

**PPSWR 2005  
Dagstuhl Seminar**



# **REWERSE WG 12**

## **Policy Language, Enforcement, Composition**

Hannover, Linkoeping, Naples, St. Gallen, Turin, Zurich

September 15<sup>th</sup>, 2005

# REVERSE WG I2

## Mission

- **Integration of policies**
  - Security policies, Trust management
  - Business rules, Quality of service specs.
  
- Enhance **user control and awareness** on system behavior
  
- **Reduce the cost of building and maintaining** cooperative systems

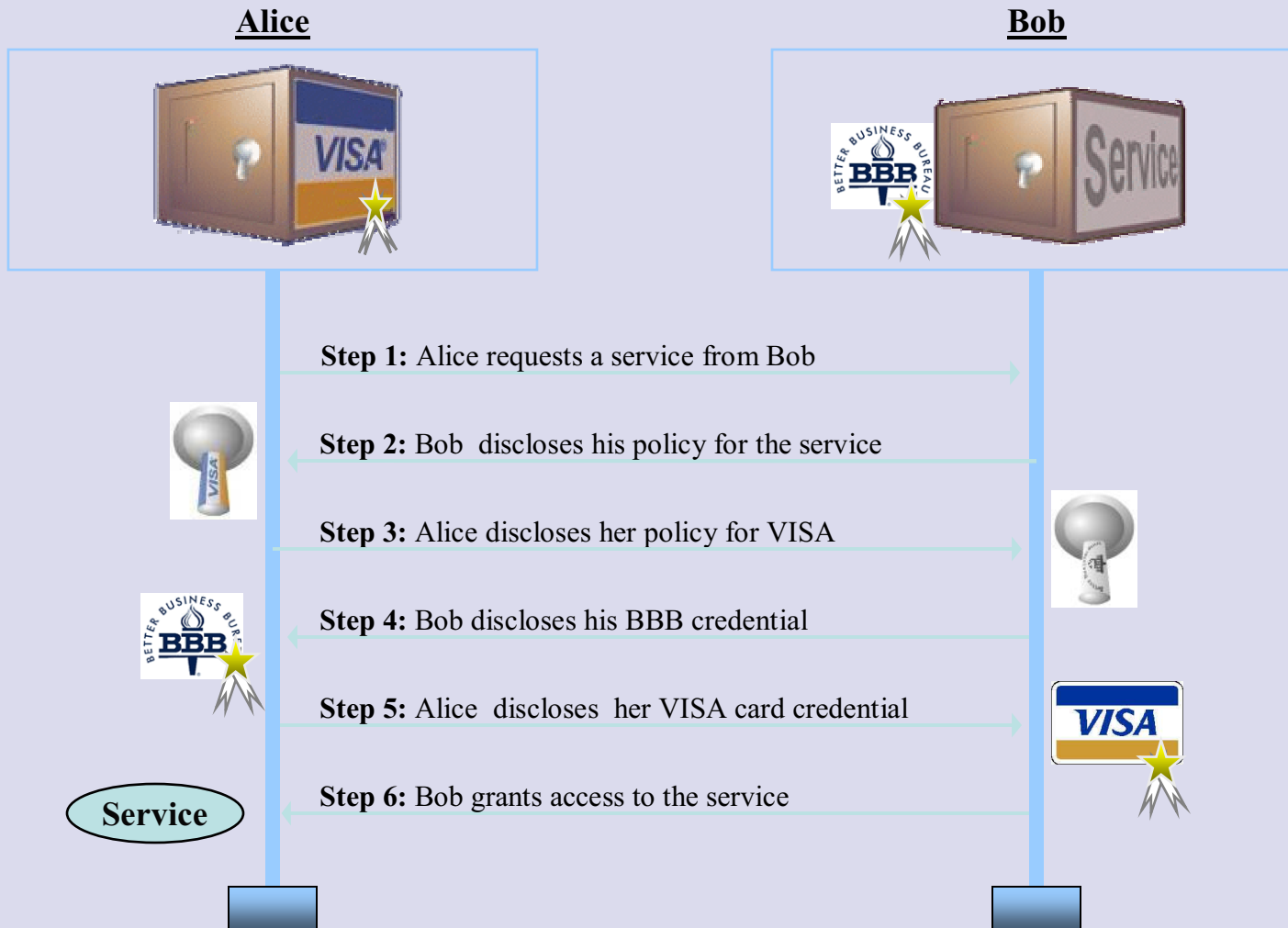
# Reference Scenario (I)

## General Picture

- **Each party has a policy to control the use of its resources**
  - Service access control (security)
  - Credential disclosure control (privacy)
  - Business rules
- Decisions are based on **peer properties**
- Properties are established by **iterative, bilateral disclosure** of certificates and declarations, i.e. **negotiations**

# Reference Scenario (II)

## Example: Security and Privacy



# Reference Scenario (III)

## Explanations

**Suppose Alice's request is rejected**

**She may want to ask questions like:**

- Why didn't you accept my credit card?

**Other possible queries**

- How-to queries
- What-if queries
  - Would I get the special discount on financial products X if I were locally employed?

# Reference Scenario (& IV)

## Natural Language

We are aiming at natural rule/query formulation

- **Users can download the files in folder `historical_data` if the creation date precedes 1/1/2000**

Policy enforcement, negotiations, query answering should all be **automatically** derived from such specifications

- **Attempto Controlled English**

# Current Achievements (I)

## I2-D1: Rule-based Policy Specification: State of the Art and Future Work

- A revised version is currently submitted as a chapter of the book on  
**Security in Decentralized Data Management**  
Publisher: Springer
- Already received positive feedback from reviewers

# Current Achievements (II)

## I2-D2: Policy Language Specification

- **PROTUNE** Policy Language

Excerpts and extensions published in the proceedings of

- IEEE POLICY'05
- Semantic Web Policy Workshop at ISWC'05

# Current Achievements (III)

## I2-D3,I2-D5: ACE Extensions

- Prototype evolving as expected
- Start of a cooperation between I2 (Zurich) and A2 (Dresden) with the goal to express knowledge of protein interactions in Attempto Controlled English.
  - Started as a master thesis with the working title "Expressing Ontological Knowledge of Protein Interactions in Attempto Controlled English".

# Current Achievements (IV)

## I2-D4: Advanced Policy Queries (I)

- Currently being released
  - Waiting for second external review
- Advanced Query Answering
  - Why, Why-Not, How-to, What-if queries
- Two level of explanations
  - Concise
  - Full or detailed
- Constructed at client side
- Based on
  - Set of (computed) answer substitutions
  - Verbalization patterns

# Current Achievements (V)

## I2-D4: Advanced Policy Queries (II)

### ■ Policy example

```
[r2] allow( download(Resource) ) :-  
        public(Resource).
```

```
[r3] allow( download(Resource) ) :-  
        authenticated(User),  
        has_subscription(User,Subscription),  
        available_for(Resource,Subscription).
```

```
[r4] allow( download(Resource) ) :-  
        authenticated(User),  
        payed(User,Resource).
```

```
authenticated(User).explanation:[User,is,authenticated].  
allow(download(Resource)).  
    explanation:[it,is,allowed,to,download,Resource].
```

# Current Achievements (VI)

## I2-D4: Advanced Policy Queries (III)

- Why-not concise explanation for “allow(download(paper14.pdf))”

I CAN'T PROVE THAT it is allowed to download paper14.pdf  
BECAUSE:

Rule [r3] is not applicable:

THERE IS NO User SUCH THAT User is authenticated [details]

AND

Rule [r4] is not applicable:

THERE IS NO User SUCH THAT

    User is authenticated [details]

MOREOVER

THERE IS NO User SUCH THAT

    User has paid for paper14.pdf [details]

# Current Achievements (VII)

## I2-D4: Advanced Policy Queries (IV)

- Why-not concise explanation for  
“allow(download(paper14.pdf))”

Rule [r2]:

Everybody can download public objects  
the rule is not applicable

[see internal format]  
[details]

Rule [r3]: (no summary available)

it is allowed to download paper01234.pdf IF  
THERE EXIST User AND Subscription SUCH THAT  
User is authenticated  
AND  
User has subscription Subscription  
AND  
paper01234.pdf is available for Subscription  
the rule is not applicable

[details]

Rule [r4]:

Users can download any object if they pay for it  
the rule is not applicable

[see internal format]  
[details]

# Current Achievements (VIII)

## I2-D4: Advanced Policy Queries (V)

- Policy Example

```
[r9] id(Cred) :-  
    valid_credential(Cred),  
    Cred.type:T,  
    Cred.issuer:CA,  
    isa(T,id),  
    trusted_for(CA,T).
```

```
Id(Cred).explanation:[Cred,is,a,credential].  
trusted_for(CA,Type):[CA,is,trusted,for,Type].
```

- Why-not concise explanation for “id(Credential)” (with a disclosed credential)

I CAN'T FIND ANY Cred SUCH THAT Cred is an id BECAUSE:  
c012 is a credential with type id and issuer Open University [details]  
id is an id\_type [details]  
BUT  
IT IS NOT THE CASE THAT Open University is trusted for id [details]

**Thanks !**

# Questions

WG I2 Coordinator: Piero A. Bonatti (Naples)

# PROTUNE Policy Language (I)

## The Language

A logic programming language with “reserved predicates”

Types of predicates:

- **Decision Predicates:** can be queried by users
- **Abbreviation/Abstraction Predicates**
- **Constraint Predicates:** comprise usual equality and disequality predicates
- **State Query Predicates:** read the state without modifying it
- **Provisional Predicates:** may be made true by means of associated actions that may modify the current state

# PROTUNE Policy Language (II)

## Example

### Authorization rules

```
allow(enter_site()) :-  
  declaration(usr=U,pwd=P),  
  has_passwd(U,P)
```

```
allow(buy()) :-  
  credential(type=visa, num=N, exp=E),  
  not blocked(N), E>today
```

# PROTUNE Policy Language (III)

## Provisional Predicates (I)

### Authorization rules

```
allow(enter_site()) :-  
    declaration(usr=U,pwd=P),  
    has_passwd(U,P)
```

```
allow(buy()) :-  
    credential(type=visa, num=N, exp=E),  
    not blocked(N), E>today
```

- To be provided by the client
- Similar to an abduction problem

# PROTUNE Policy Language (IV)

## Provisional Predicates (& II)

### Authorization rules

```
allow(enter_site()) :-  
  declaration(usr=U,pwd=P),  
  has_passwd(U,P)
```

```
allow(buy()) :-  
  credential(type=visa, num=N, exp=E),  
  not blocked(N), E>today
```

- May be associated to actions
- E.g. Contact VISA web site

# PROTUNE Policy Language (V)

## Local State Predicates

### Authorization rules

```
allow(enter_site()) :-  
  declaration(usr=U,pwd=P),  
  has_passwd(U,P)
```

```
allow(buy()) :-  
  credential(type=visa, num=N, exp=E),  
  not blocked(N), E>today
```

- Local state predicates – **sensitive**
- Requires **policy filtering**

# PROTUNE Policy Language (VI)

## Other State Predicates

### Authorization rules

```
allow(enter_site()) :-  
  declaration(usr=U,pwd=P),  
  has_passwd(U,P)
```

```
allow(buy()) :-  
  credential(type=visa, num=N, exp=E),  
  not blocked(N), E > today
```

- Time dependent state terms/predicates
- Including the **negotiation state**

# PROTUNE Policy Language (VII)

## Metapolicies (I)

Attribute	Domain	Range
action	provisional predicates	commands
actor	provisional predicates	self, peer
aggregation_method	cost and sensitivity attributes	max, min, sum, adopt(Predicate)
cost	provisional predicates	number
evaluation	state predicates	immediate, delayed, concurrent
expected_outcome	provisional predicates	success, failure, undefined, unknown
explanation	literals and rules	string expression
ontology	abbreviation predicates, credentials, declarations, actions	URI
predicate	literals	predicate names
selection_method	negotiator	certain_first, order(attribute_list), adopt(Predicate)
sensitivity	predicates, literals, rules	public, private, not_applicable
type	predicates, literals	abbreviation, constraint, decision, state_predicate, provisional, state_query

# PROTUNE Policy Language (& VIII)

## Metapolicies (& II)

```
table(Key,Data).evaluation:immediate ←  
    ground(Key).
```

```
logged(Msg,File).action:'echo'+Msg+'>'+File.
```

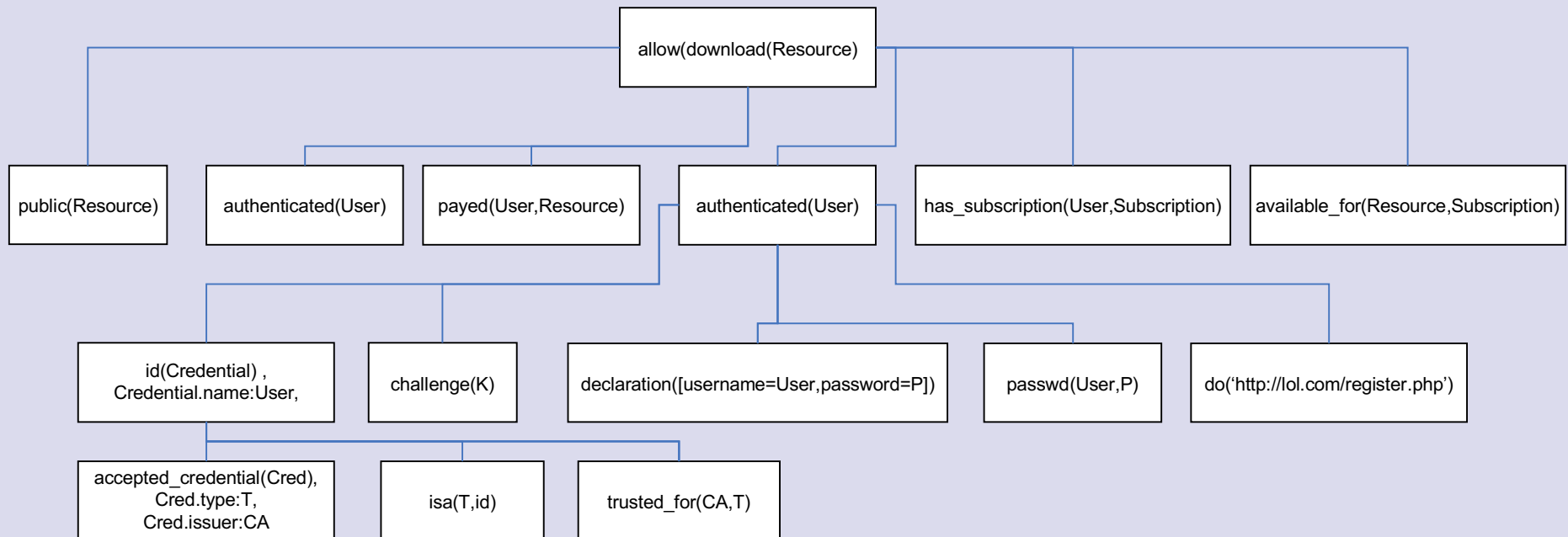
```
credential(_).ontology:URI.
```

```
abbrev(_).explanation:"this condition checks..."
```

# Current Achievements (III)

## I2-D4: Policy Language Specification (I)

### Policy example



`authenticated(User).explanation:[User,is,authenticated].`  
`allow(download(Resource)).`  
`explanation:[it,is,allowed,to,download,Resource].`

# Promotional Activities

- Position paper at Semantic Web Policy Workshop at ISWC'05
- Invitation as a panelist at COMPSAC'05 in Edinburgh in a Panel on Security and Privacy in Distributed Collaborative Systems (July 2005))
- Course on Attempto Controlled English at the University of Tartu (Estonia), Nov. 2005
- Invited Presentation of the Attempto/REWERSE work at the University of Tartu (Estonia), Nov. 2005
- Invitation to present the Attempto/REWERSE work in the Turing Center's Distinguished Lecture Series at the University of Washington (Seattle), Dec. 2005.