

# Verification of Access Control Policies for REA Business Processes

Vahid Karimi  
Don Cowan

University of Waterloo  
Waterloo, Ontario, Canada

# Outline

- Motivation
- Approach
- Limitations
- Future Work
- Summary

# Motivation

- The formal verification of access control policies is widely researched.
- Making sure that what we specify is what we want.
- Some typical analysis:
  - Consistency properties
  - Safety properties (e.g., an individual does not have a right that he/she should not have.)

# Motivation (contd.)

- We intend to verify access control policies.
- In addition, can we describe access control policies for business processes such that the integration, and not necessarily the inclusion, of these policies into business processes is more straightforward?
- Can we describe business models and access control policies using similar models?

# Motivation (contd.)

- Researchers advocate the separation of access control policies from applications for several reasons:
  - policies can be analyzed separately;
  - policies are not scattered in applications; and
  - policies can be shared among several applications, and their inclusion, separately, in every application is redundant.

## Motivation (contd.)

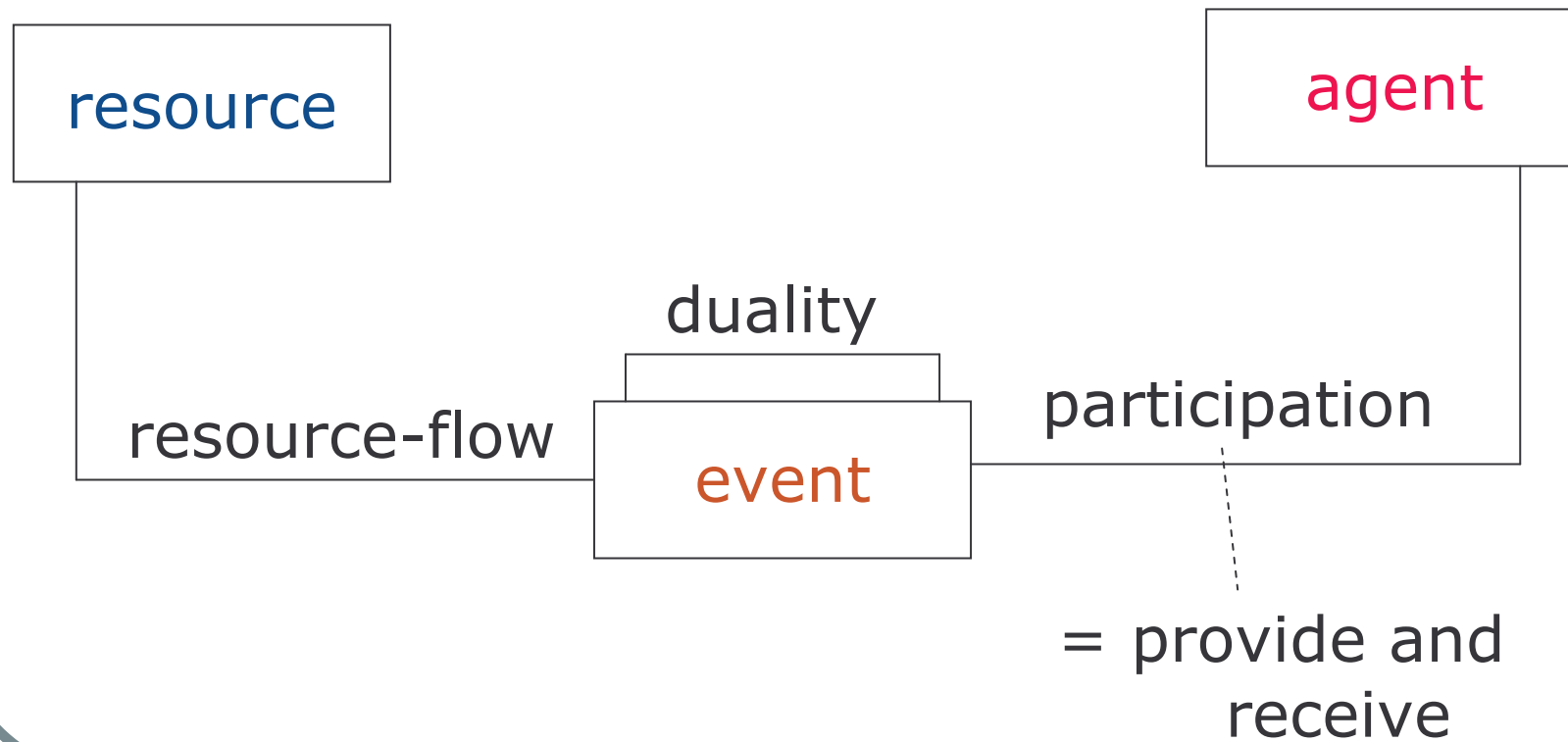
- Despite these benefits, this approach creates a problem with the integration of these policies into applications:
  - This integration may become difficult because these policies are usually based on various models and languages usually different from those of applications.

# Approach

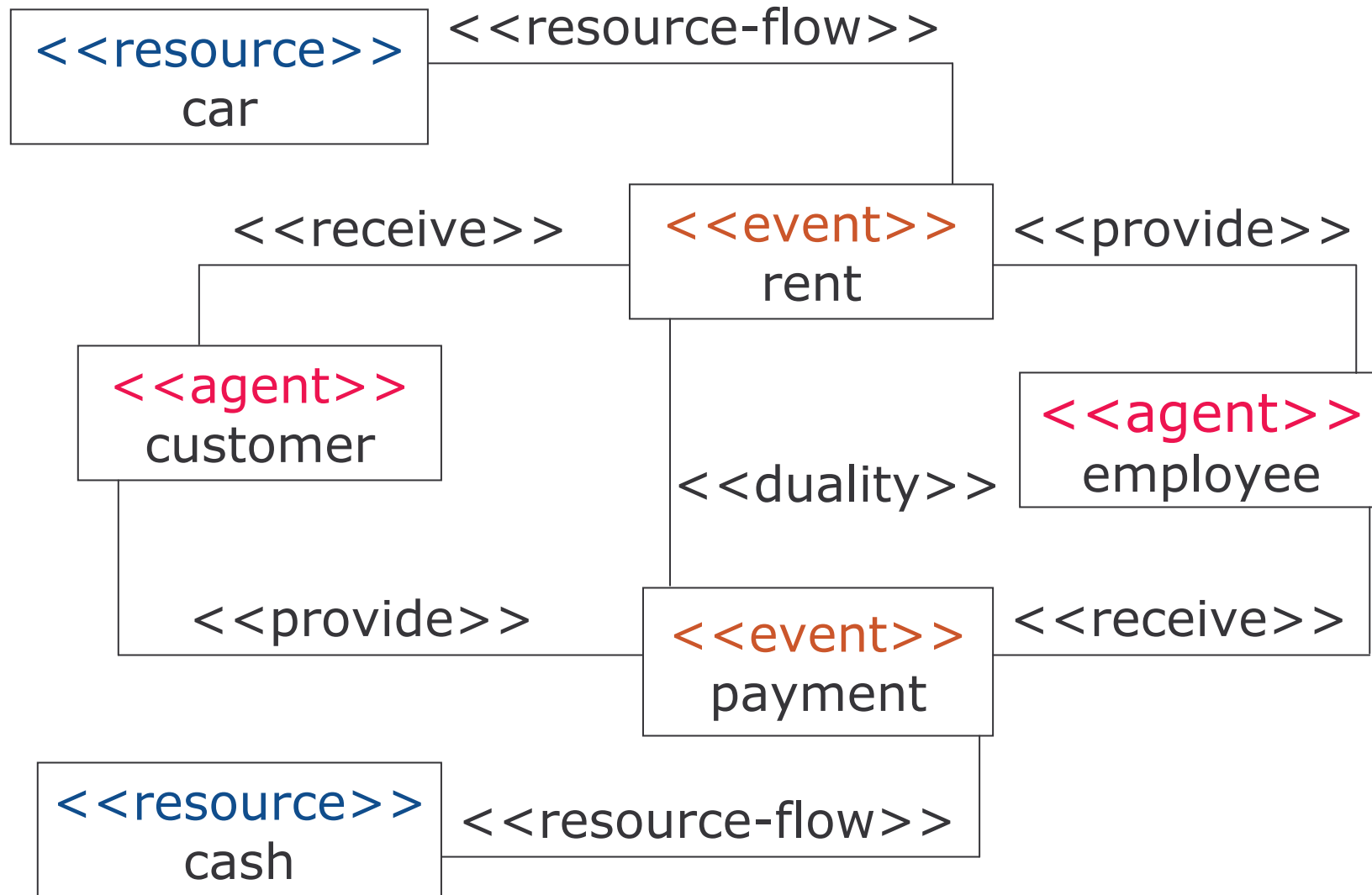
- Resource-Event-Agent (REA)
  - Introduced in 1982; extended since
  - Presented as an ontology and patterns
- Two broad categories of REA processes
  - exchange (e.g., sales and loans) and
  - conversion (e.g., product creation)
- Any number of exchanges and/or conversions can be connected.

# Approach (contd.)

- A basic REA

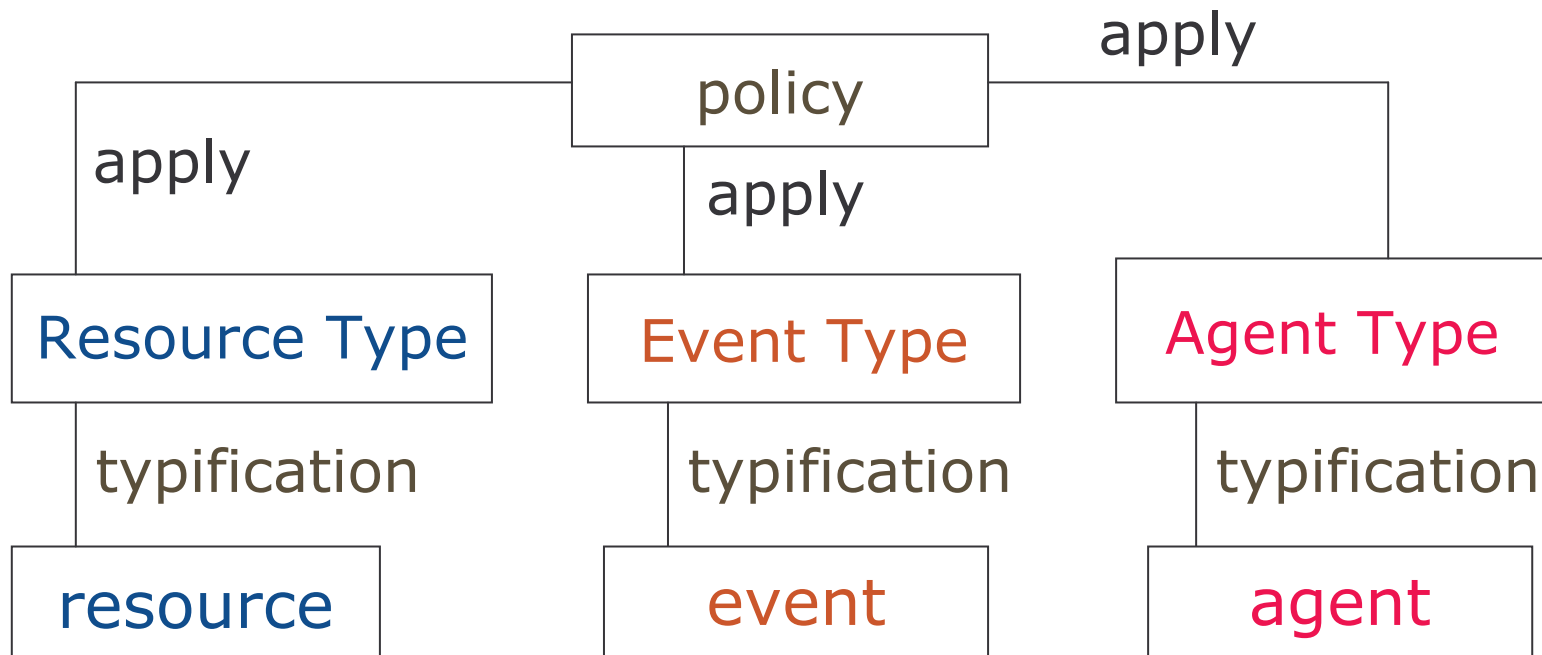


# REA Example for Renting a Car

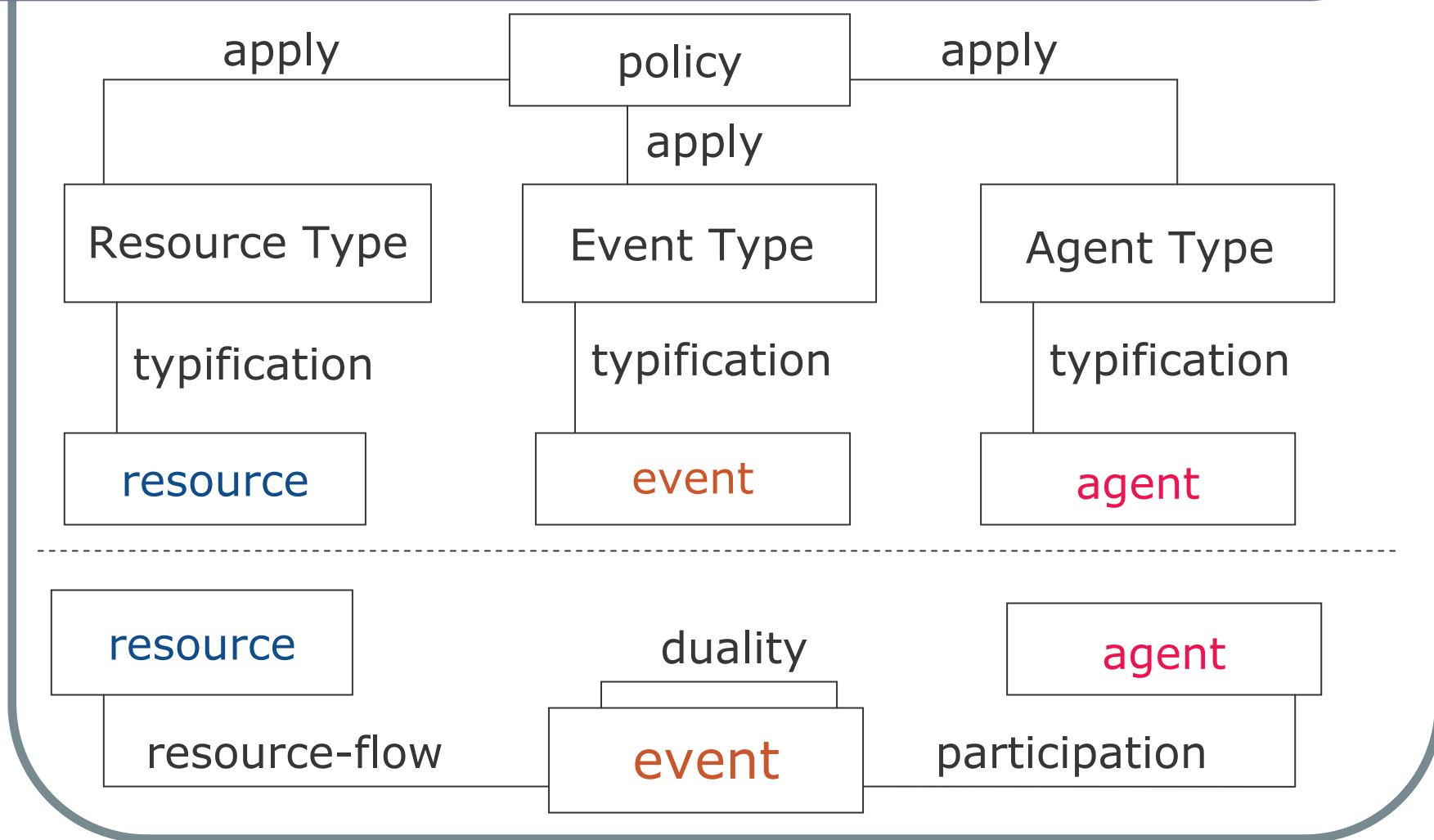


# Describing Policies, General Approach

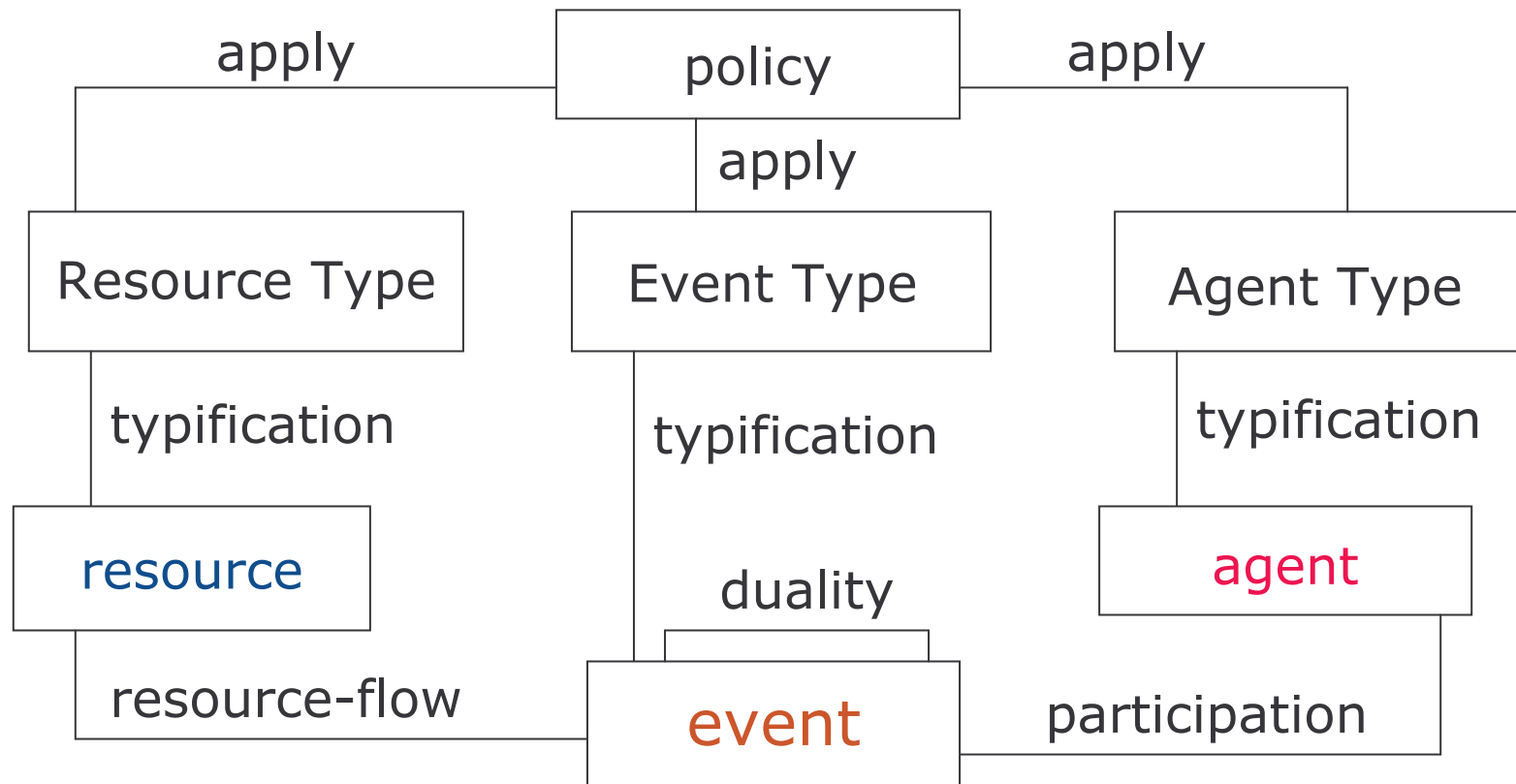
Policies apply to the types or groups of resource, agent, and event.



# Adding Policies to a Business Process, General Approach

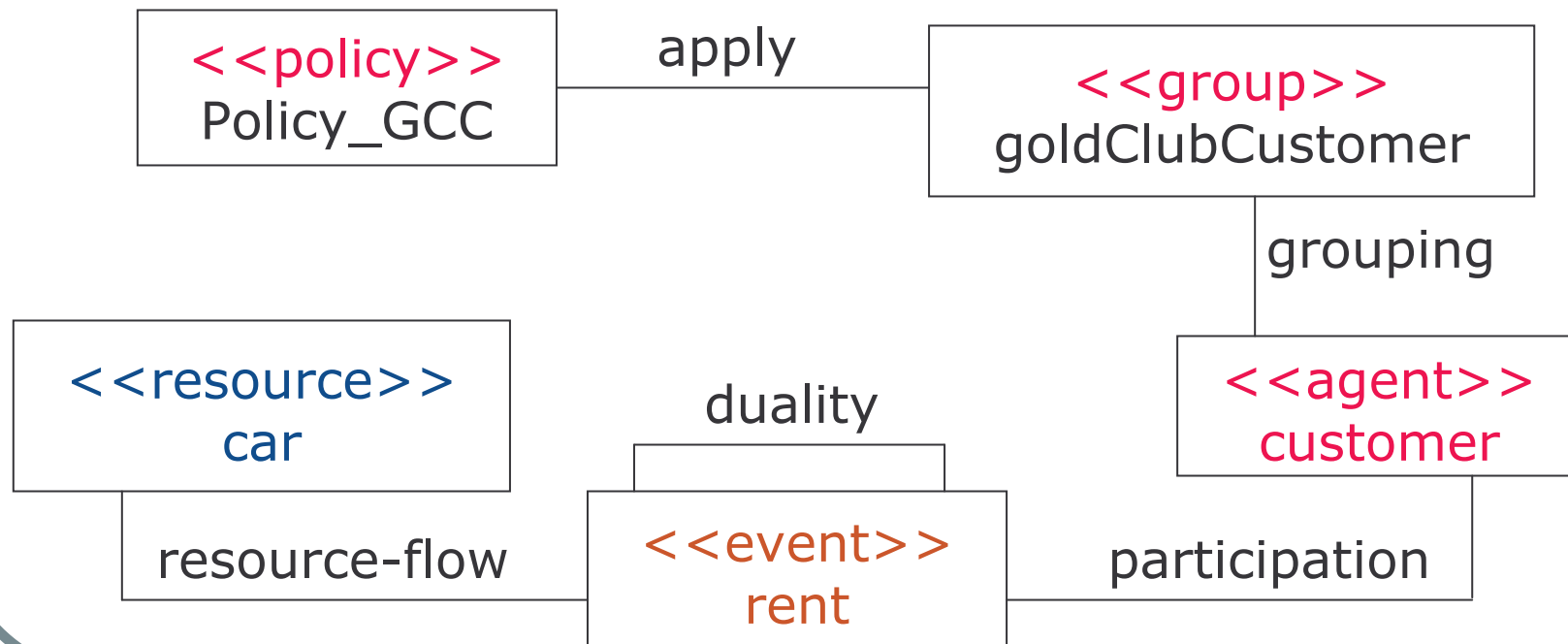


# Adding Policies to a Business Process General Approach (contd.)



# Policy One

- P1: Gold-club customers get a discount whenever they rent a car.



# Three Policies

- P1: Gold-club customers get a discount whenever they rent a car.
- P2: People who get a discount (i.e., gold-club customers) cannot be both customers and car rental employees within one exchange.
- P3: Car-rental employees can receive certain discounts whenever they rent a car.

# Policy Analysis

- Consistency of policies
- The purpose of policy P2 is to include separation of duties for preventing fraud.
- Separation of duties is a well-known principle
  - Static and dynamic
  - e.g., two separate individuals must authorize ordering items and paying for them.

# A Formal Method Approach

- **Alloy**: based on predicate logic
- **Alloy Analyzer** translates a problem into a Boolean formula and hands it to a SAT solver.
  - It also translates the SAT solver's solution into Alloy.
- Alloy, sometimes, is called a model finder rather than a model checker.
- Their use can be complementary.

# A Formal Method Approach (contd.)

## Comparing Alloy and Model Checkers:

- I) Model checkers are explicitly based on temporal properties, but Alloy is not.
- II) Alloy builds small, restricted-size models, not necessarily true for models of model checking.
- III) Because of the depth-first nature of Alloy's SAT analysis, Alloy may find an error faster than a model checker.

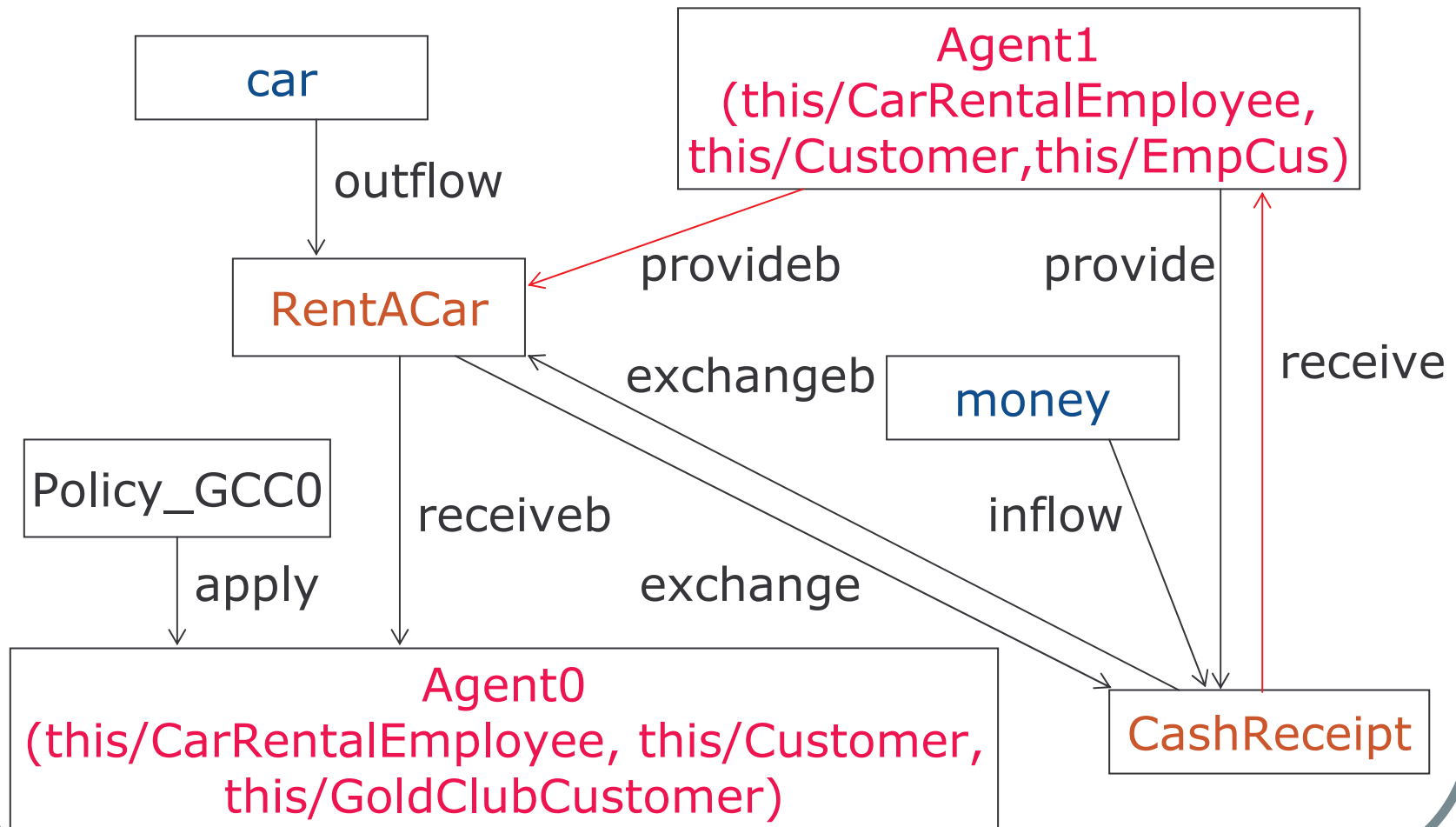
# A Formal Method Approach (contd.)

- Model checking and Alloy, more comparison:
  - Model checking
    - Designed for handling the complexity caused by the interaction of concurrent simple state machines
  - Alloy
    - Designed for handling state machines with complex structures

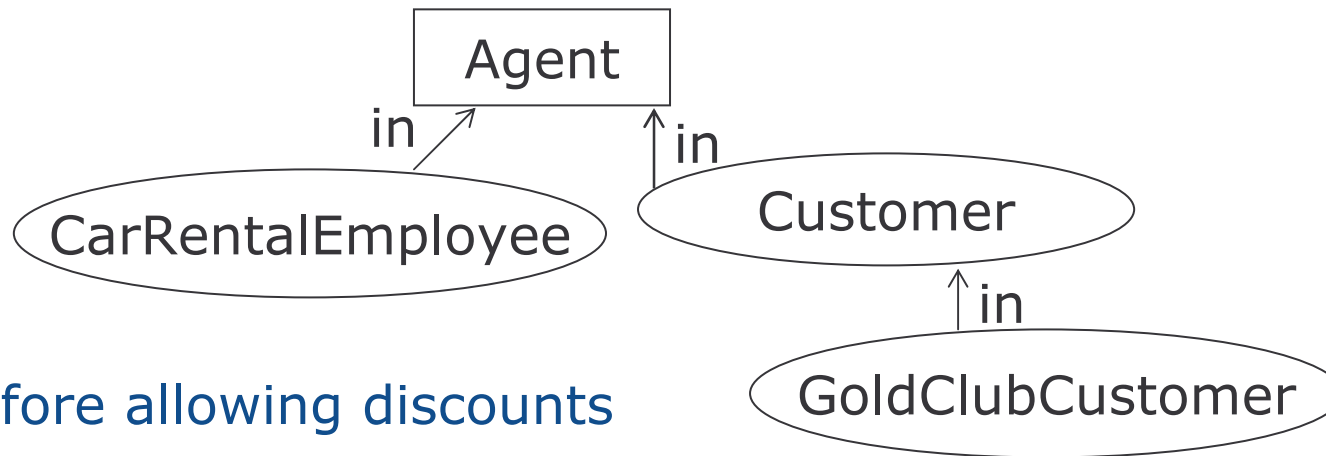
# Adding Policies One by One

- After adding policy three and verifying it
  - A counterexample is created because it does not hold.
  - A graph representation for this inconsistency can be viewed.
- On the other hand, the lack of a counterexample indicates that the policy or property is valid for the specified scope.
- In this case, a possible model of such inconsistency is shown next.

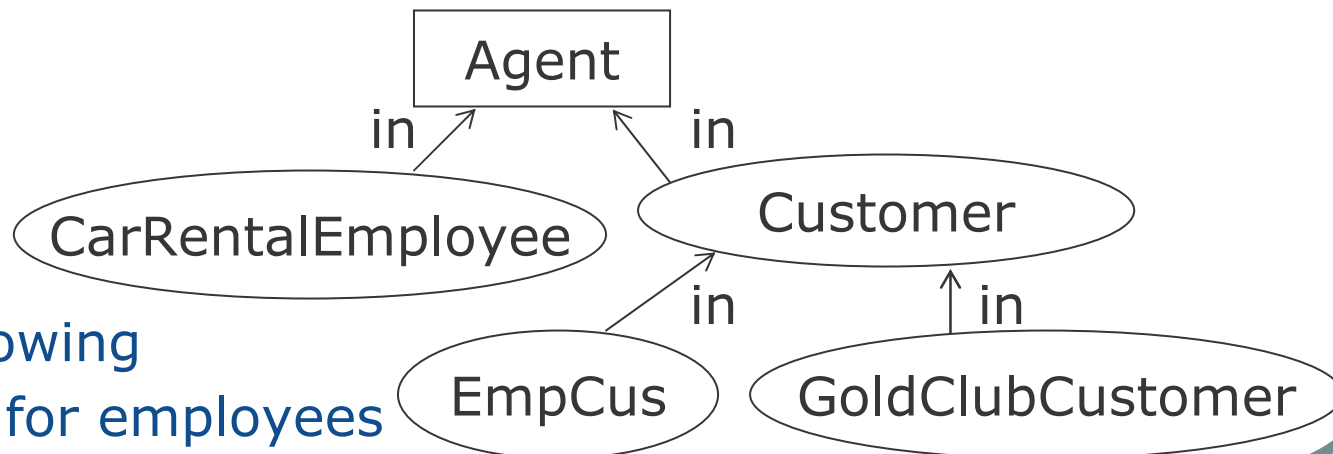
# A Counterexample



# A Counterexample (contd.)



a) Before allowing discounts for employees



b) After allowing discounts for employees

# Limitations

- Theoretical limitation
  - Undeciability
    - The safety analysis of an access control matrix for a general case is not decidable.
    - Not related to our case study
  - Intractability
    - Determining whether access control policy invariants are preserved
- Despite the intractability of analysis for the worst case, types of analysis such as SAT solving are well studied.

# Limitations (contd.)

- Formal Method approach Limitation
  - Predicate logic + transitive closure
  - The notion of scope
  - Temporal aspects
    - Alloy is not the best tool for describing and analyzing temporal properties.
- Uses REA business process
- Mapping to the other languages for describing business processes may not be straightforward

# Future Work

- The addition of dynamic aspects of REA
- The use of temporal properties
  - a richer set of properties
- The use of more than one process
  - Processes are connected together
- The use of other formal method approaches
  - e.g., model checking

# Summary

- The specification of access control policies in conjunction with a REA business process
  - using the same foundational building blocks
  - more straightforward integration of policies and business processes
- The use of Alloy as a formal method
- Limitations
- Future work