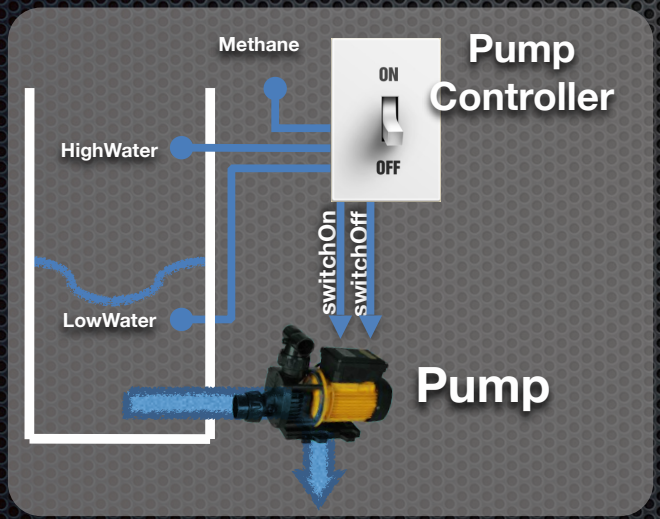
If the water level is high and there is no methane, then the pump should be on.



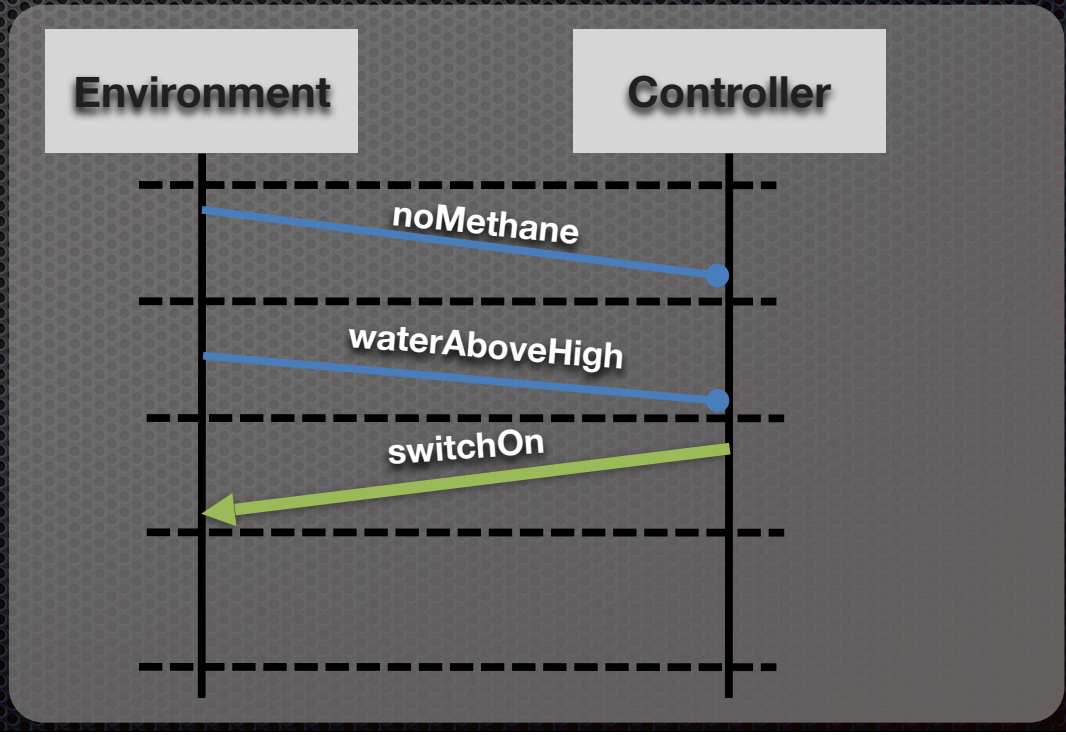
Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off



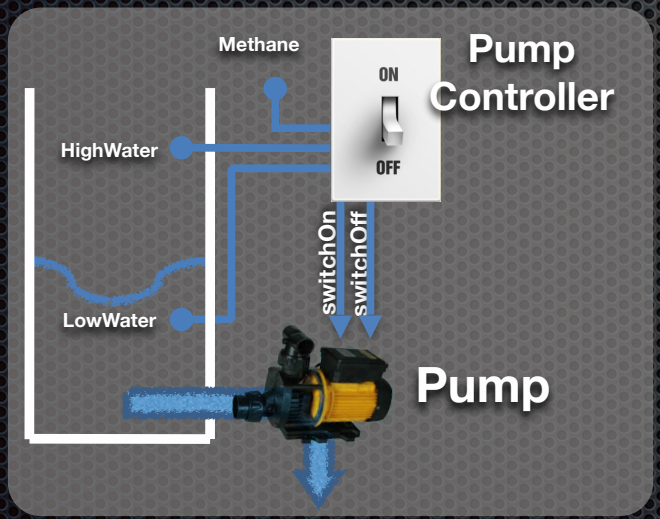
If the water level is high and there is no methane, then the pump should be on.



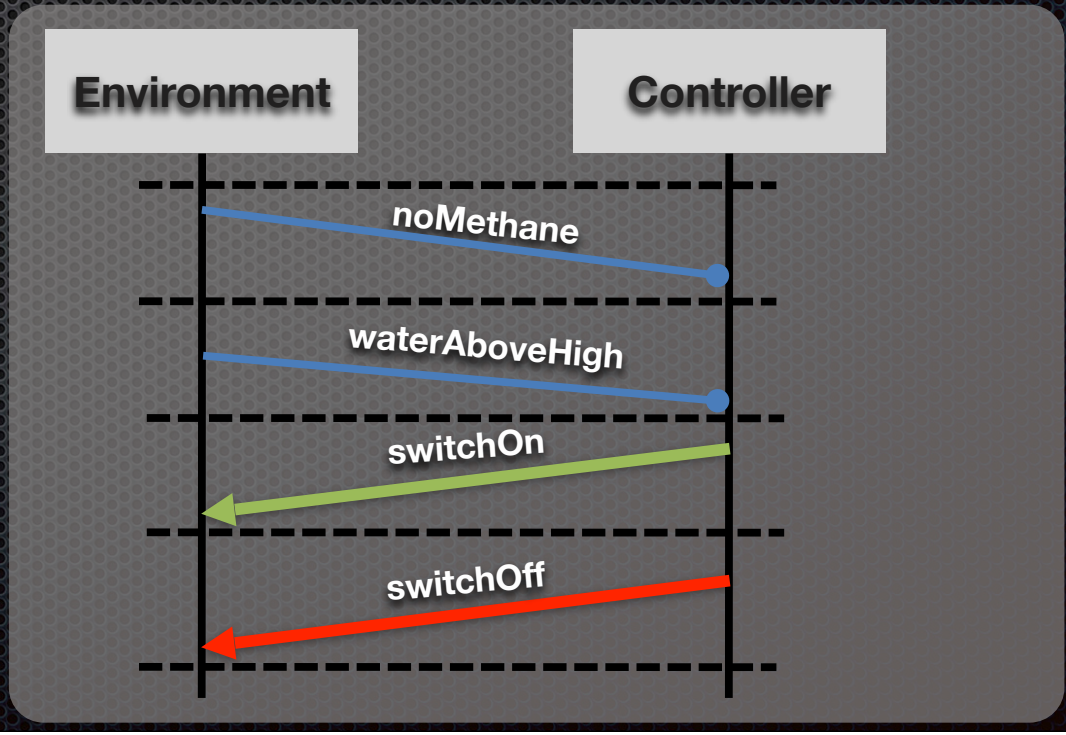
Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off



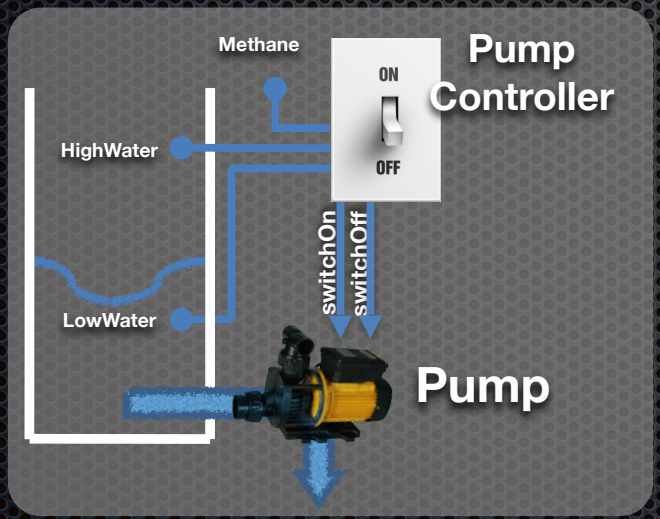
If the water level is high and there is no methane, then the pump should be on.



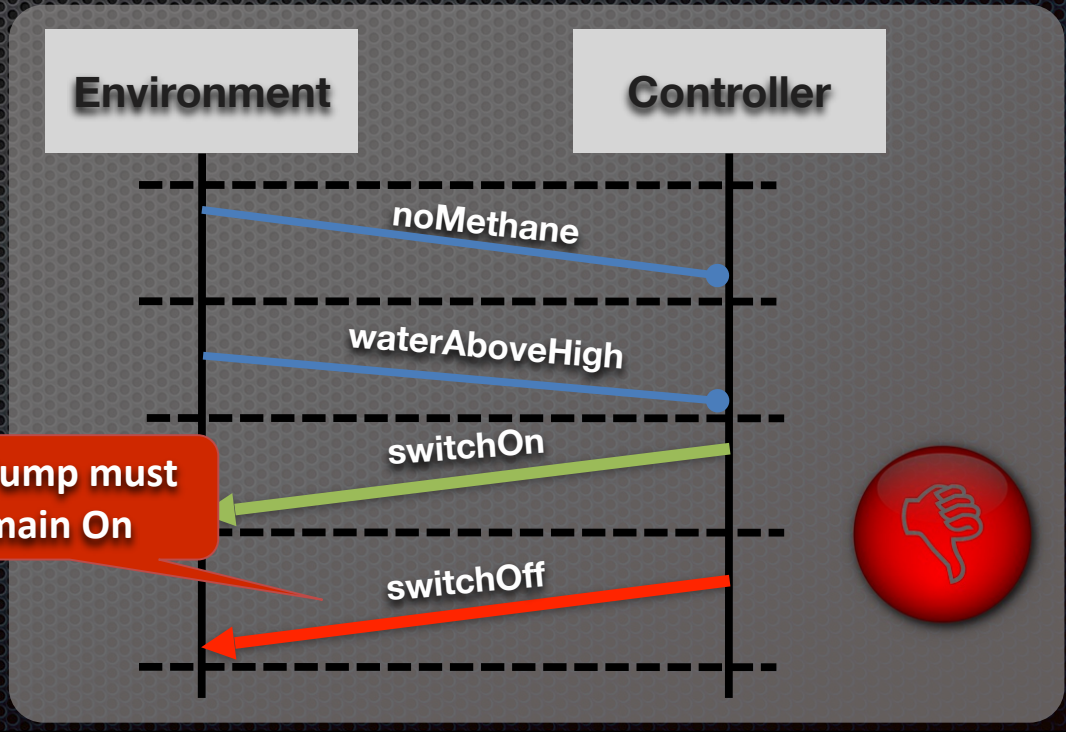
Goal Operationalisation

switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off



If the water level is high and there is no methane, then the pump should be on.



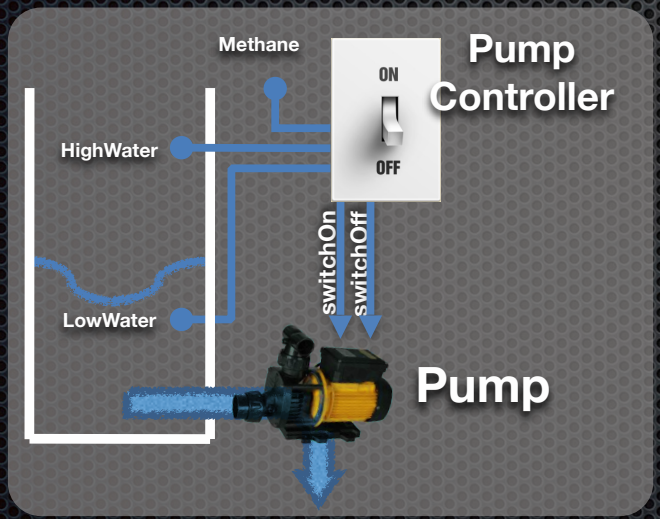
Goal Operationalisation

Restrict this operation

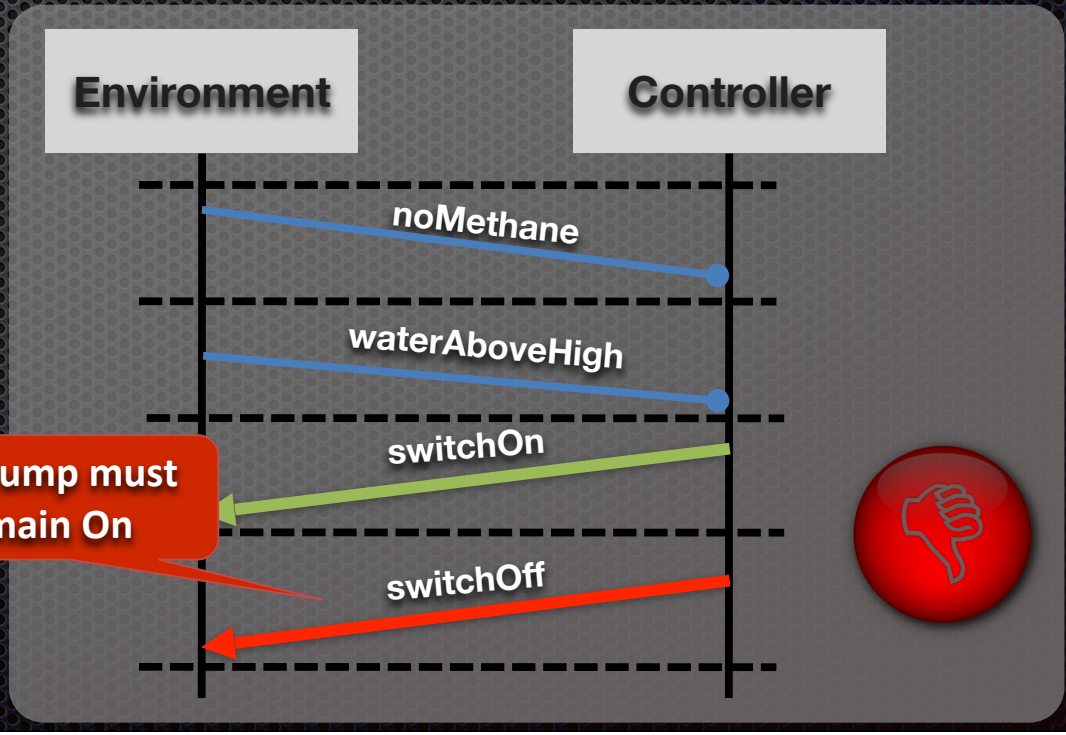
switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off

and no (HighWater and no Methane)



If the water level is high and there is no methane, then the pump should be on.

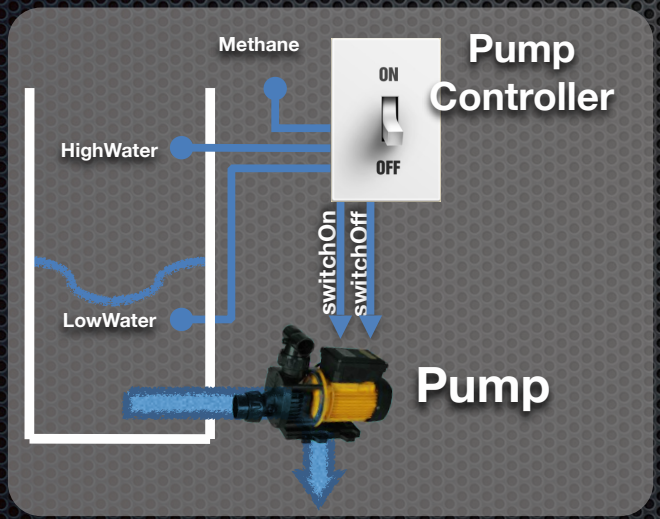


Goal Operationalisation

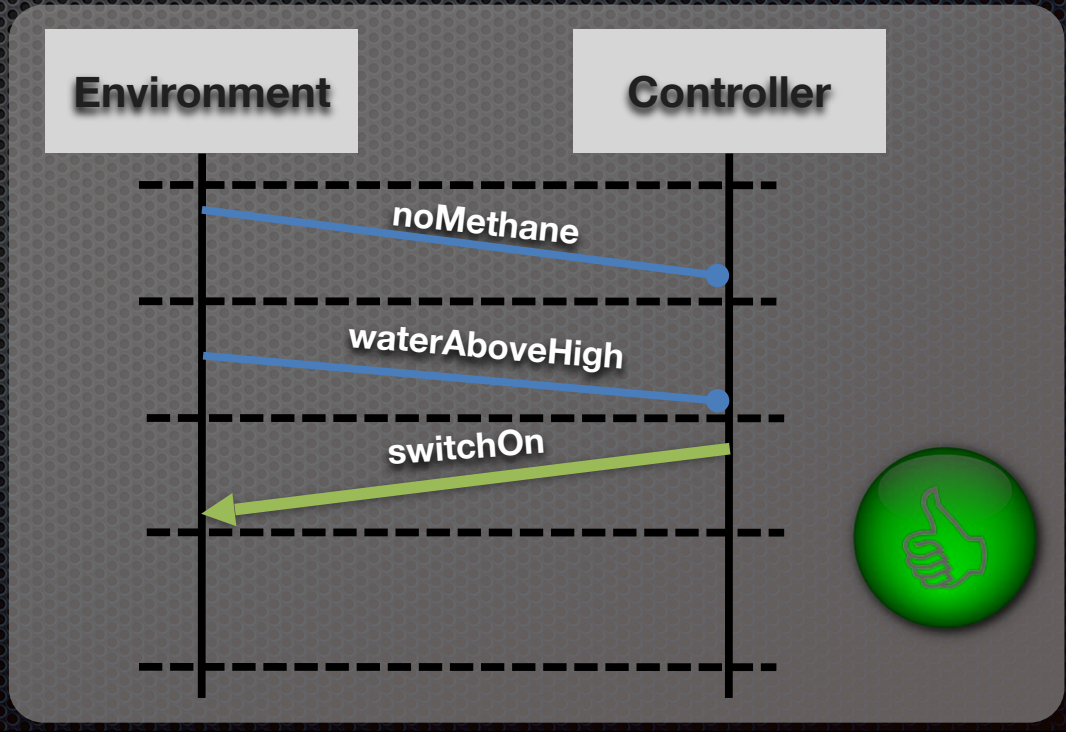
Restrict this operation

switchOn
Pre: Pump=Off
Trig: HighWater and no Methane
Post: Pump=On

switchOff
Pre: Pump=On
Trig: ——
Post: Pump=Off
and no (HighWater and no Methane)



If the water level is high and there is no methane, then the pump should be on.



Can this process be **automated**?

Can this process be **automated**?

- Unwanted scenarios generation/detection?
  - YES
    - Model Checking

Can this process be **automated**?

- Unwanted scenarios generation/detection?
  - YES
    - Model Checking
- Goal Operationalisation?
  - State of the Art: **Partially**

Can this process be **automated**?

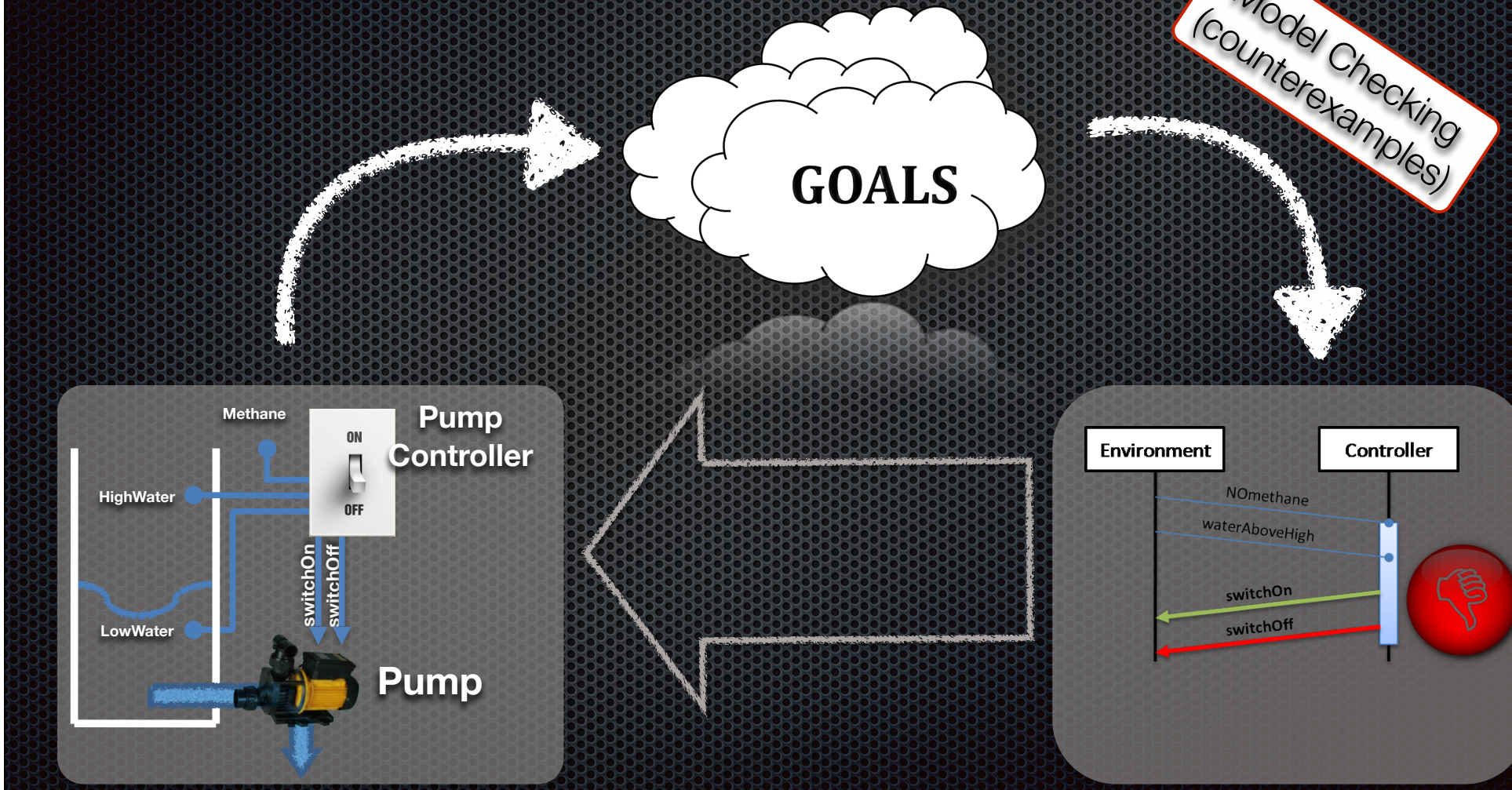
- Unwanted scenarios generation/detection?
  - YES
    - Model Checking
- Goal Operationalisation?
  - State of the Art: **Partially Semiautomated**
    - Inductive techniques
      - Requires user assistance.

Can this process be **automated**?

- Unwanted scenarios generation/detection?
  - YES
    - Model Checking
- Goal Operationalisation?
  - State of the Art: **Partially Semiautomated**
    - Inductive techniques
    - Requires user assistance.
    - Safety Goals
    - Progress
    - not Liveness

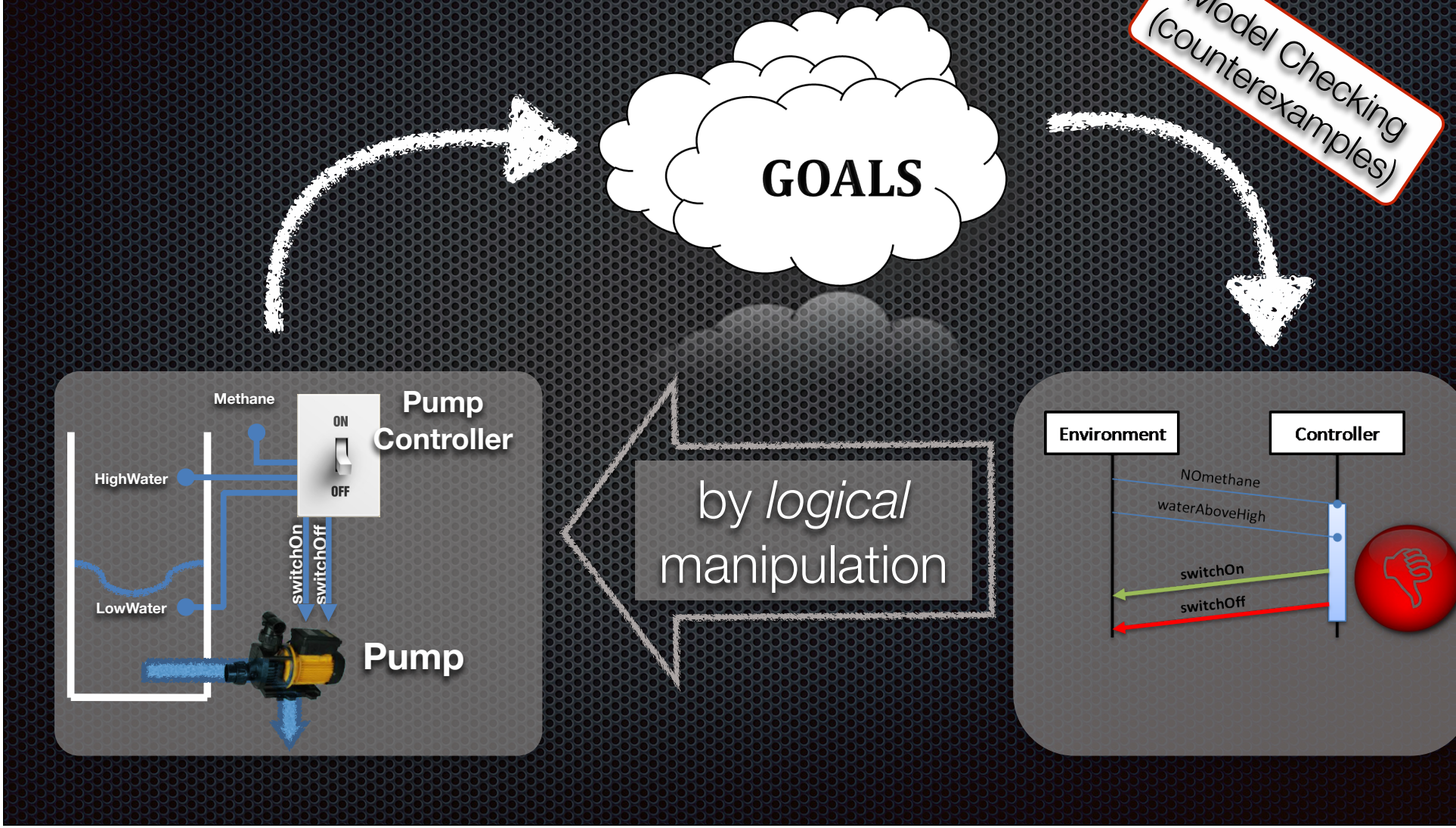
Goal Operationalisation

(iterative/**automated**)



Goal Operationalisation

(iterative/**automated**)



Goal Operationalisation

(iterative/**automated**)

Model Checking (counterexamples)

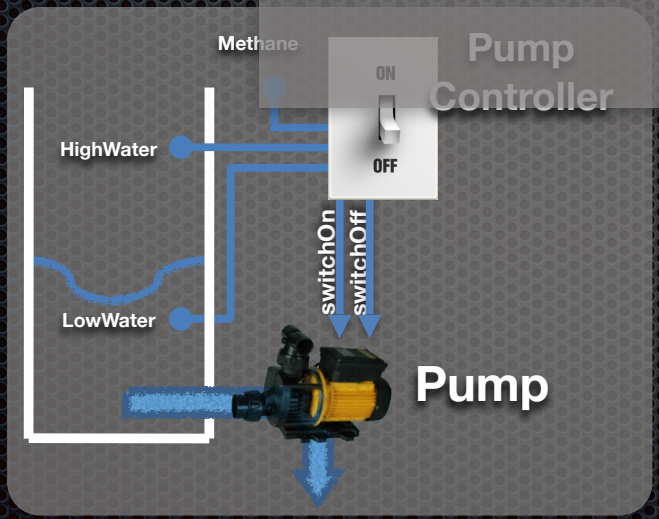
Automated

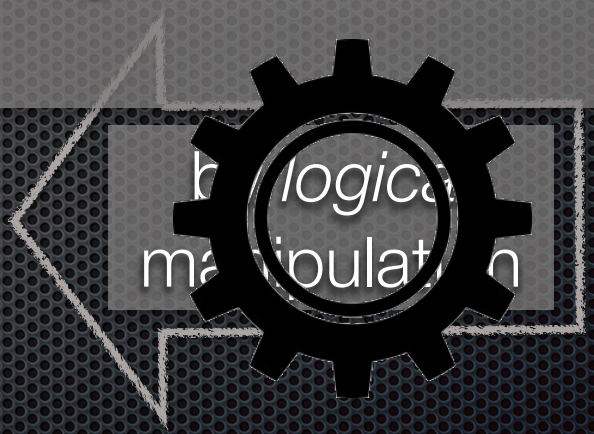
- Interpolation
- SAT solving

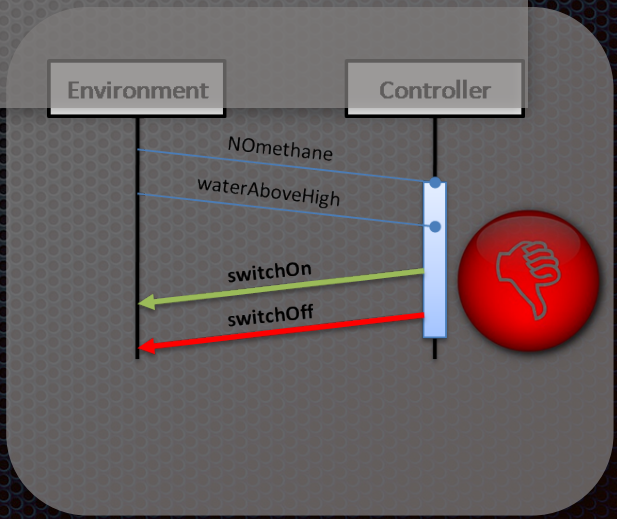
Safety Goals

Progress

Liveness







Interpolation

Interpolation

Interpolation

- Given *A* and *B*, such that $A \wedge B$ is inconsistent, an interpolant for *A* and *B* is a formula ***I*** with the following properties:

Interpolation

- Given *A* and *B*, such that $A \wedge B$ is inconsistent, an interpolant for *A* and *B* is a formula ***I*** with the following properties:

$$A \Longrightarrow I$$

Interpolation

- Given *A* and *B*, such that $A \wedge B$ is inconsistent, an interpolant for *A* and *B* is a formula ***I*** with the following properties:

$$A \Longrightarrow I$$

$$UNSAT(I, B)$$

Interpolation

- Given *A* and *B*, such that $A \wedge B$ is inconsistent, an interpolant for *A* and *B* is a formula ***I*** with the following properties:

$$A \Longrightarrow I$$

$$UNSAT(I, B)$$

I is in the common language of *A* and *B*

Interpolation

- Given A a… $A \wedge B$ is inconsis… A and B is a formula I … properties:

"**explains**" why A is inconsistent with B

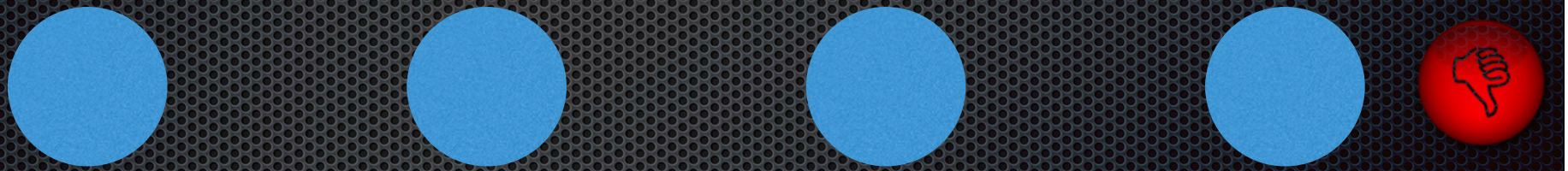
$$A \Rightarrow I$$

$$UNSAT(I, B)$$

I is in the common language of A and B

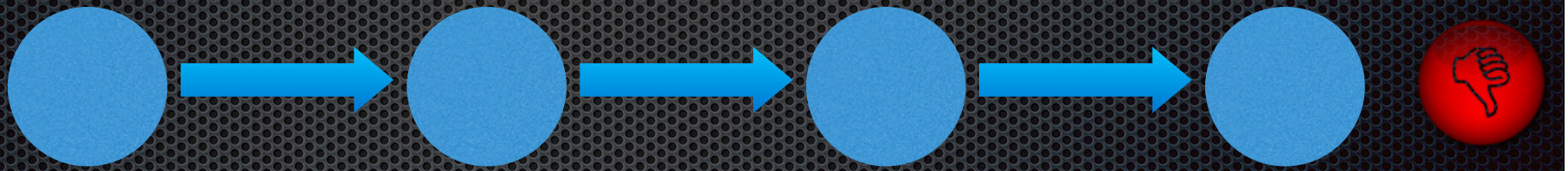
Interpolation

- Succesfully used for **removing counterexamples**
  - Abstraction Based Model Checking
    - CEGAR



Interpolation

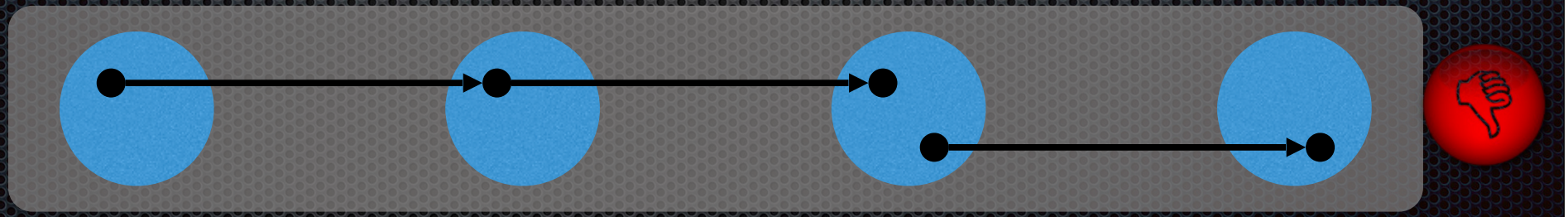
- Succesfully used for **removing counterexamples**
  - Abstraction Based Model Checking
    - CEGAR



Interpolation

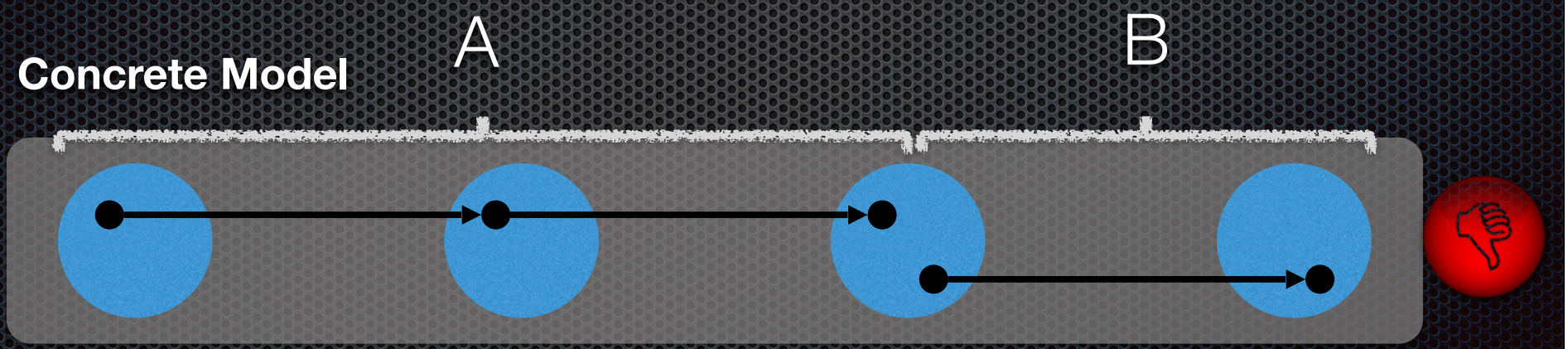
- Succesfully used for **removing counterexamples**
  - Abstraction Based Model Checking
    - CEGAR

Concrete Model



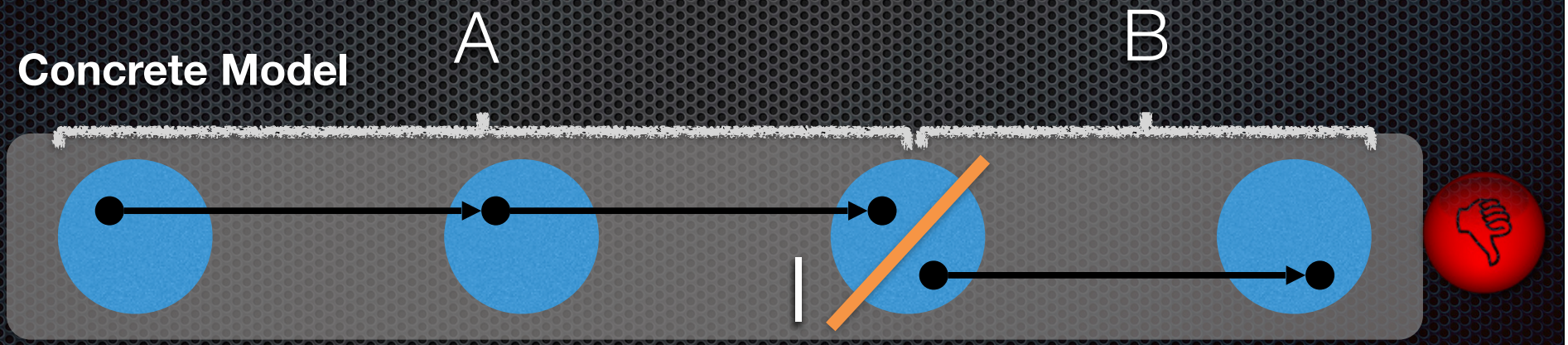
Interpolation

- Succesfully used for **removing counterexamples**
  - Abstraction Based Model Checking
    - CEGAR



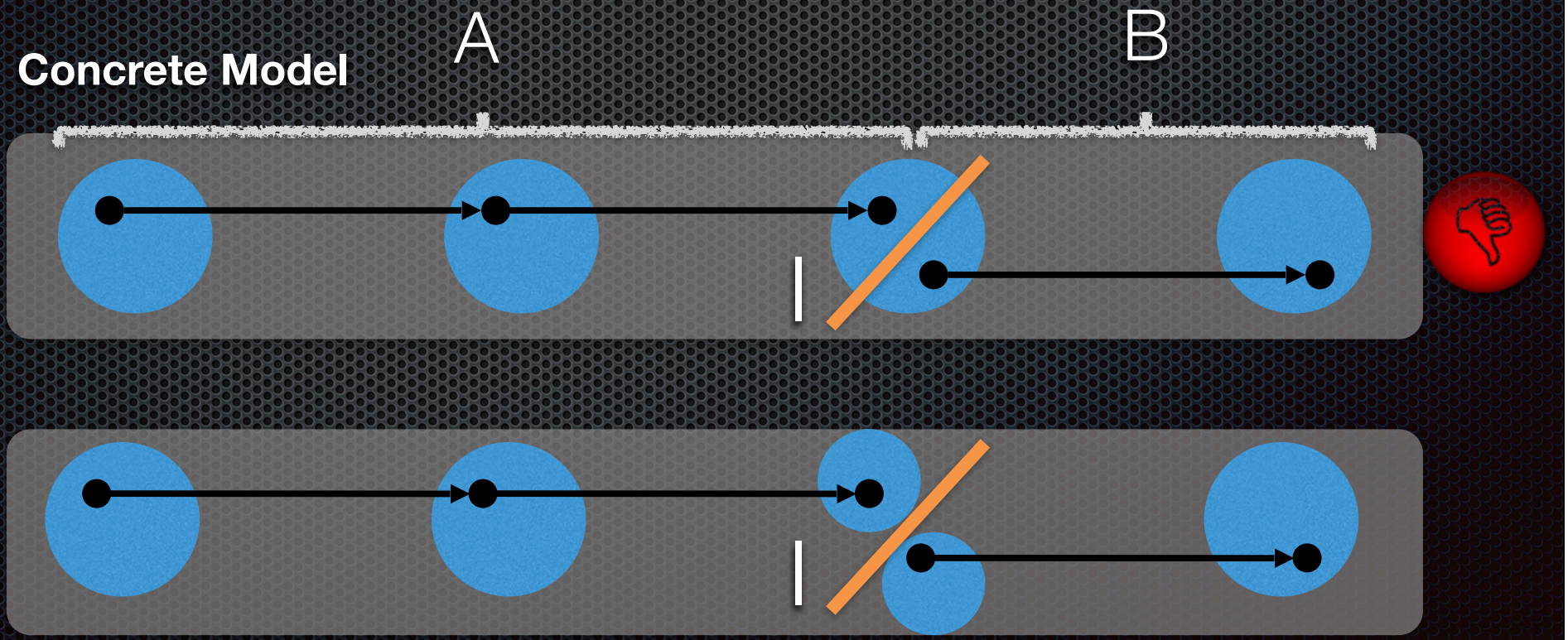
Interpolation

- Succesfully used for **removing counterexamples**
  - Abstraction Based Model Checking
    - CEGAR



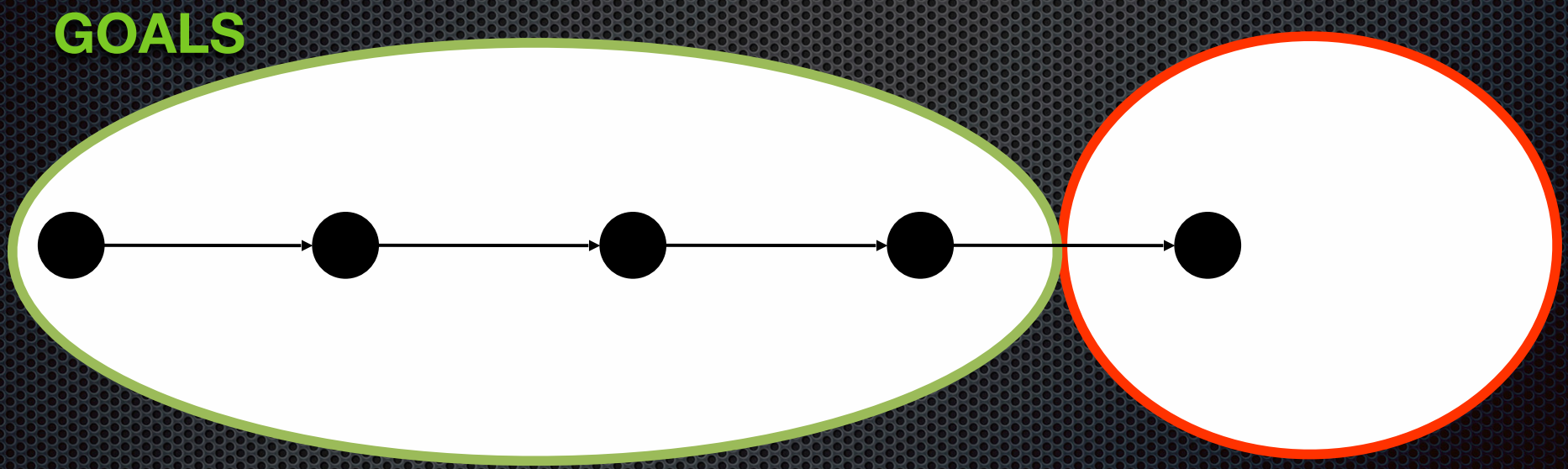
Interpolation

- Succesfully used for **removing counterexamples**
  - Abstraction Based Model Checking
    - CEGAR



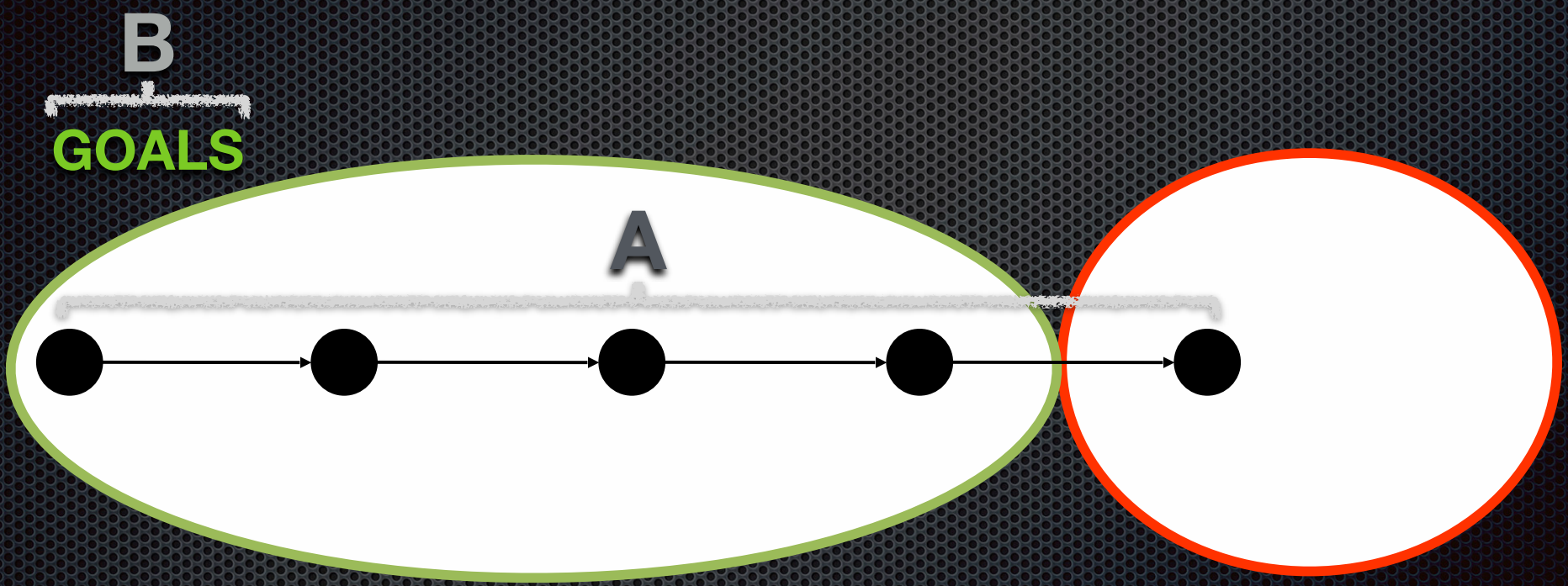
Requirements Refinement

- Our setting: **Concrete counterexamples**



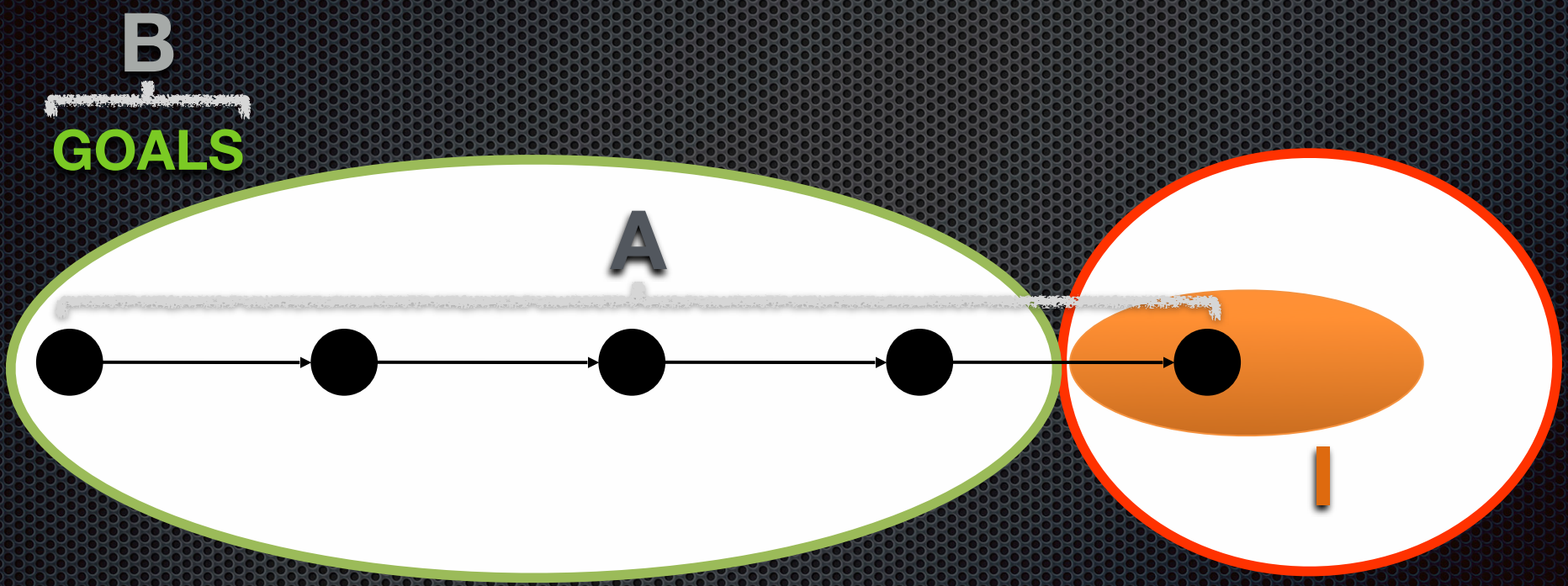
Requirements Refinement

- Our setting: **Concrete counterexamples**



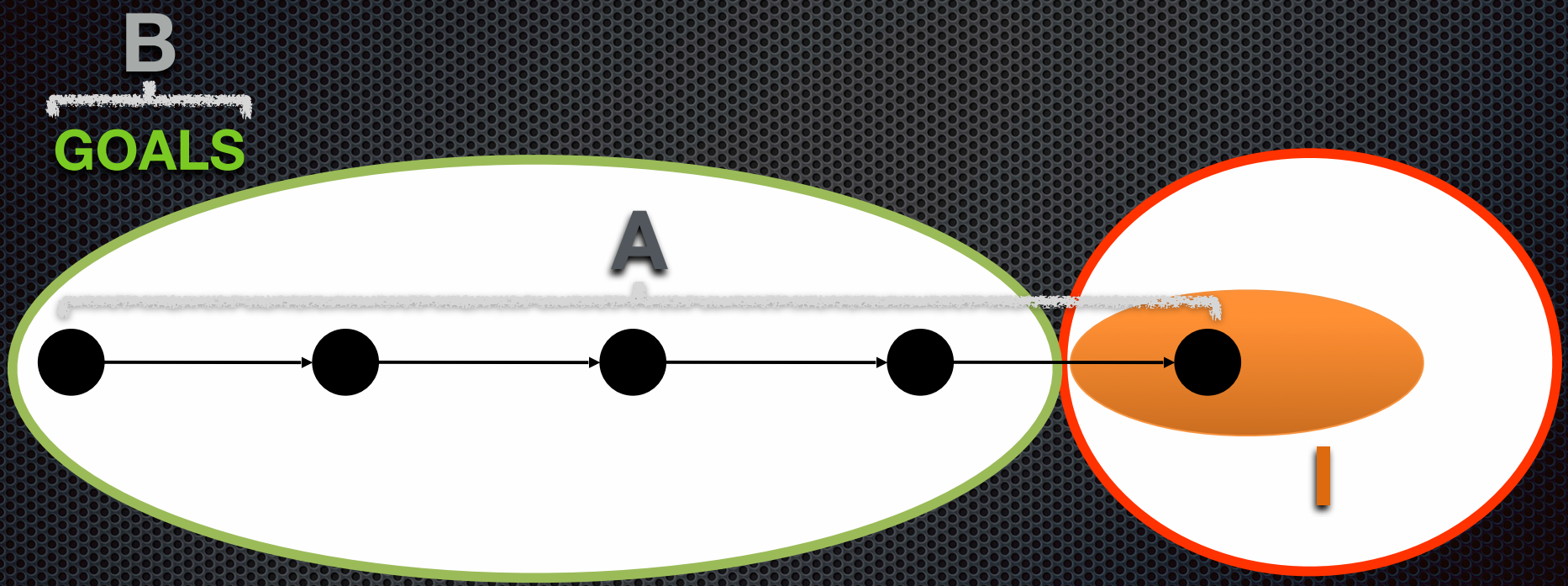
Requirements Refinement

- Our setting: **Concrete counterexamples**



Requirements Refinement

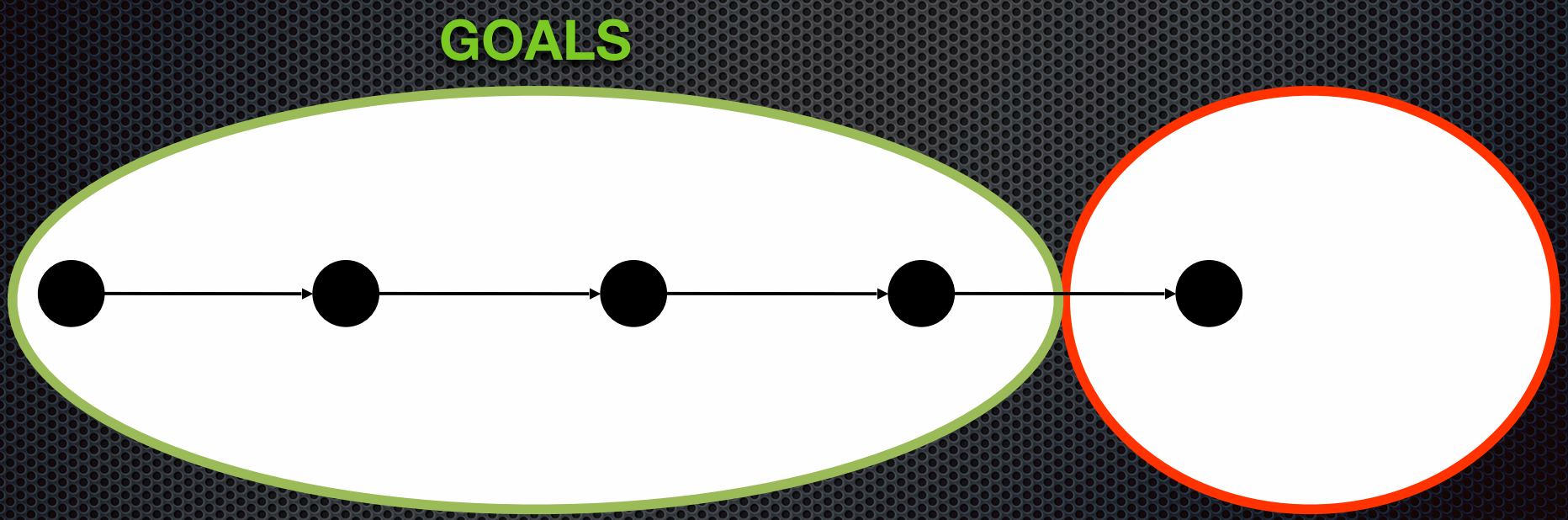
- Our setting: **Concrete counterexamples**



How can we exploit the information given by the interpolant?

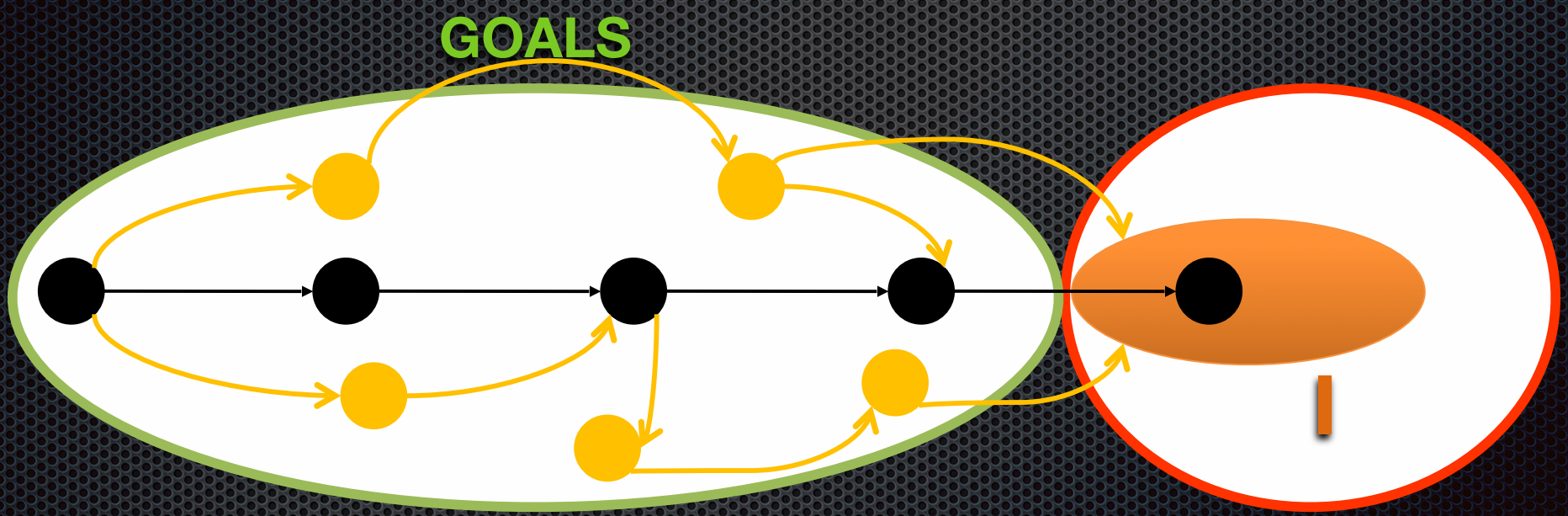
Requirements Refinement

$Counterexample \Rightarrow I$ $UNSAT(I, GOALS)$



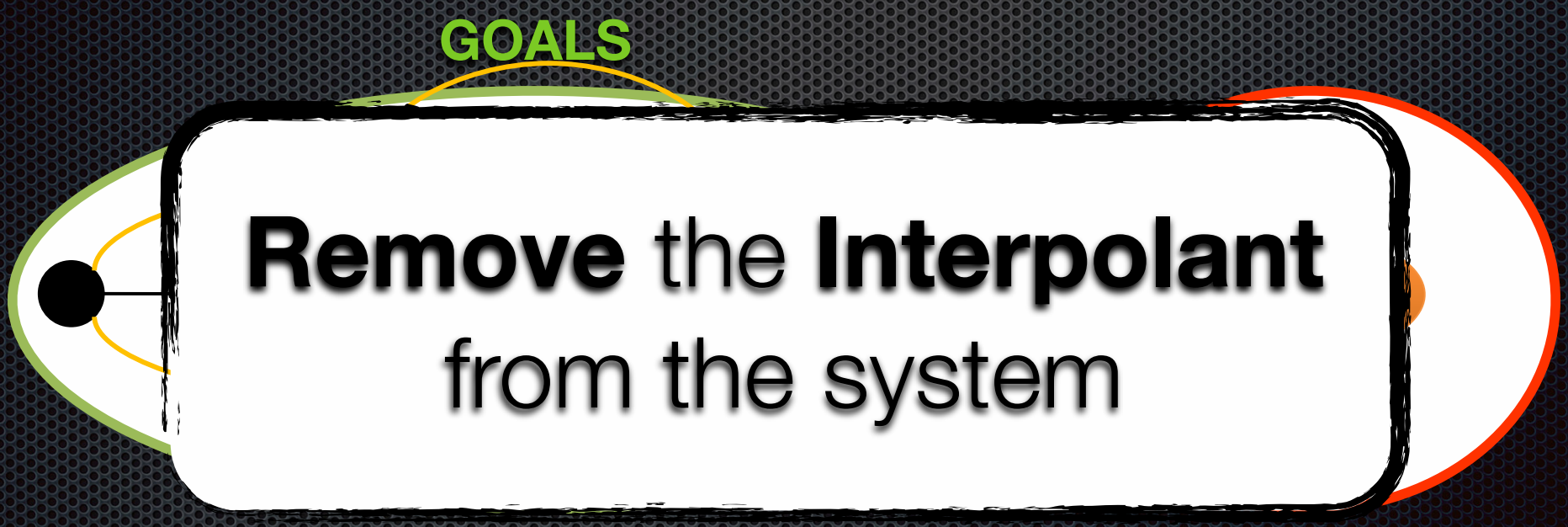
Requirements Refinement

$Counterexample \Rightarrow I$ $UNSAT(I, GOALS)$



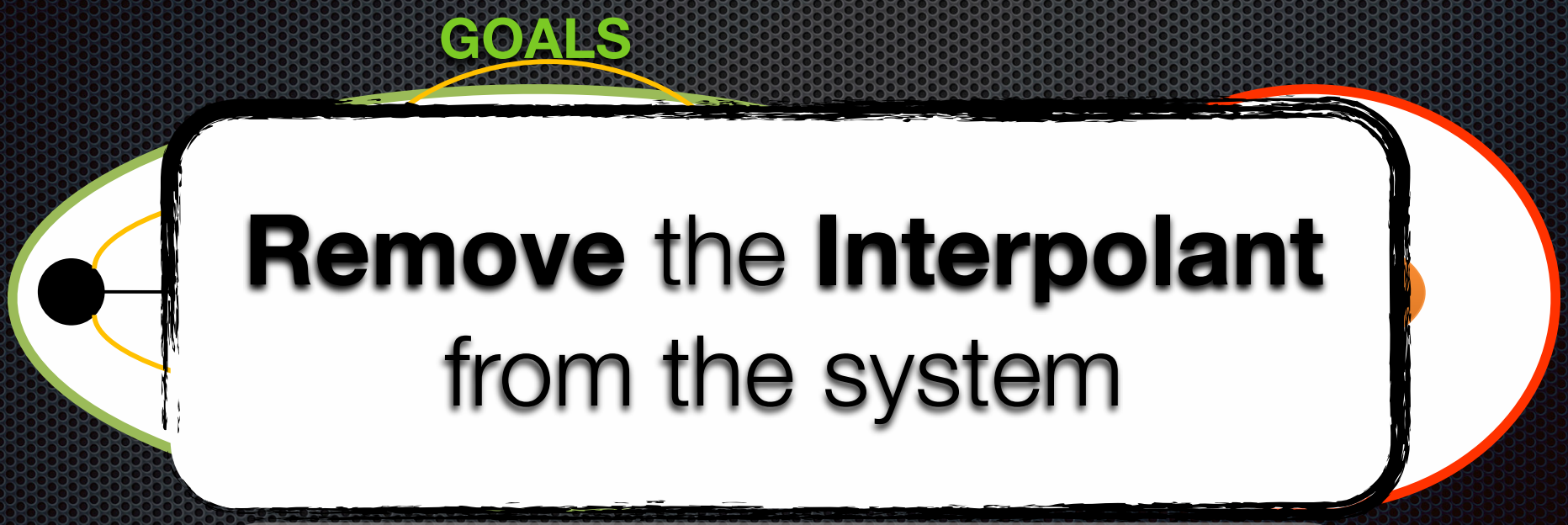
Requirements Refinement

$Counterexample \Rightarrow I$ $UNSAT(I, GOALS)$



Requirements Refinement

$Counterexample \Rightarrow I$ $UNSAT(I, GOALS)$



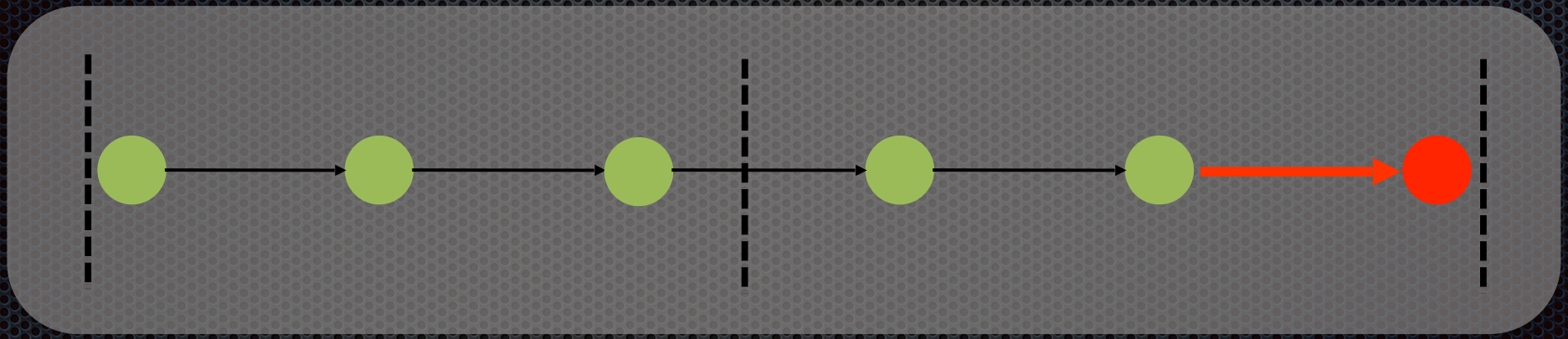
Removing I doesn't necessarily guarantee the satisfaction of the goals

Why counterexamples arise?

first situation

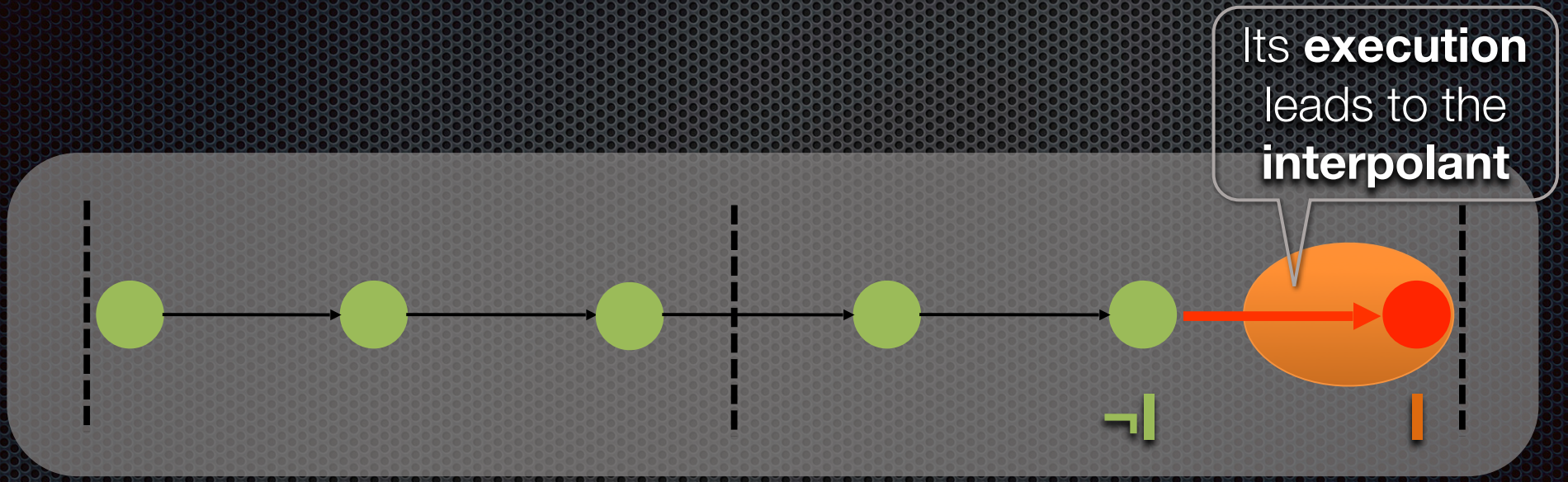
Why counterexamples arise?

first situation



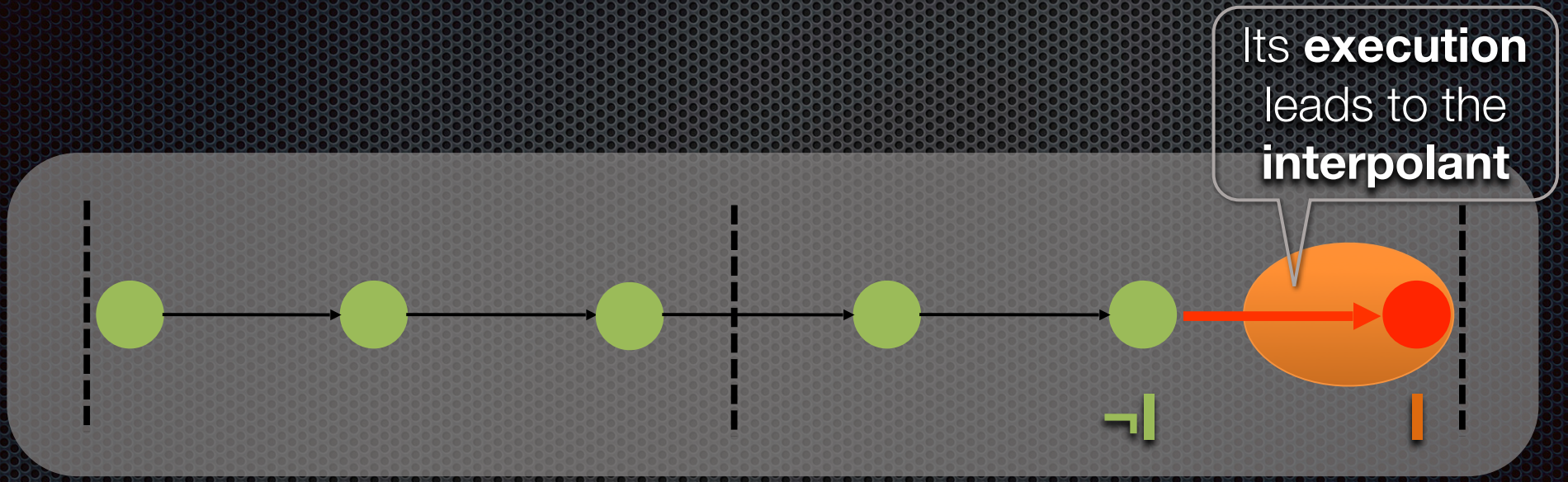
Why counterexamples arise?

first situation



Why counterexamples arise?

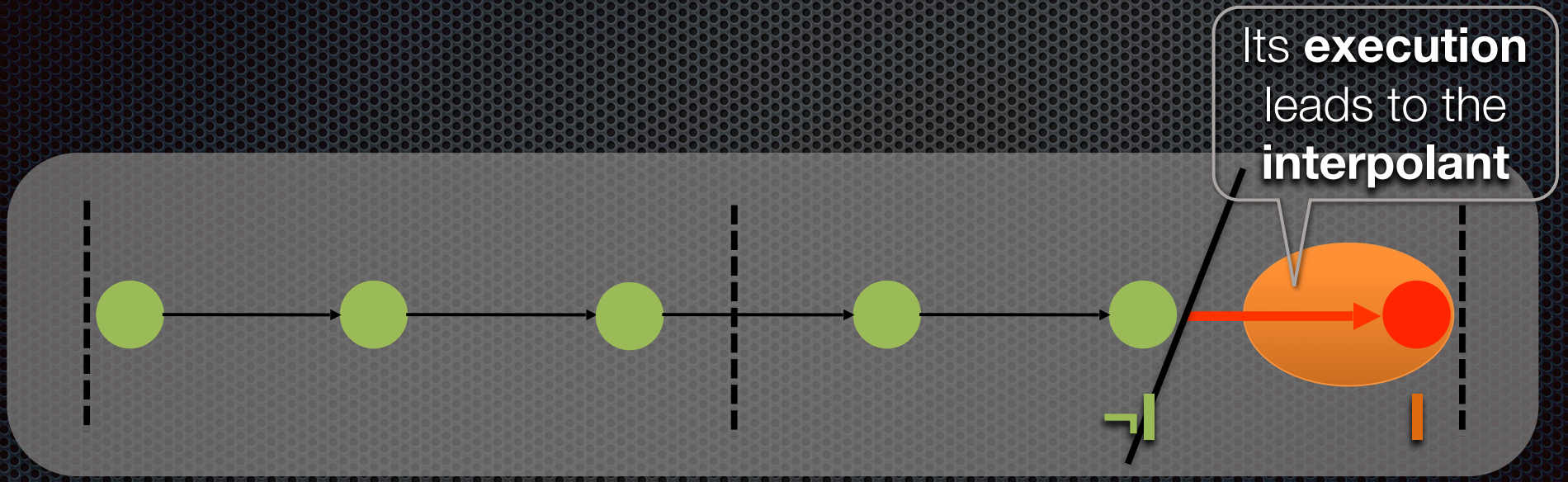
first situation



How can this counterexample **be removed**?

Why counterexamples arise?

first situation



How can this counterexample **be removed**?

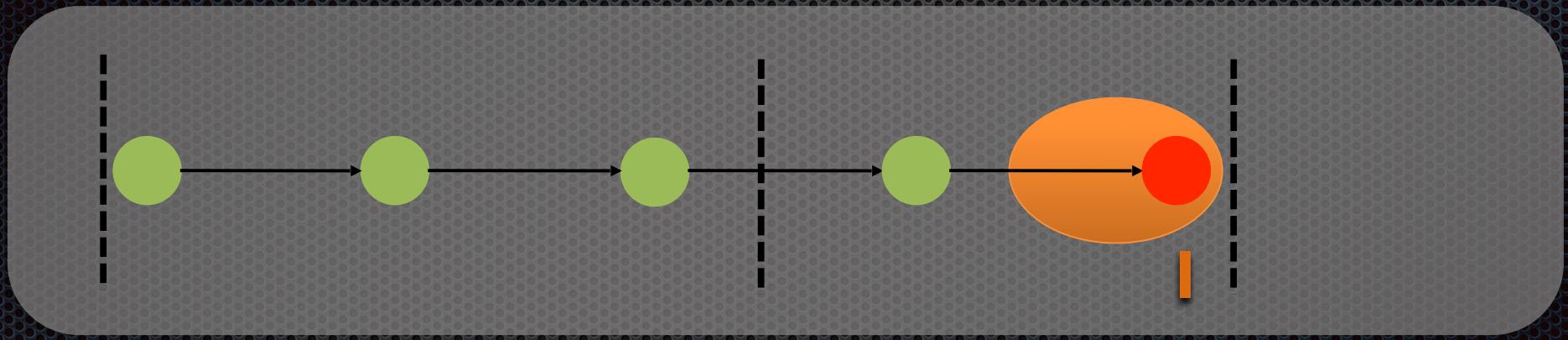
- prohibiting this operation to occur in certain states.
  - **strengthening** its **precondition**.

Why counterexamples arise?

second situation

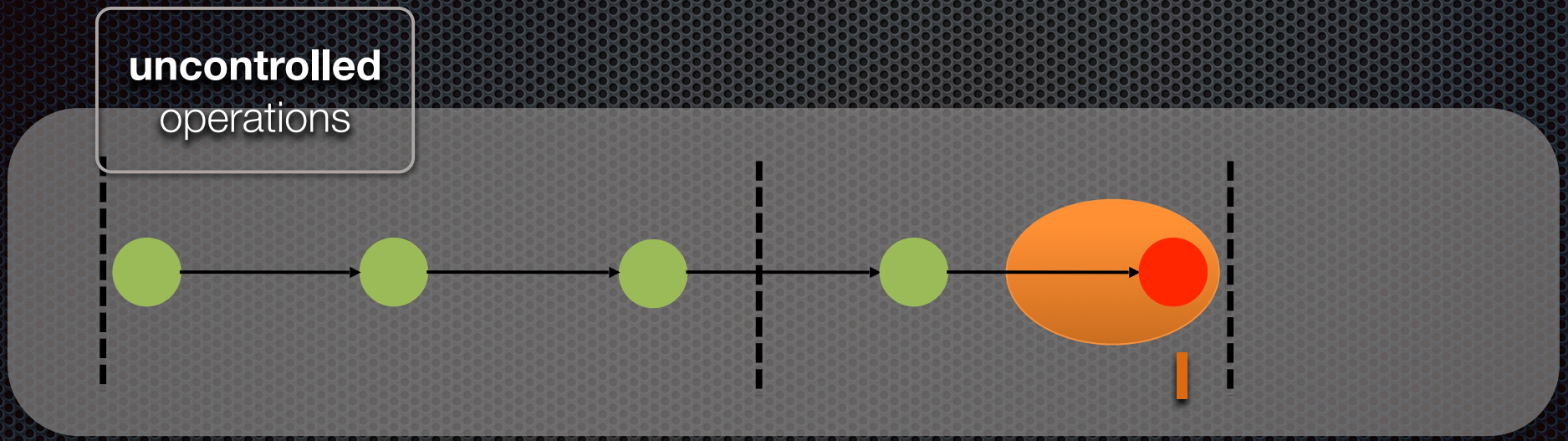
Why counterexamples arise?

second situation



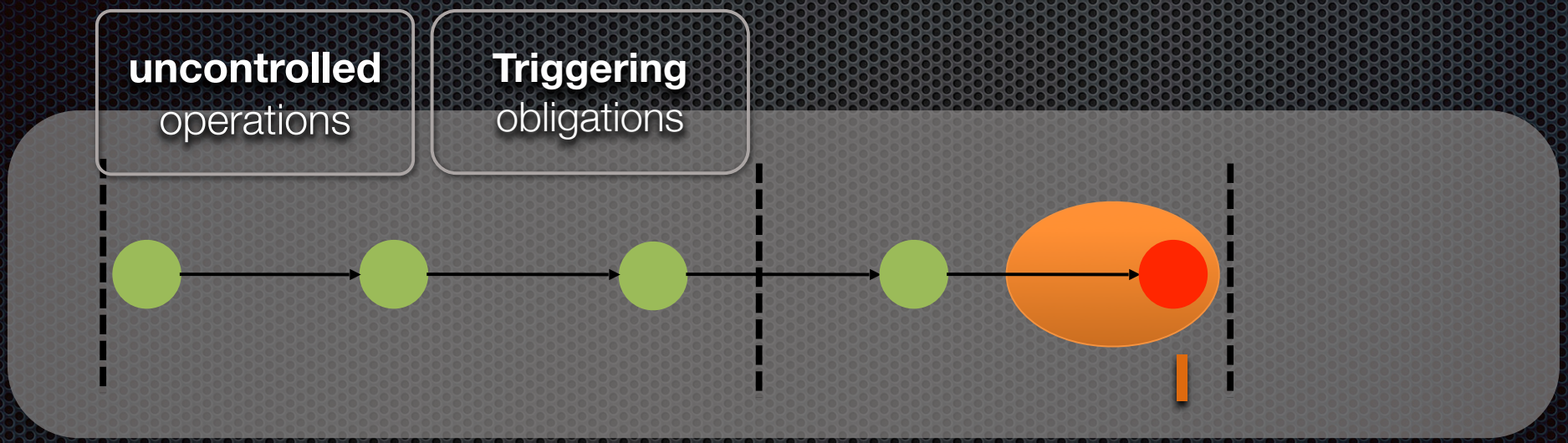
Why counterexamples arise?

second situation



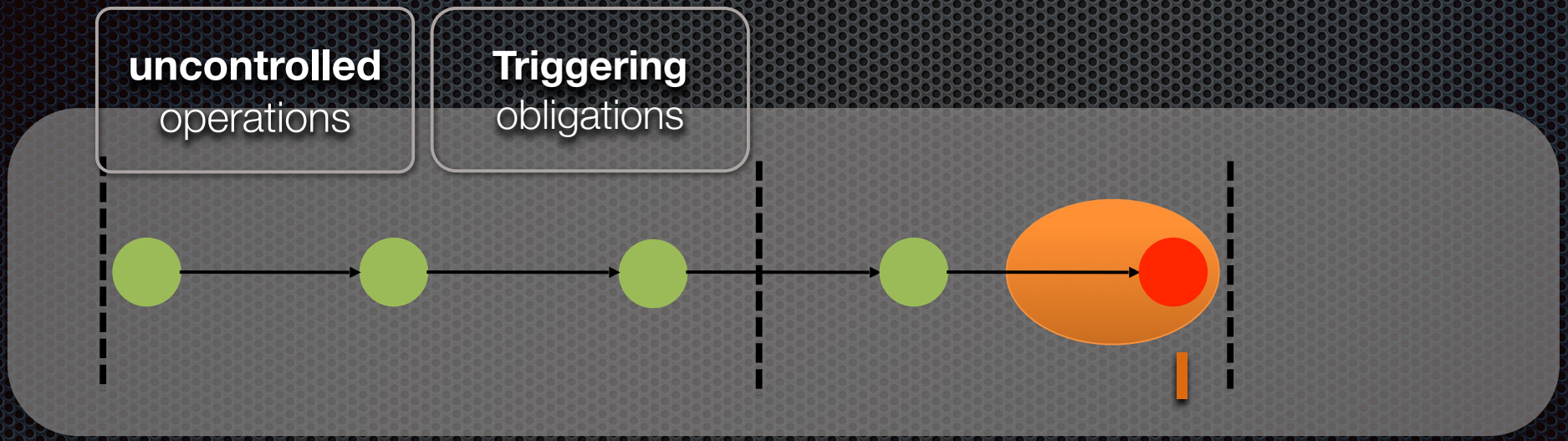
Why counterexamples arise?

second situation



Why counterexamples arise?

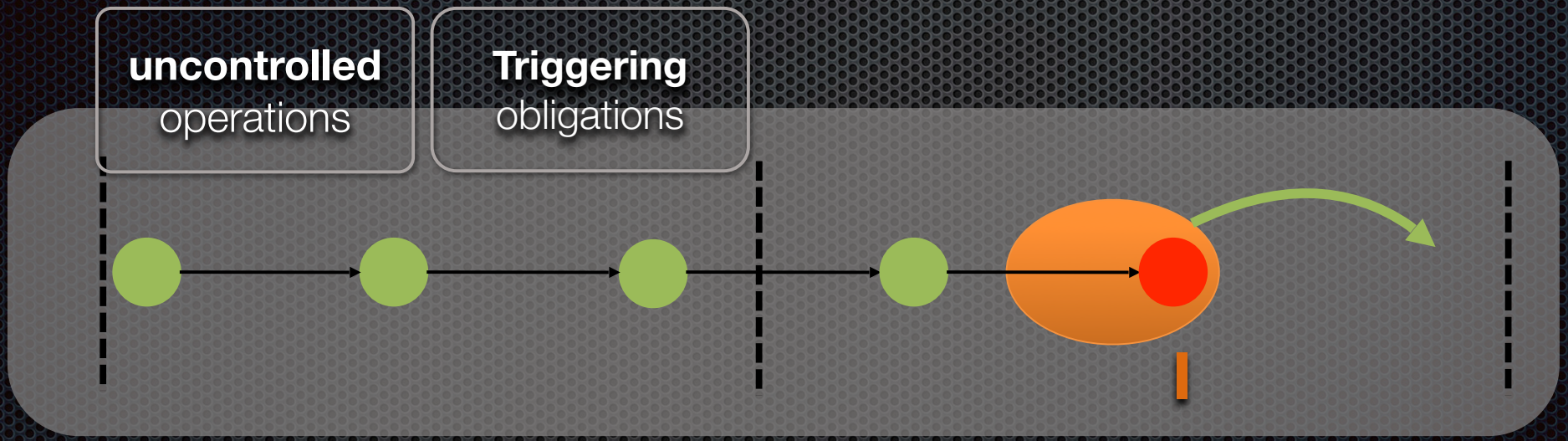
second situation



How can this counterexample **be removed**?

Why counterexamples arise?

second situation

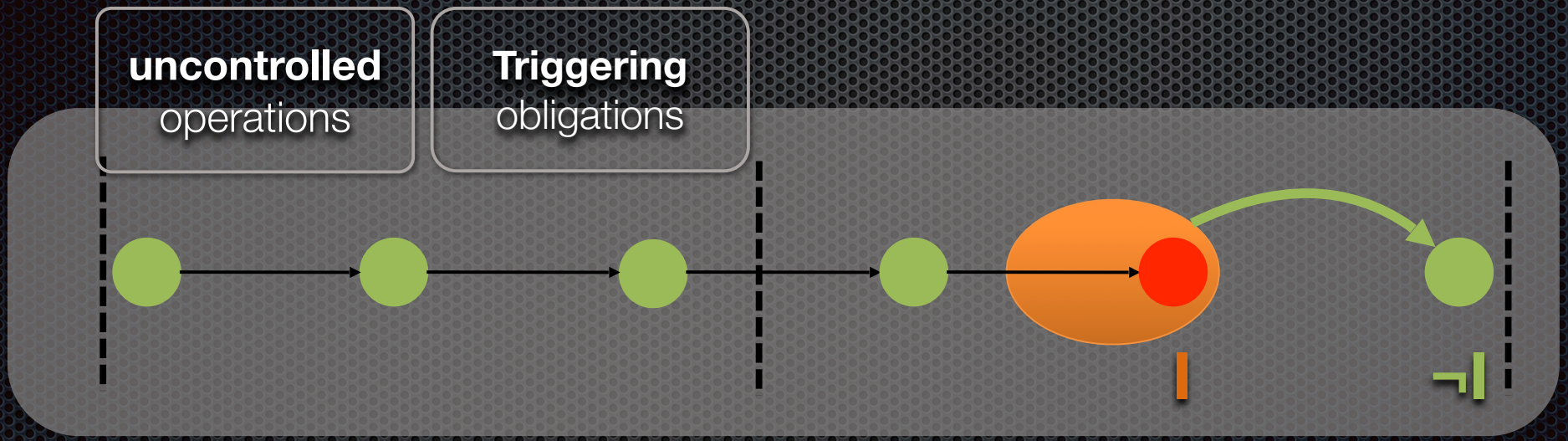


How can this counterexample **be removed**?

- forcing an operation to occur in certain states.

Why counterexamples arise?

second situation

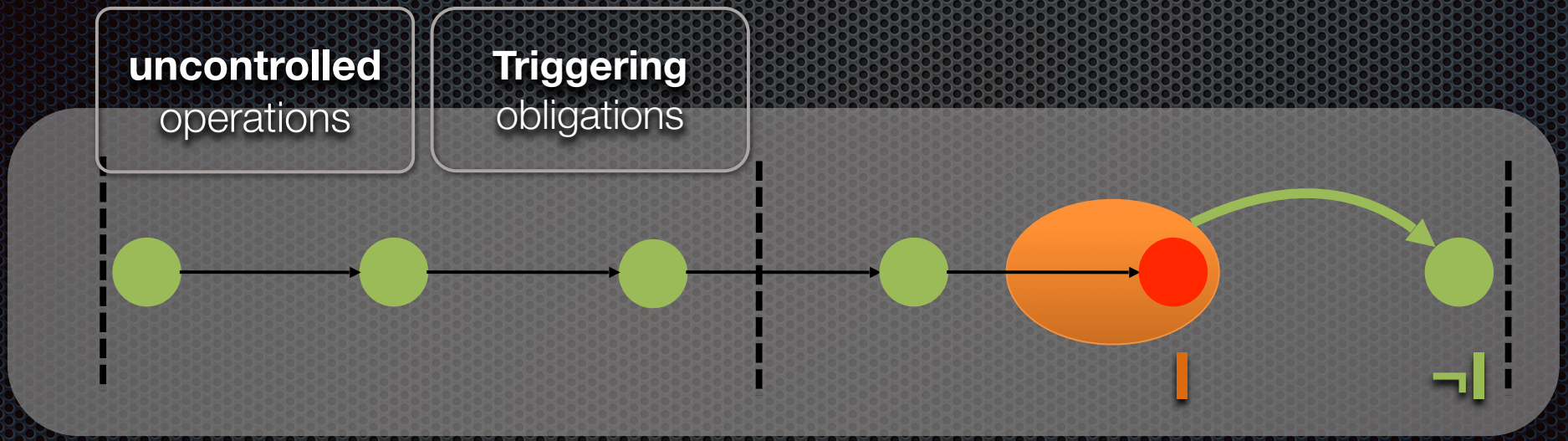


How can this counterexample **be removed**?

- forcing an operation to occur in certain states.

Why counterexamples arise?

second situation



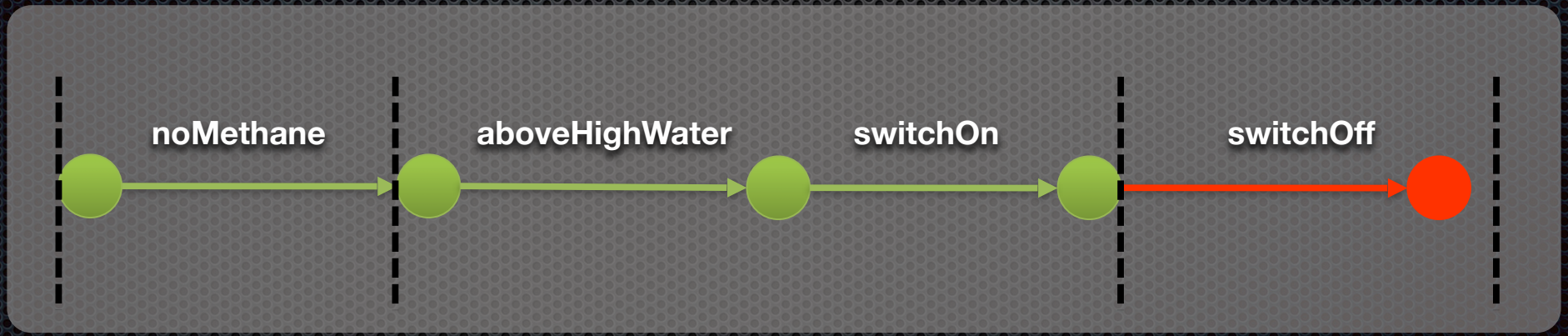
How can this counterexample **be removed**?

- forcing an operation to occur in certain states.
  - **weakening** its **triggering** condition.

Strengthening Preconditions

Pump Controller

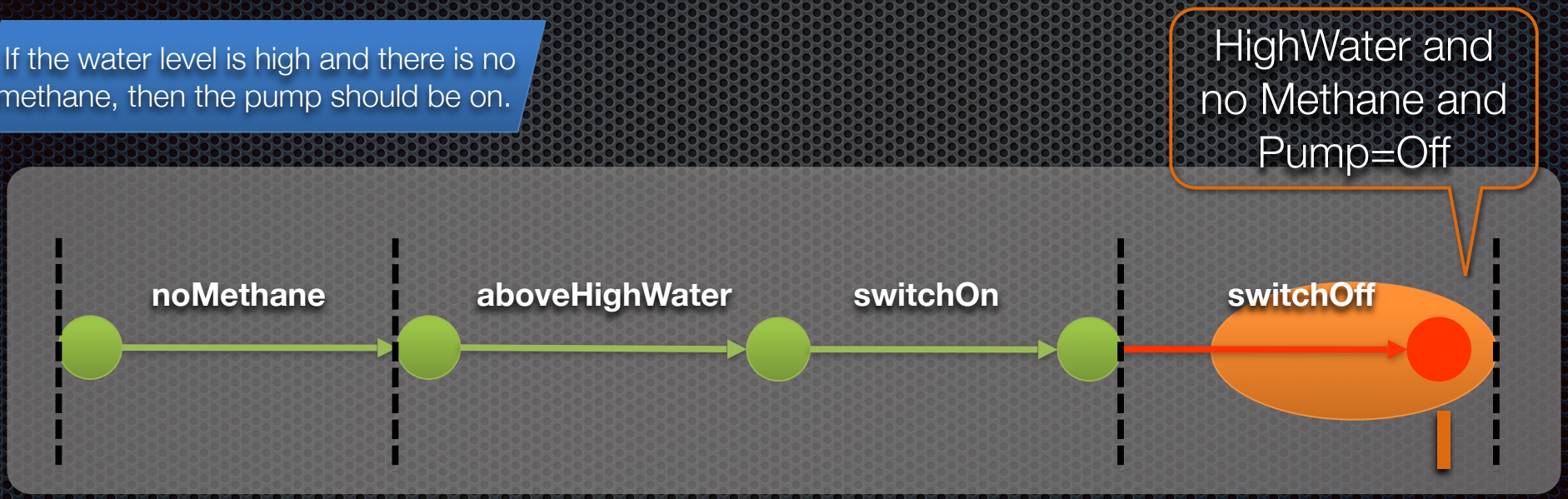
If the water level is high and there is no methane, then the pump should be on.



Strengthening Preconditions

Pump Controller

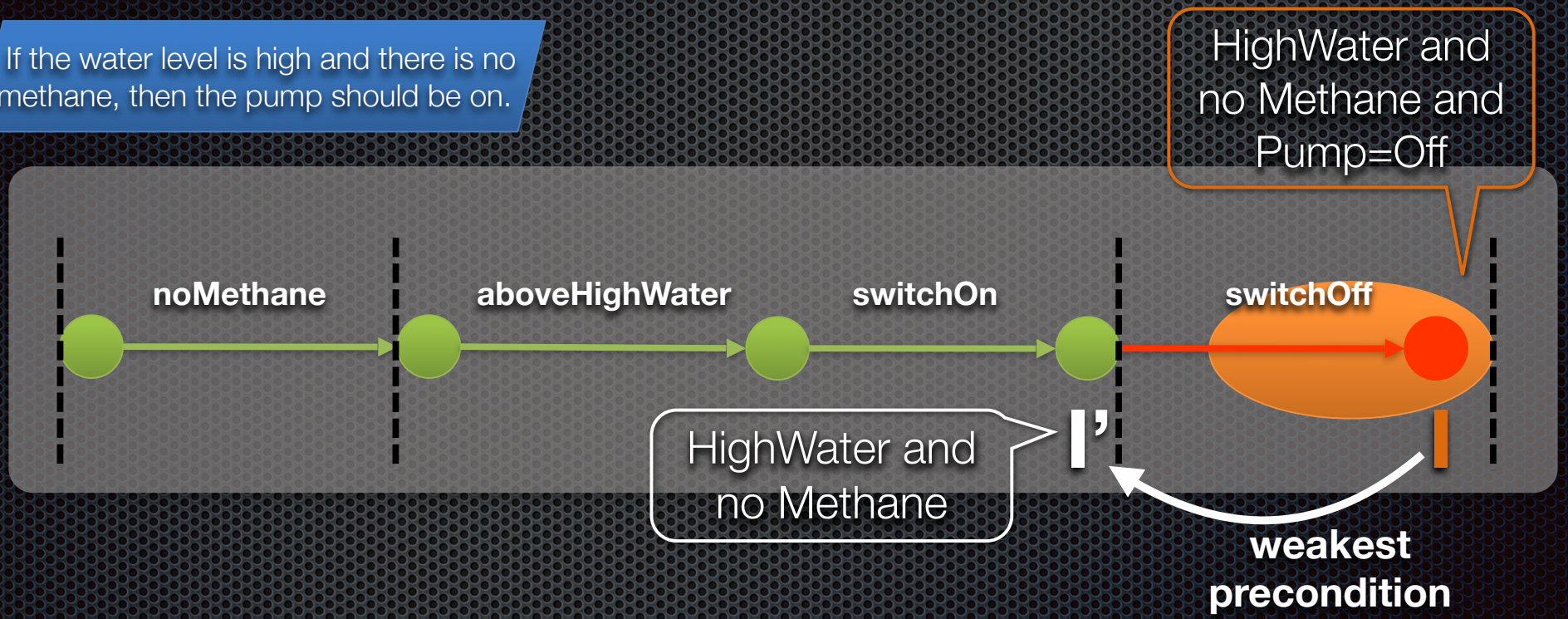
If the water level is high and there is no methane, then the pump should be on.



Strengthening Preconditions

Pump Controller

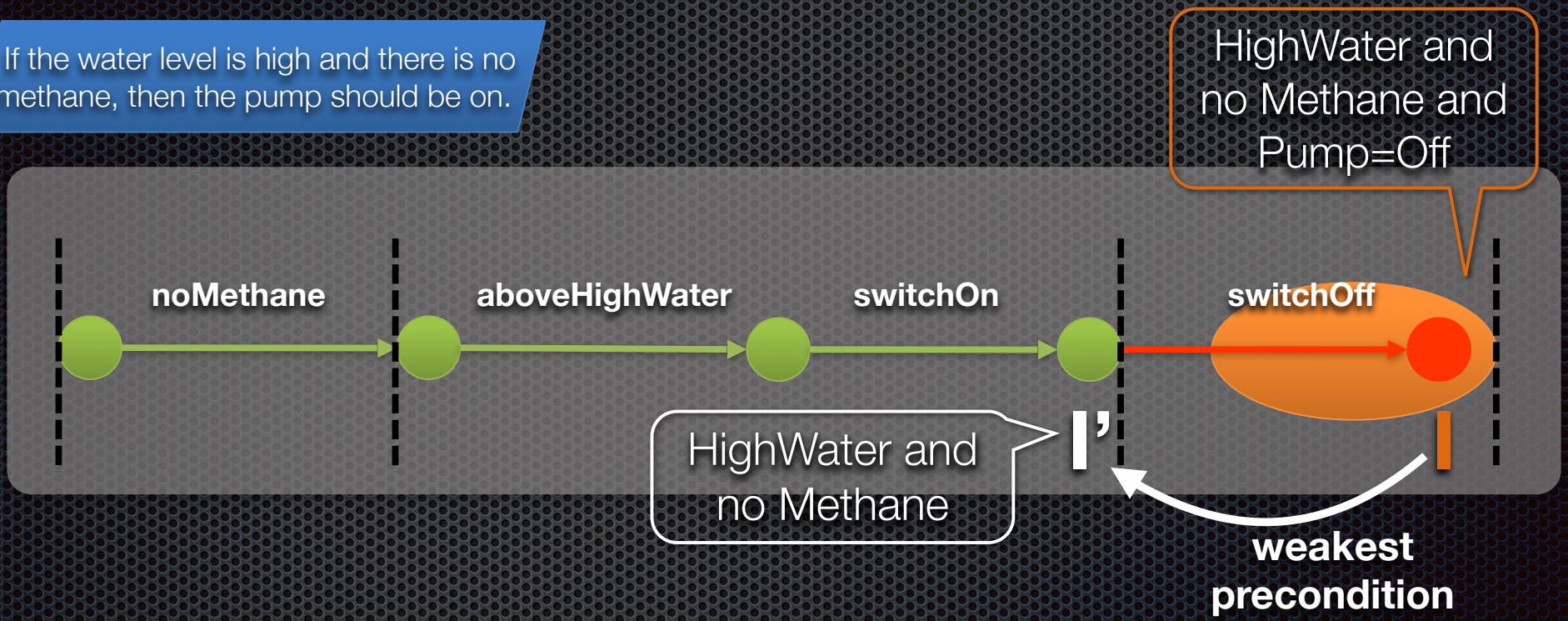
If the water level is high and there is no methane, then the pump should be on.



Strengthening Preconditions

Pump Controller

If the water level is high and there is no methane, then the pump should be on.

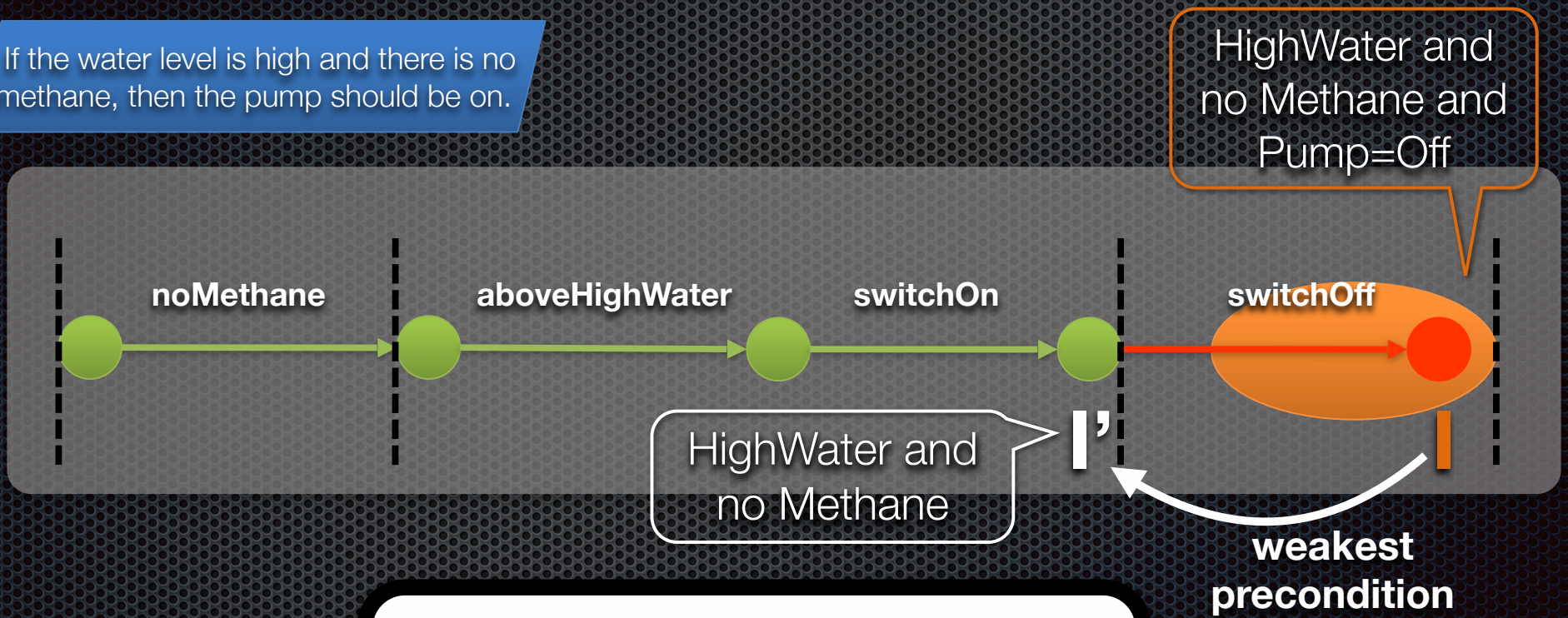


Prohibit **switchOff** from occuring when **I'** holds.

Strengthening Preconditions

Pump Controller

If the water level is high and there is no methane, then the pump should be on.



Prohibit s

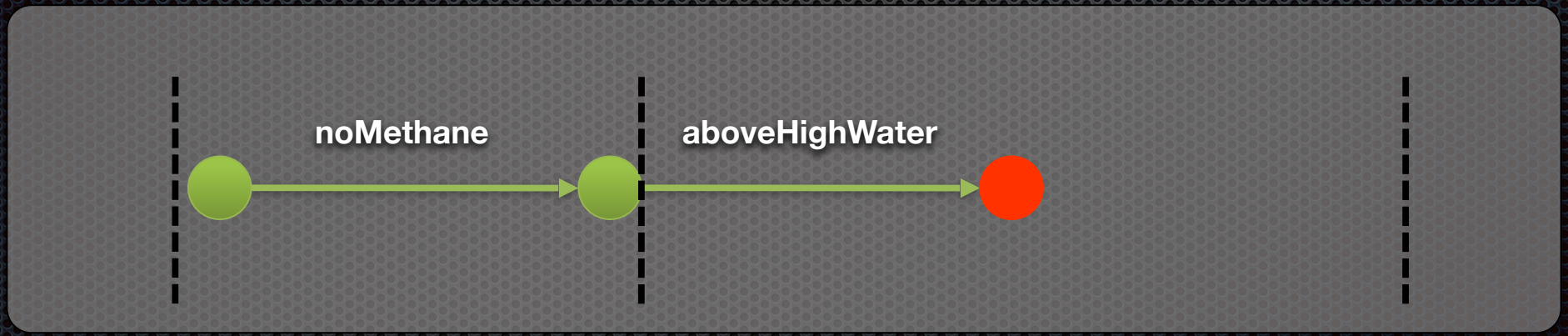
Pre(switchOff) ≡ (Pump=On) **and ¬I'**

strengthening switchOff **precondition**

Weakening Triggering conditions

Pump Controller

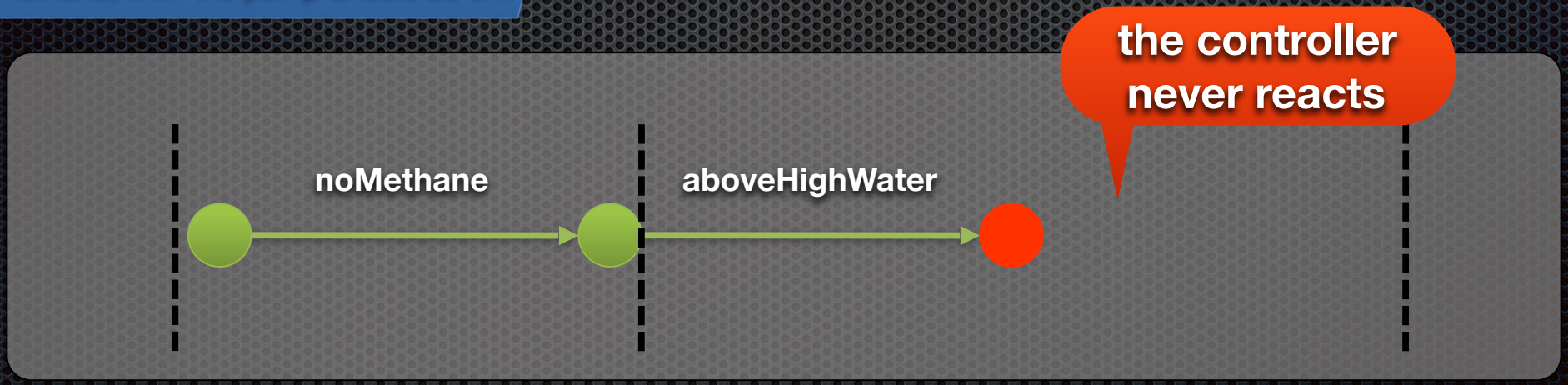
If the water level is high and there is no methane, then the pump should be on.



Weakening Triggering conditions

Pump Controller

If the water level is high and there is no methane, then the pump should be on.

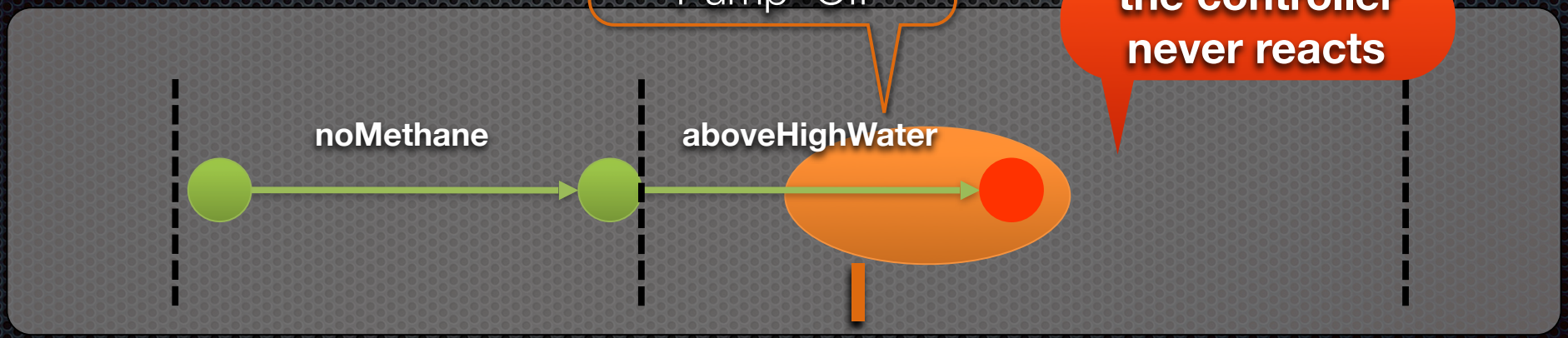


Weakening Triggering conditions

Pump Controller

If the water level is high and there is no methane, then the pump should be on.

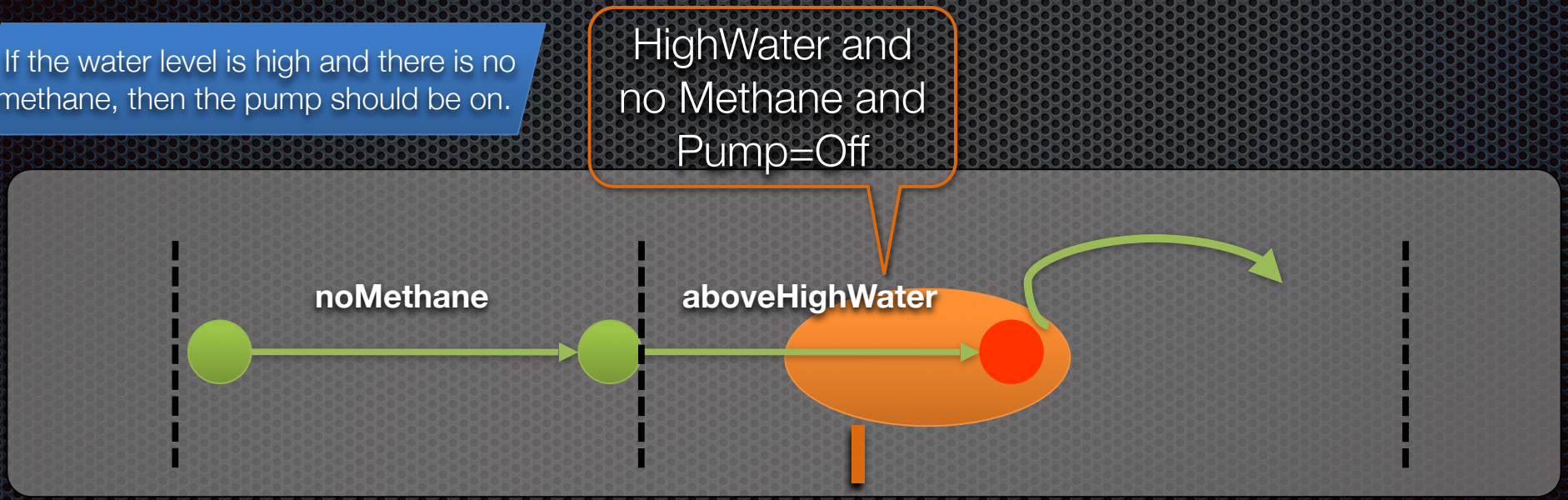
HighWater and no Methane and Pump=Off



Weakening Triggering conditions

Pump Controller

If the water level is high and there is no methane, then the pump should be on.

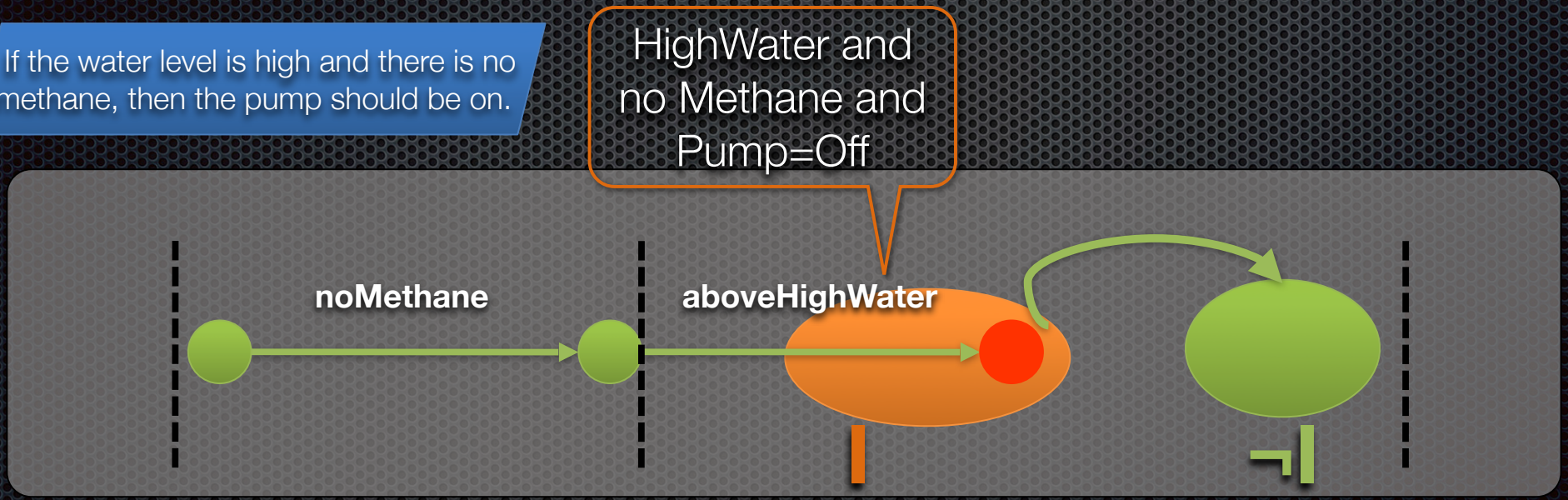


- Force an operation to occur when **I** holds,

Weakening Triggering conditions

Pump Controller

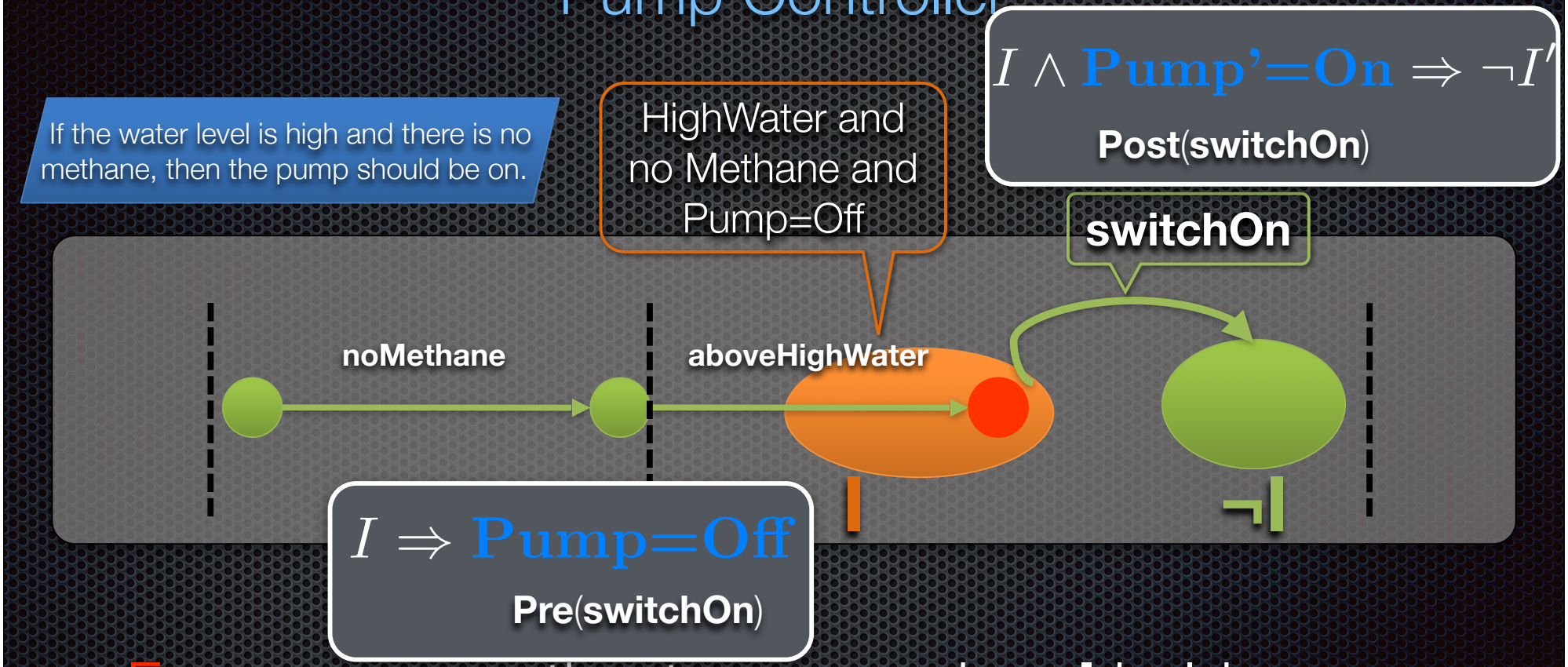
If the water level is high and there is no methane, then the pump should be on.



- Force an operation to occur when **I** holds,
- and, its execution should not reach **I**.

Weakening Triggering conditions

Pump Controller



- Force an operation to occur when **I** holds,
- and, its execution should not reach **I**.

Weakening Triggering conditions

Pump Controller

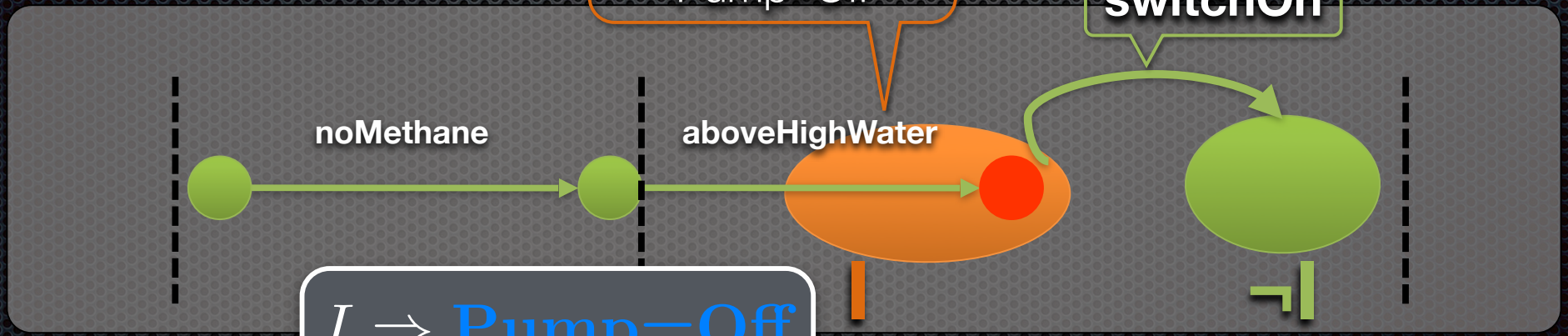
If the water level is high and there is no methane, then the pump should be on.

HighWater and no Methane and Pump=Off

$I \wedge \mathbf{Pump'{=}On} \Rightarrow \neg I'$

Post(switchOn)

switchOn



$I \Rightarrow \mathbf{Pump{=}Off}$

Pre(switchOn)

- Force an c
- and, its ex

Trig(switchOn) ≡ false or I

weakening switchOn **Trig. condition**

Liveness

Reactivity Pattern

Liveness

Reactivity Pattern

Liveness

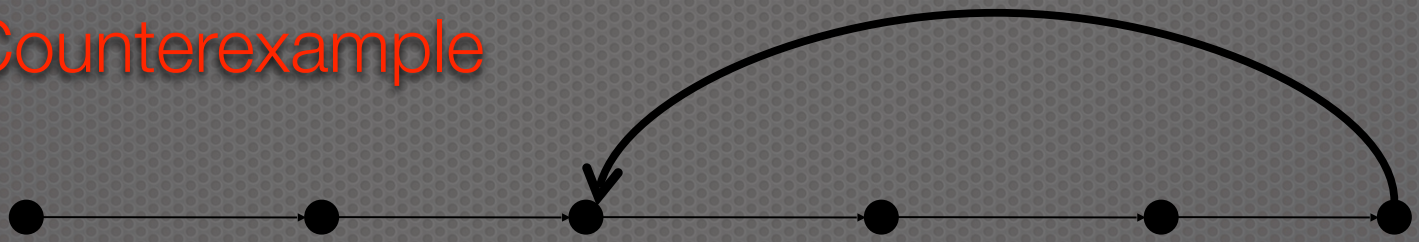
Reactivity Pattern

Infinitely often

$\square\Diamond$ **Assumptions** $\Longrightarrow$ $\square\Diamond$ **Goals**

Infinitely often

Counterexample



Liveness

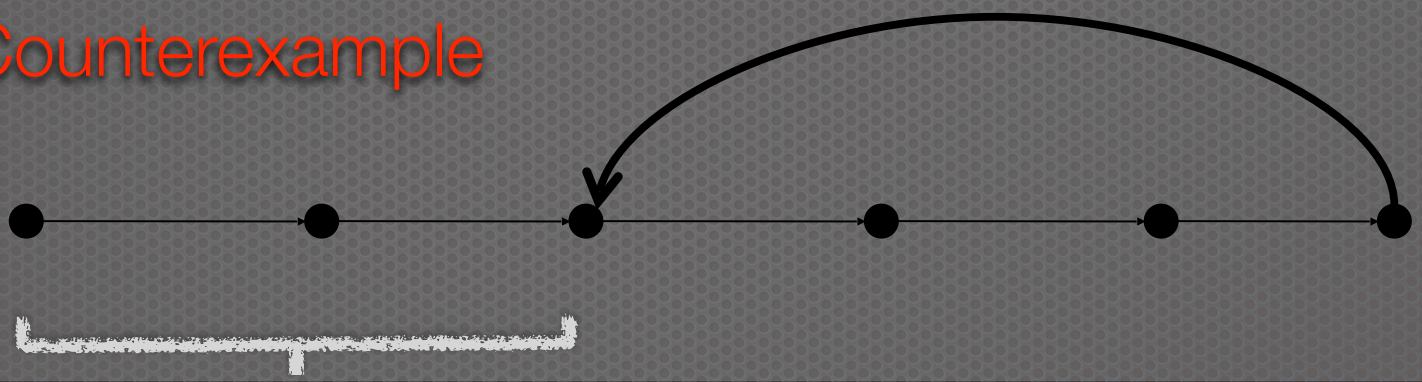
Reactivity Pattern

Infinitely often

$\square\Diamond$ **Assumptions** $\Rightarrow$ $\square\Diamond$ **Goals**

Infinitely often

Counterexample



prefix

Liveness

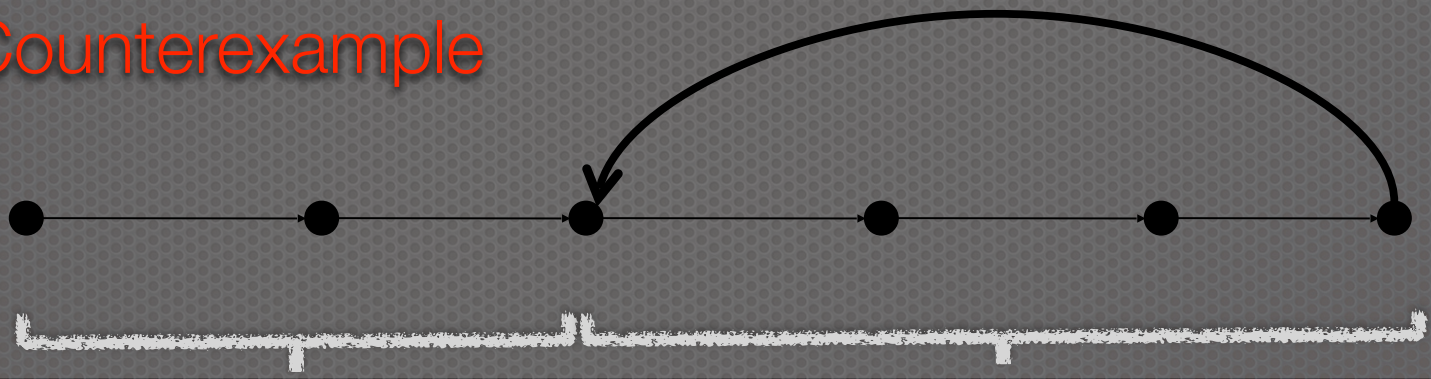
Reactivity Pattern

Infinitely often

□◇ **Assumptions** ⟹ □◇ **Goals**

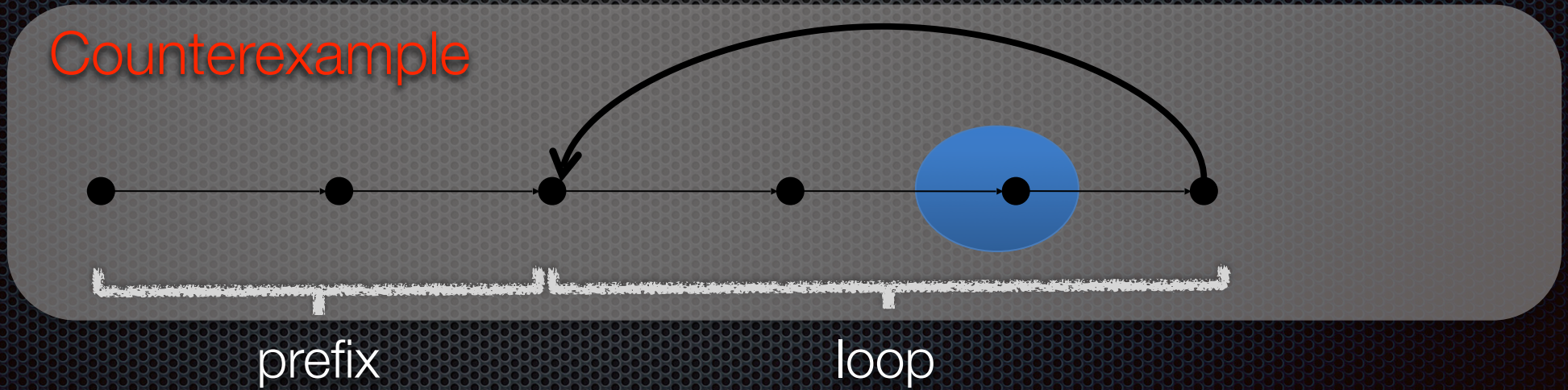
Infinitely often

Counterexample



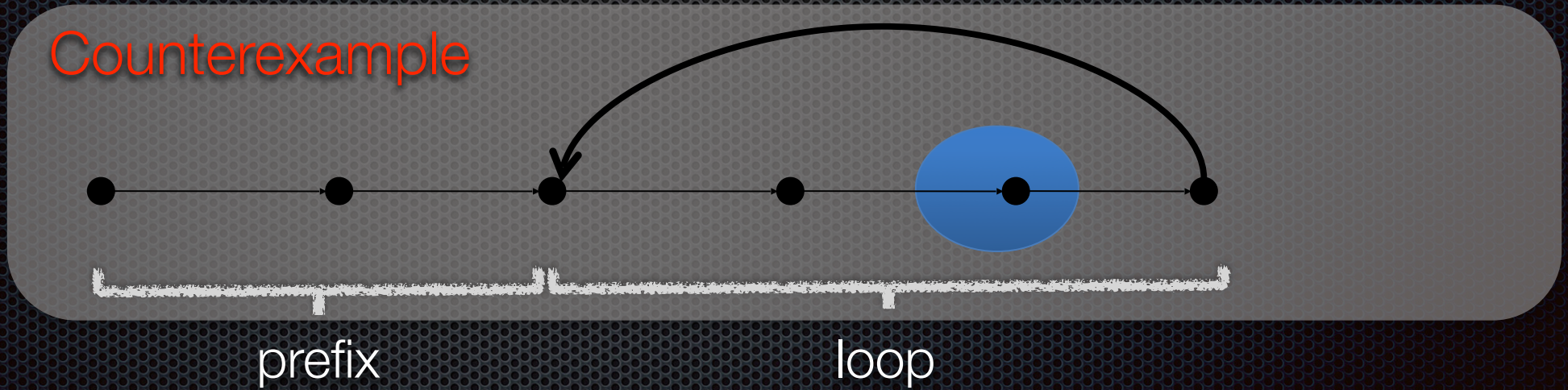
Liveness

Reactivity Pattern



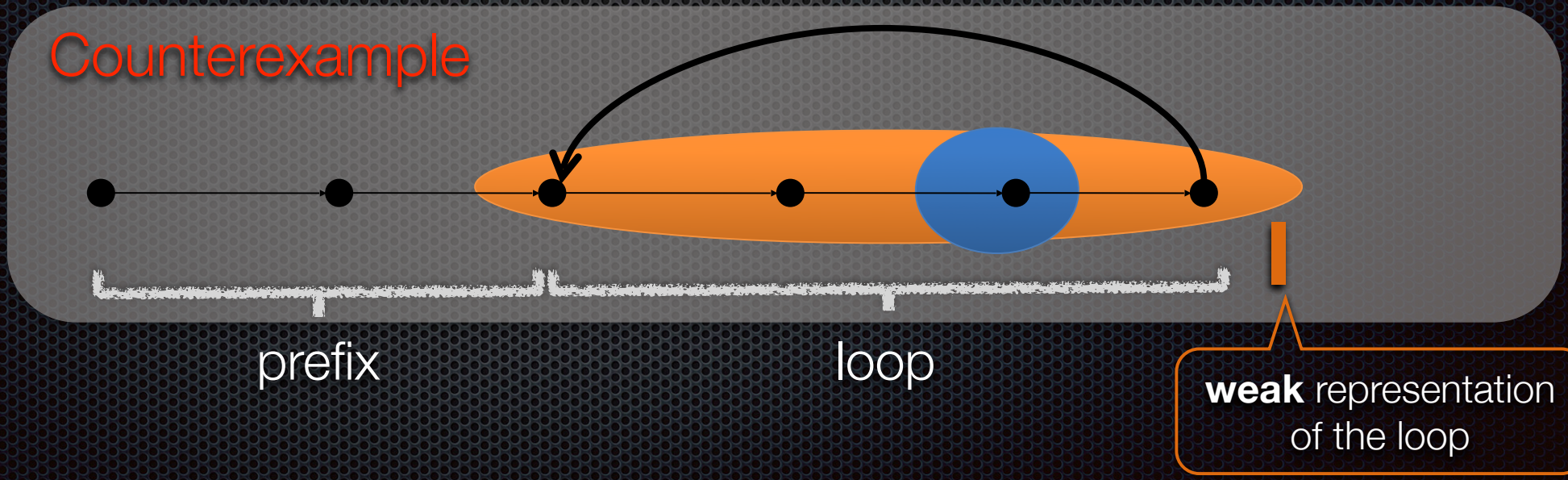
Liveness

Reactivity Pattern



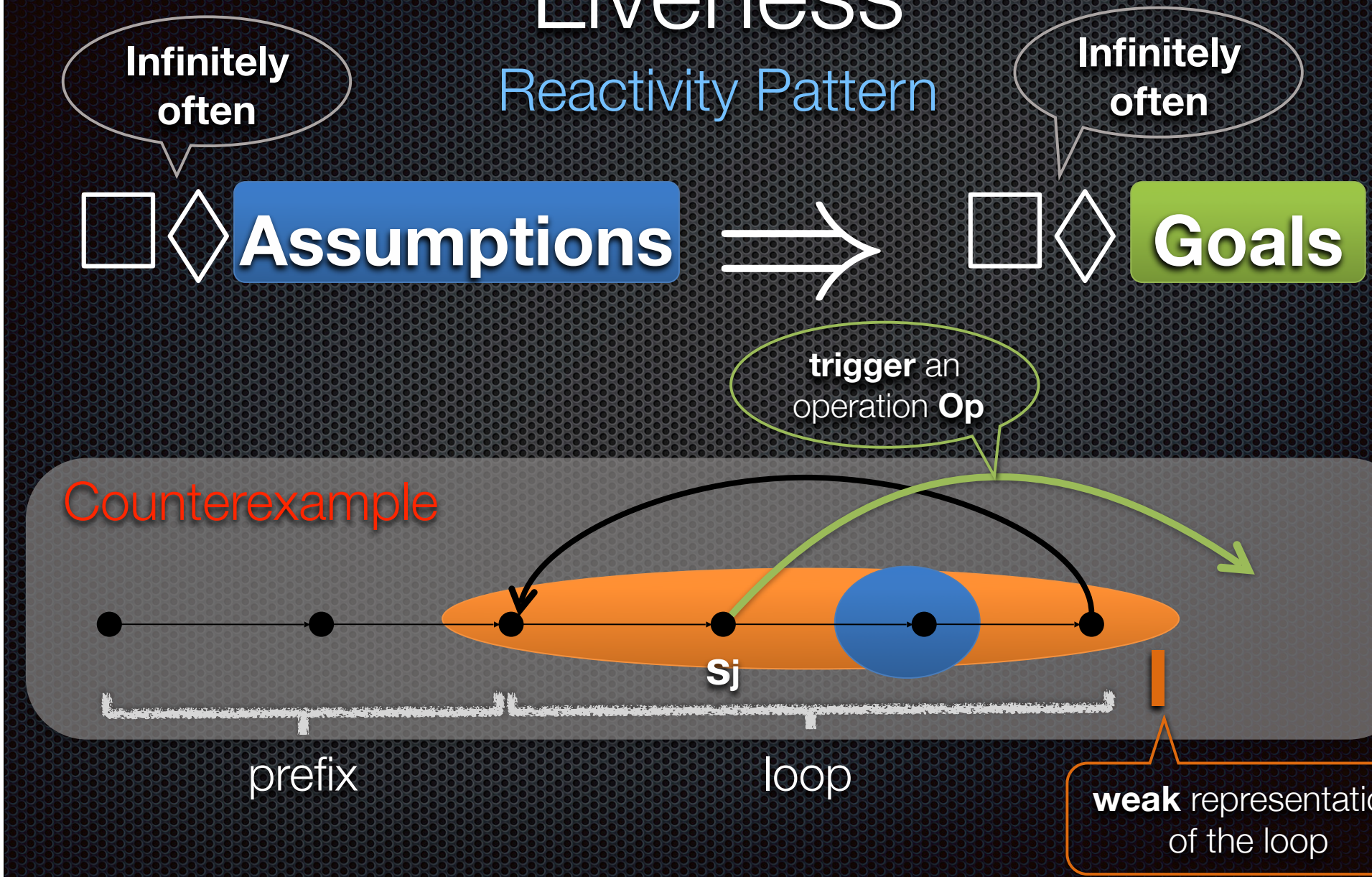
Liveness

Reactivity Pattern



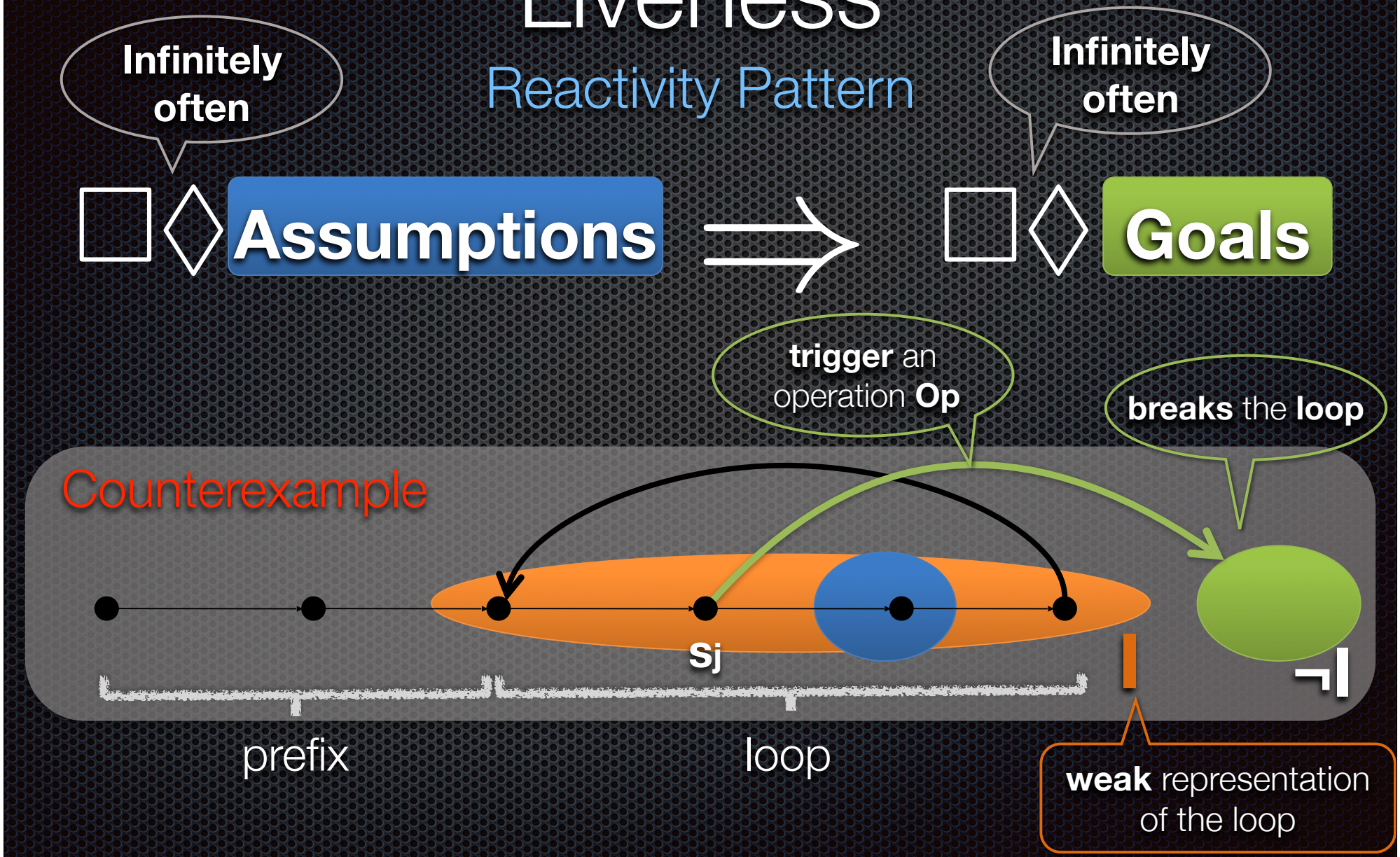
Liveness

Reactivity Pattern



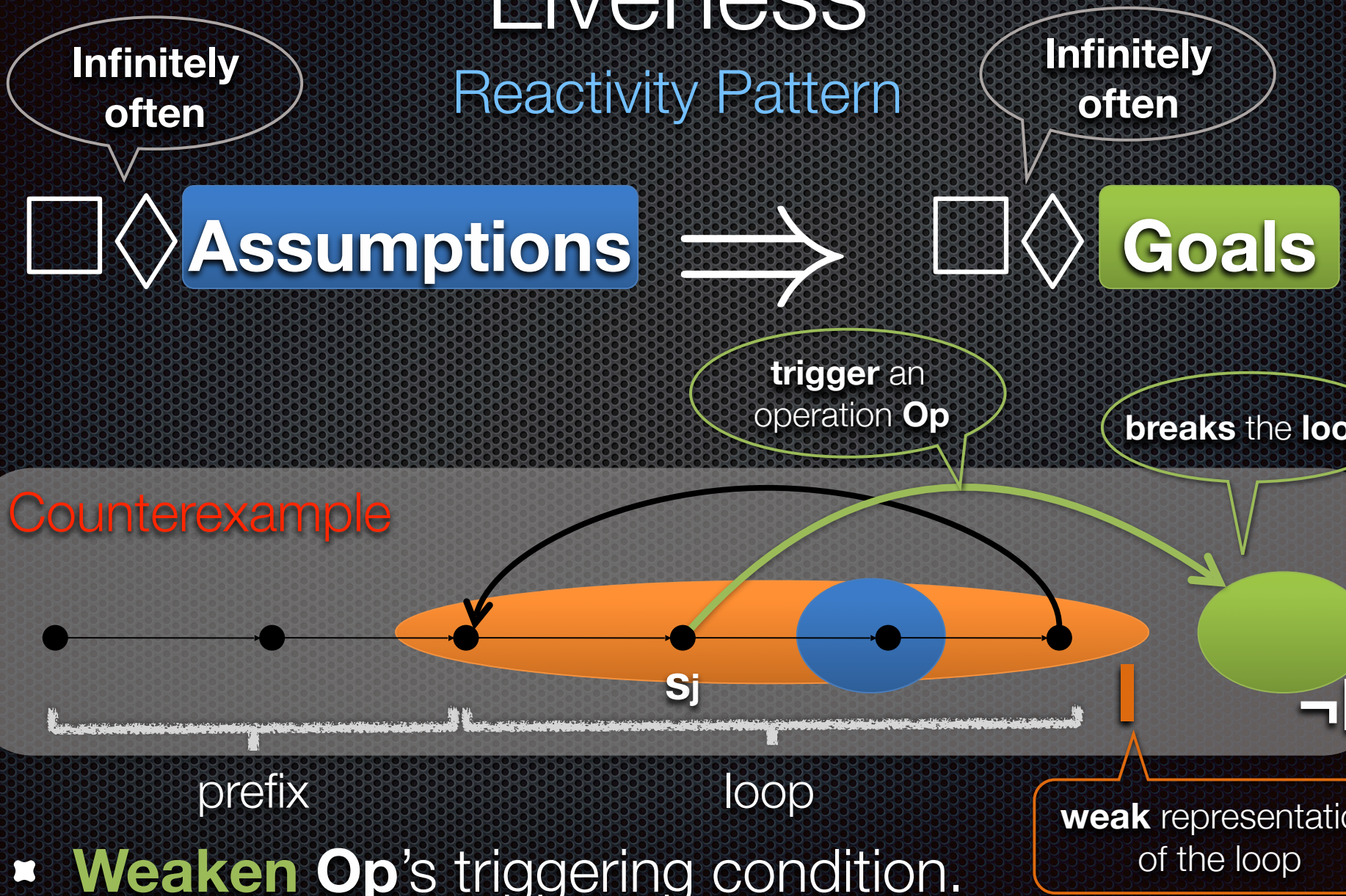
Liveness

Reactivity Pattern



Liveness

Reactivity Pattern



- **Weaken Op**'s triggering condition.

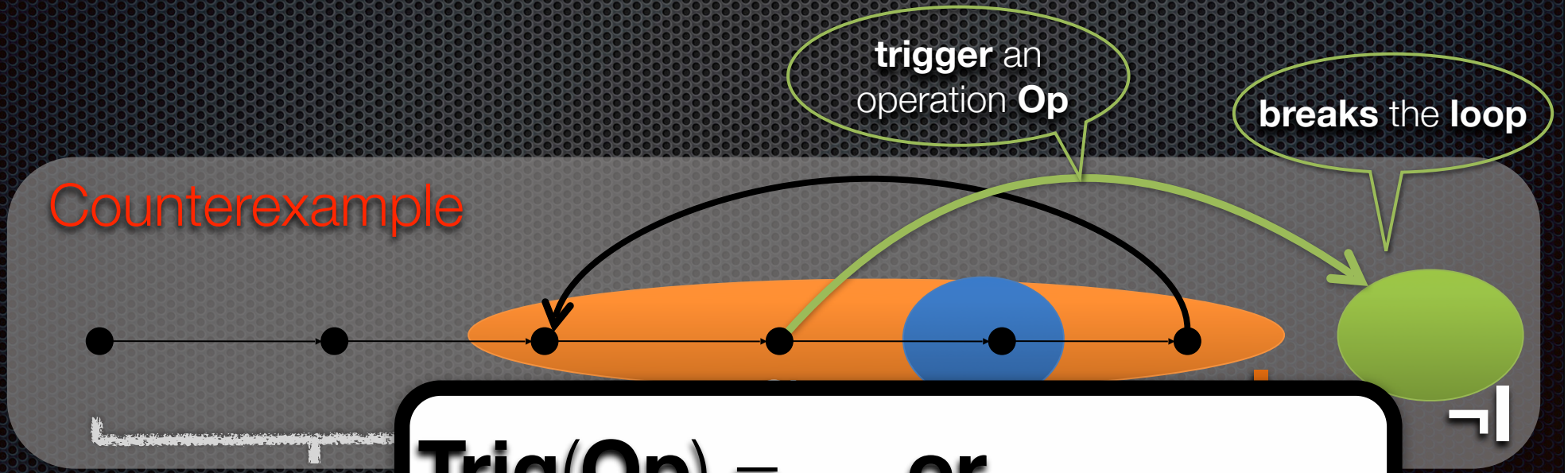
Liveness

Reactivity Pattern

Infinitely often

□◇ **Assumptions** ⟹ □◇ **Goals**

Infinitely often



Trig(Op) = … or
(Pre(Op) and no Goals)

- **Weaken**

entation
op

Contributions

- Fully automated Goal Operationalisation:
  - Safety Goals,
  - Time Progress property,
  - **Liveness** properties (Reactivity).
- Successfully evaluated on four demonstrating examples.
  - We obtained less restrictive operationalisations than inductive techniques, in some cases.
  - **Interpolation+SAT** improves inductive techniques **running times**.

Limitations

- Correct, but Incomplete, goal operationalisation process.
  - Successful operationalisation may depend on the order in which counterexamples arise.
- Liveness
  - The refined triggering condition is not based on the interpolant information.
  - We may produce a stronger than necessary refinement,
  - But, it is consistent with the preconditions.

Future Work

- Evaluate **scalability** issues on case studies.
- Investigate the relationship between:
  - **interpolation** and **inductive logic programming**.
- Analyse a potential notion of “most general" operationalisation.
- Evaluate alternative mechanisms for interpolant computation
  - **Quality** of interpolants for our purposes.

Questions?