Specifying Policies Using UML Sequence Diagrams

An Evaluation Based on a Case Study

Bjørnar Solhaug,
Dag Elgesem and Ketil Stølen

2007 IEEE Workshop on Policies for Distributed Systems and Networks, Bologna, 13 June 2007

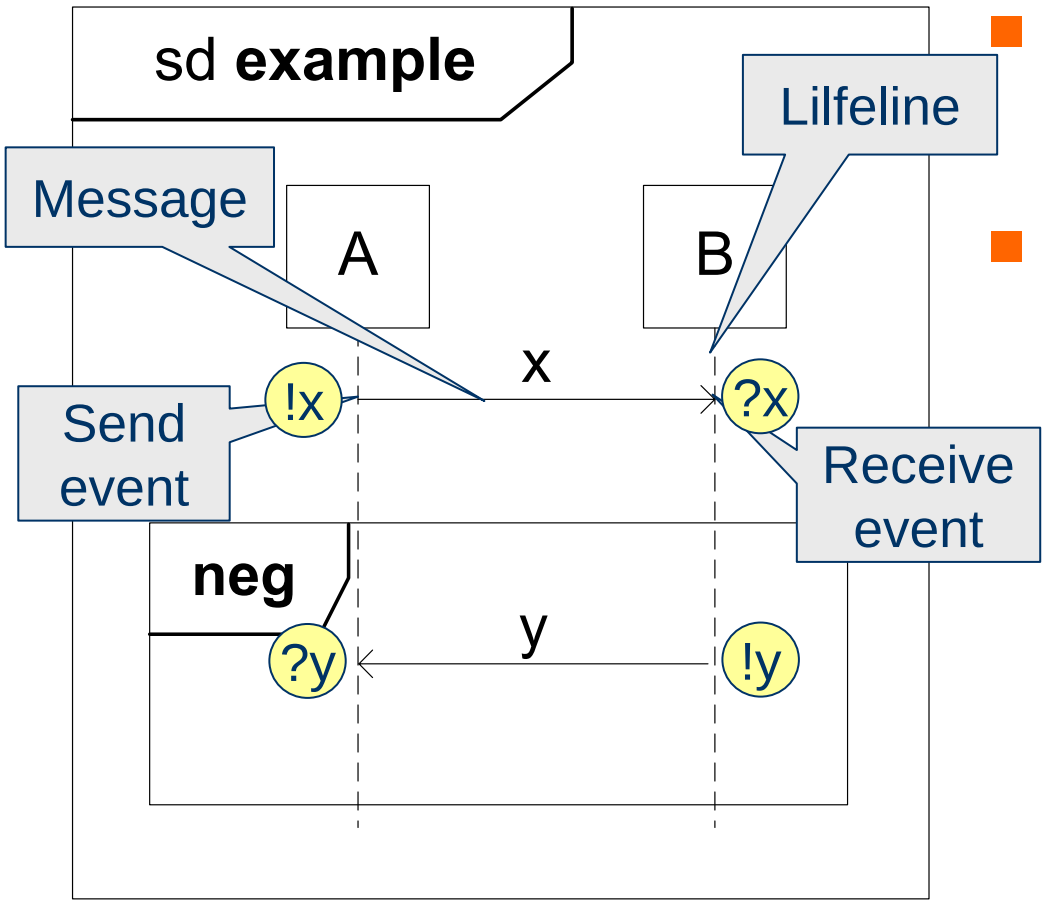
Motivation and Objective

- Policies are increasingly being used for the management of information systems
- The UML is currently the *de facto* standard for the modeling of information system
- Can we specify policies using the UML?

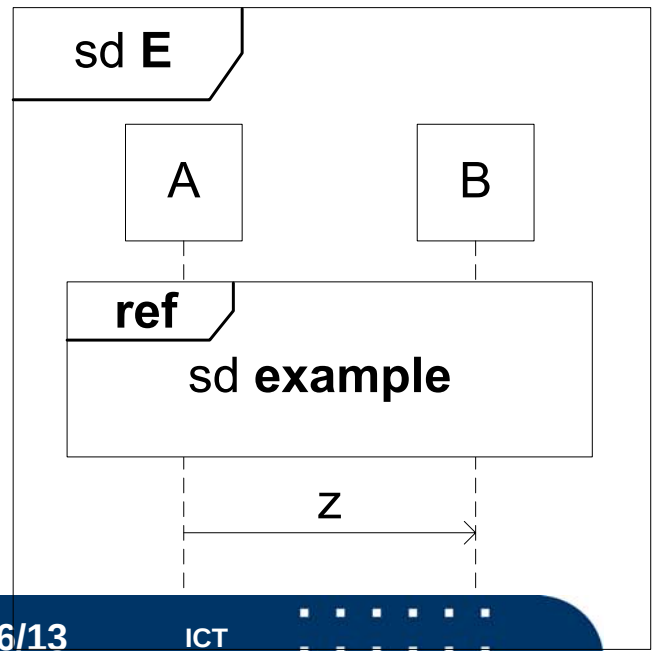
- The UML is not a policy specification language

- Objective: Evaluate the suitability of using the UML for policy specification
  - Specifically UML sequence diagrams

Sequence Diagrams

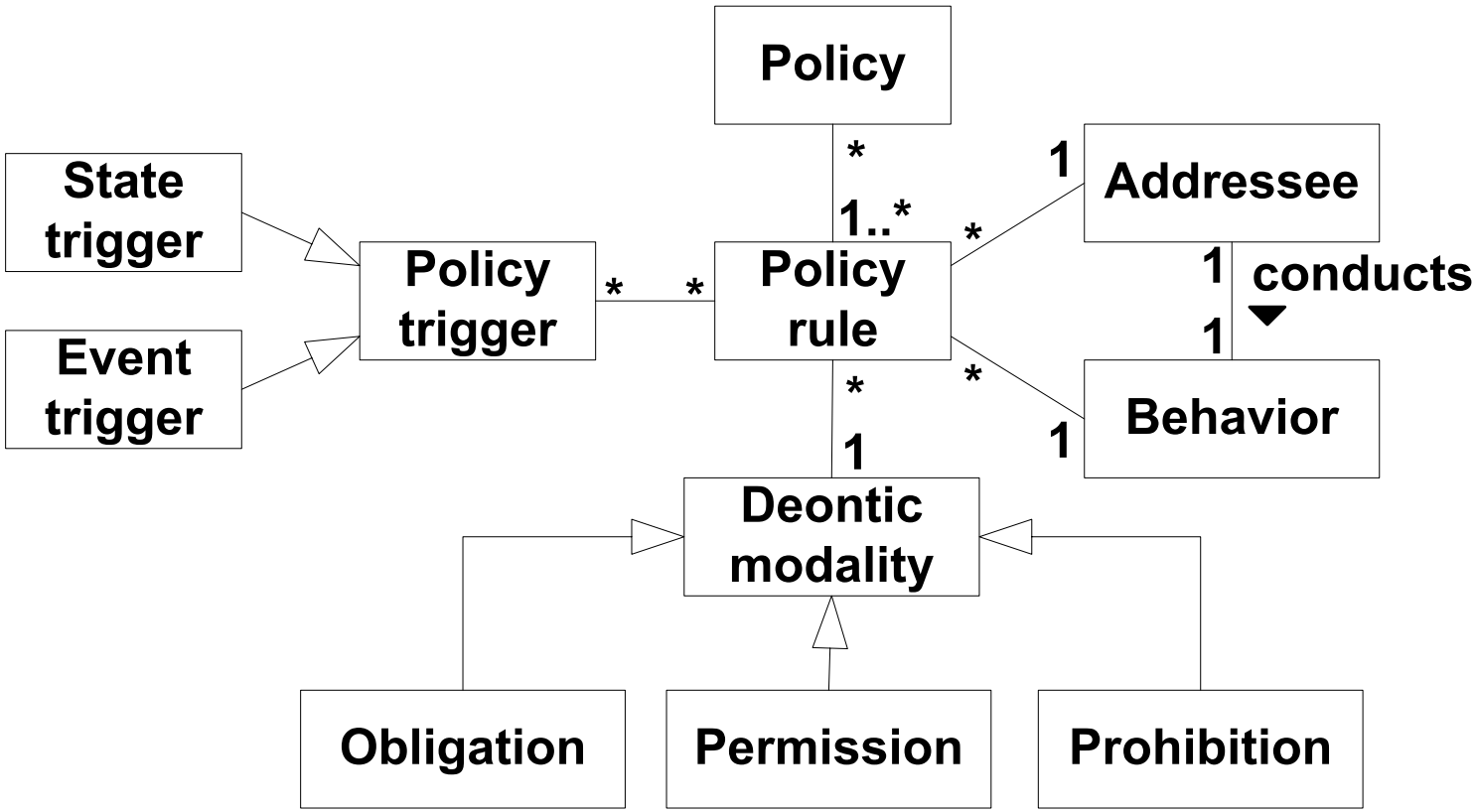


- Positive trace
  - Valid behavior
  - !x, ?x
- Negative traces
  - Invalid behavior
  - !x, ?x, !y, ?y

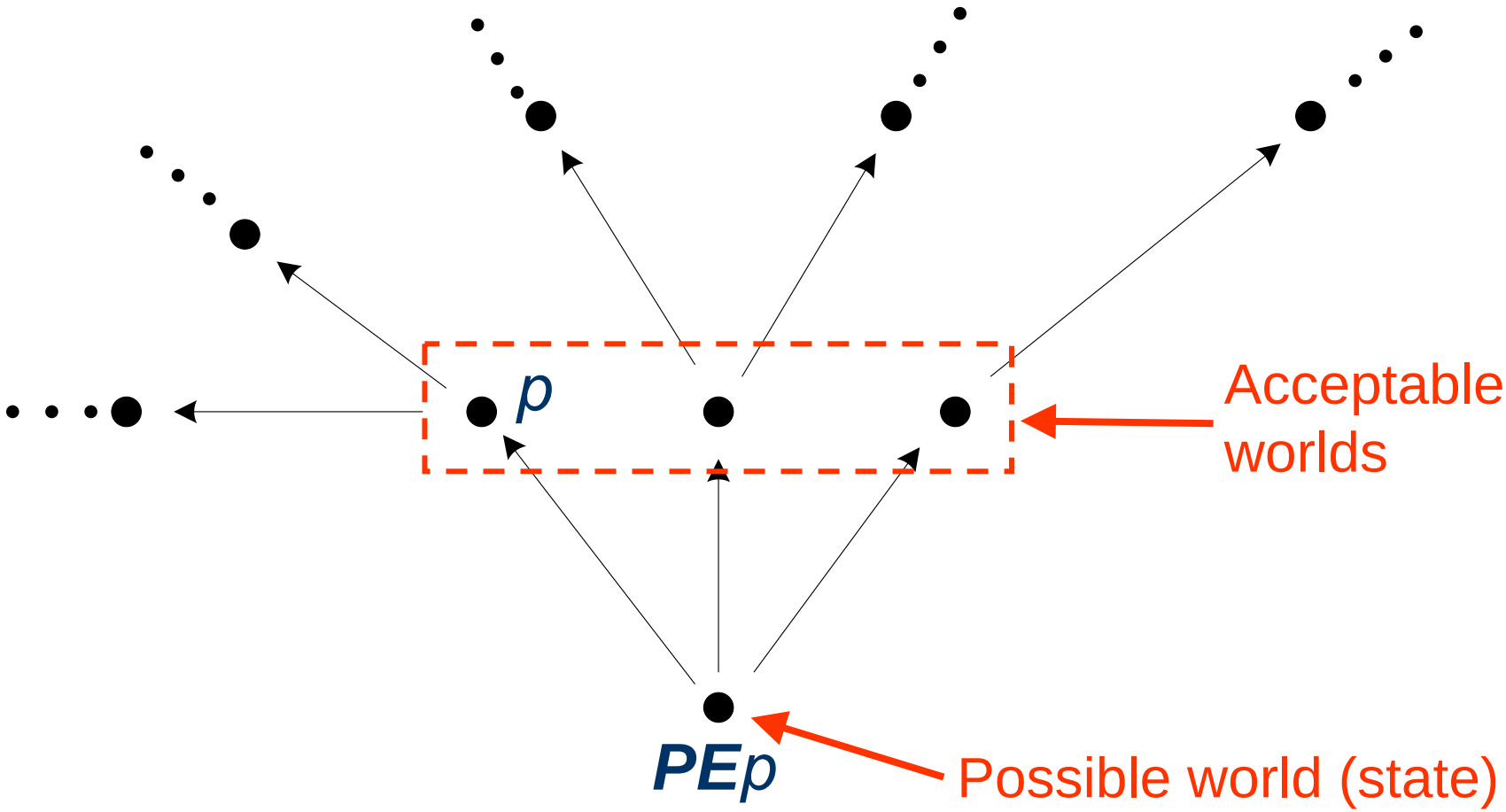


Policy Concepts

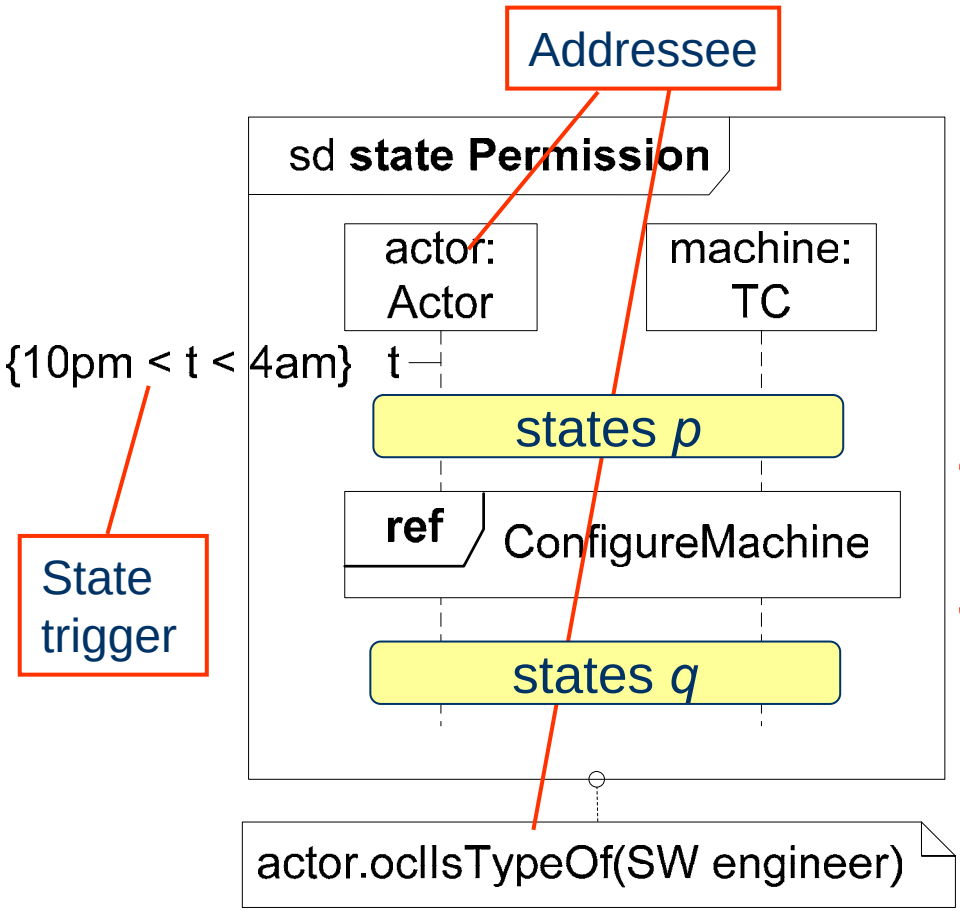
- A policy is a set of rules governing the choices in the behavior of a system (M. Sloman, 1996)



Kripke Semantics for Deontic Logic



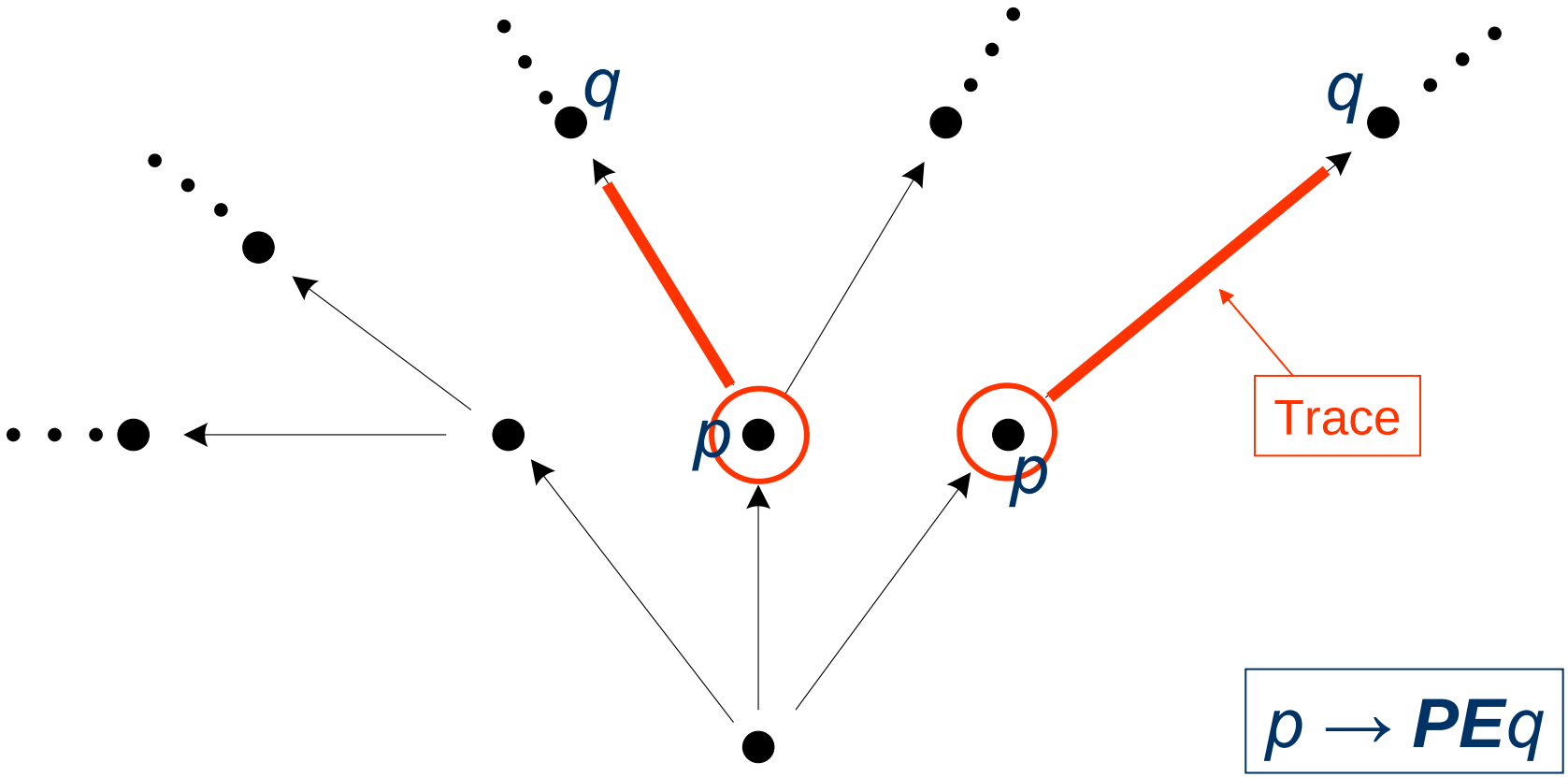
State Triggered Permission



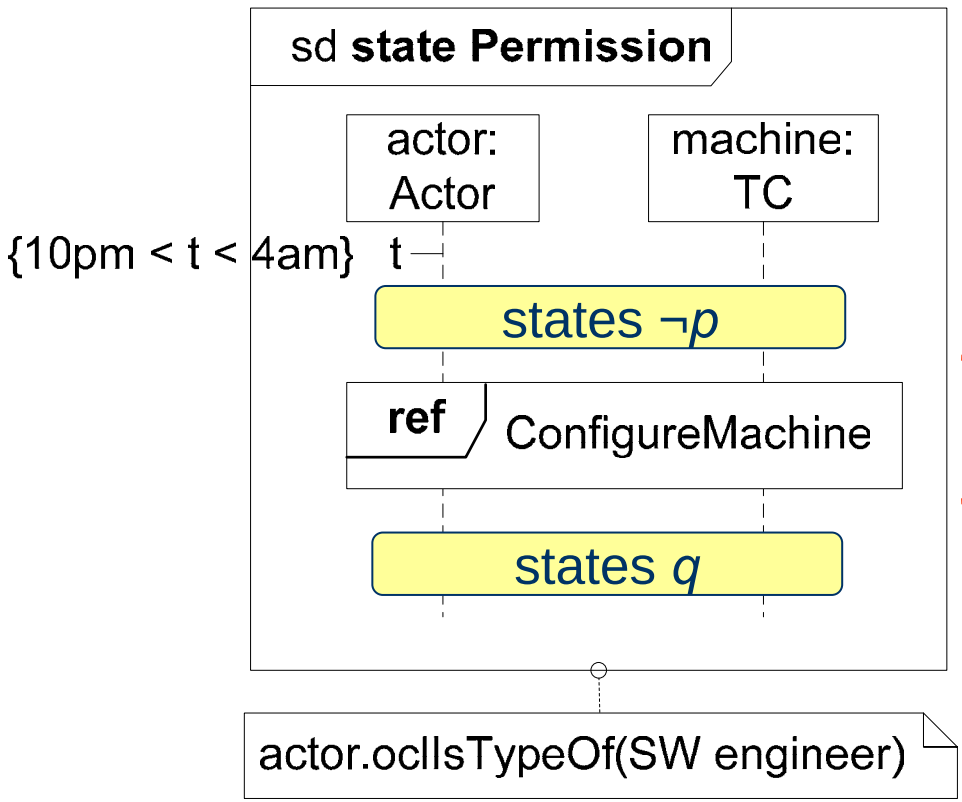
- The *ConfigureMachine* traces are positive when the time constraint and the OCL expression hold (permission)

Software engineers are **permitted** to *configure TC* when *time is between 10pm and 4am*

Kripke Semantics for Deontic Logic



Implied Prohibition



- The *ConfigureMachine* traces are negative when the time constraint or the OCL expression does not hold (prohibition)

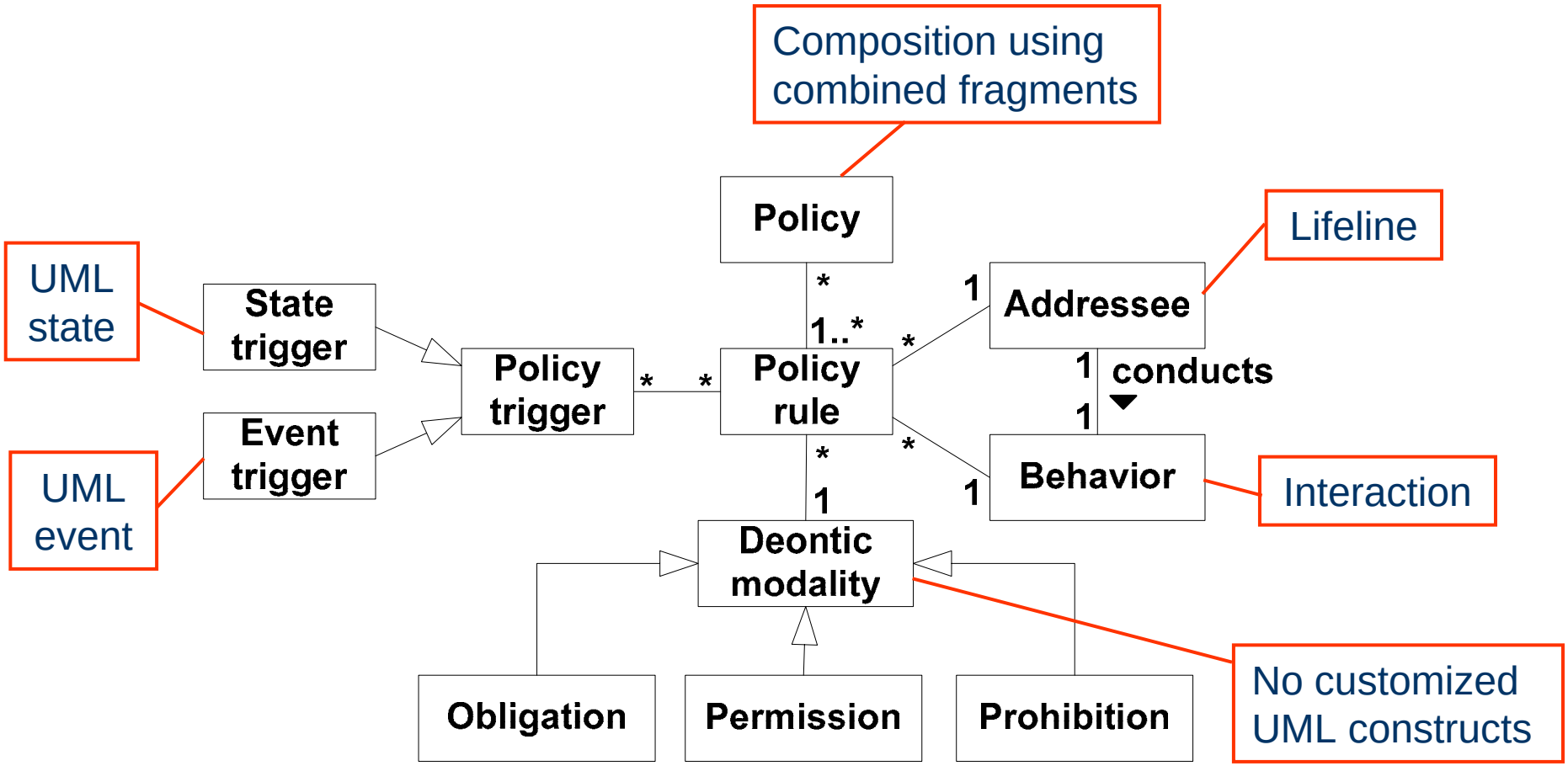
Negative trace

Software engineers are **permitted** to *configure TC* when *time is between 10pm and 4am*

Evaluation – Deontic Expressions

- Sequence diagrams can express permissions, obligations and prohibitions
- Fulfill the requirements to the semantics as defined in standard deontic logic
- Particularly, the following hold
  - $\boldsymbol{OB}p \rightarrow \boldsymbol{PE}p$ (SDL axiom)
  - $\boldsymbol{PE}p \leftrightarrow \neg\boldsymbol{OB}\neg p$ (SDL definition)
  - $\boldsymbol{PR}p \leftrightarrow \boldsymbol{OB}\neg p$ (SDL definition)
- But with state triggers, we additionally get
  - $p \rightarrow \boldsymbol{PE}q$ implies $\neg p \rightarrow \boldsymbol{PR}q$
  - $p \rightarrow \boldsymbol{PR}q$ implies $\neg p \rightarrow \boldsymbol{PE}q$

Evaluation


Composition using combined fragments
Policy
UML state
State trigger
Event trigger
UML event
Policy trigger
Policy rule
Addressee
Lifeline
conducts
Behavior
Interaction
Deontic modality
Obligation
Permission
Prohibition
No customized UML constructs
*
1..*
1
1
1
1

Future Work

- Based on the evaluation
- introduce customized policy specification constructs
- with a semantics matching deontic logic

- The syntax and semantics should capture the theory of standard deontic logic