Blockchain based Access Control

Paolo Mori,
Damiano Di Francesco Maesa

Istituto di Informatica e Telematica
Consiglio Nazionale delle Ricerche

Laura Ricci

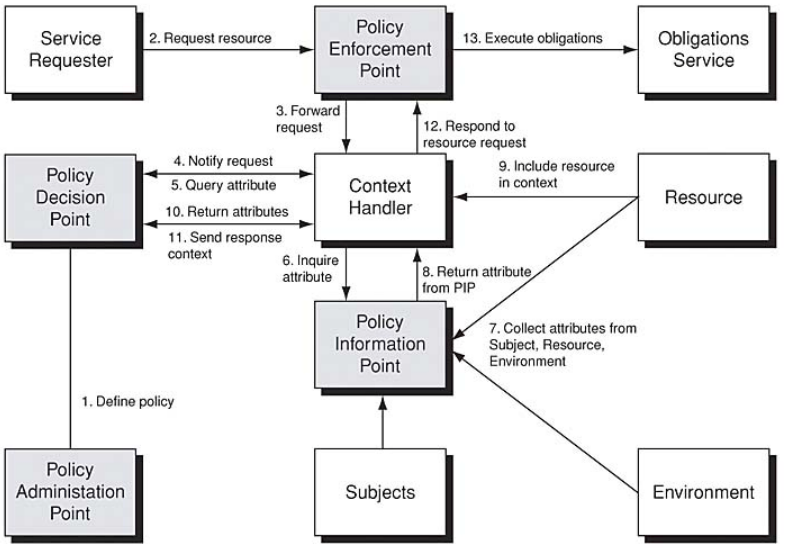
Dipartimento di Informatica
Università di Pisa

Agenda

- Background
  - Attribute Based Access Control
  - XACML
- Our Proposal
  - Implementation of the XACML based Access Control Service exploiting the Blockchain tecnology
  - Implementation details

Background:

Access Control and XACML



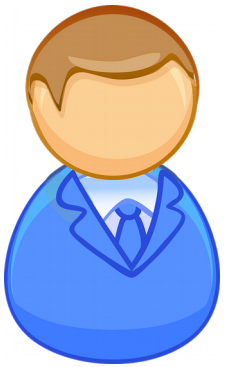
OASIS

Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right the perform it

Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right the perform it

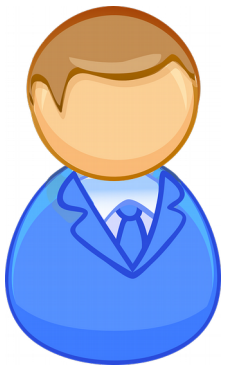


Subject

Resources

Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right the perform it



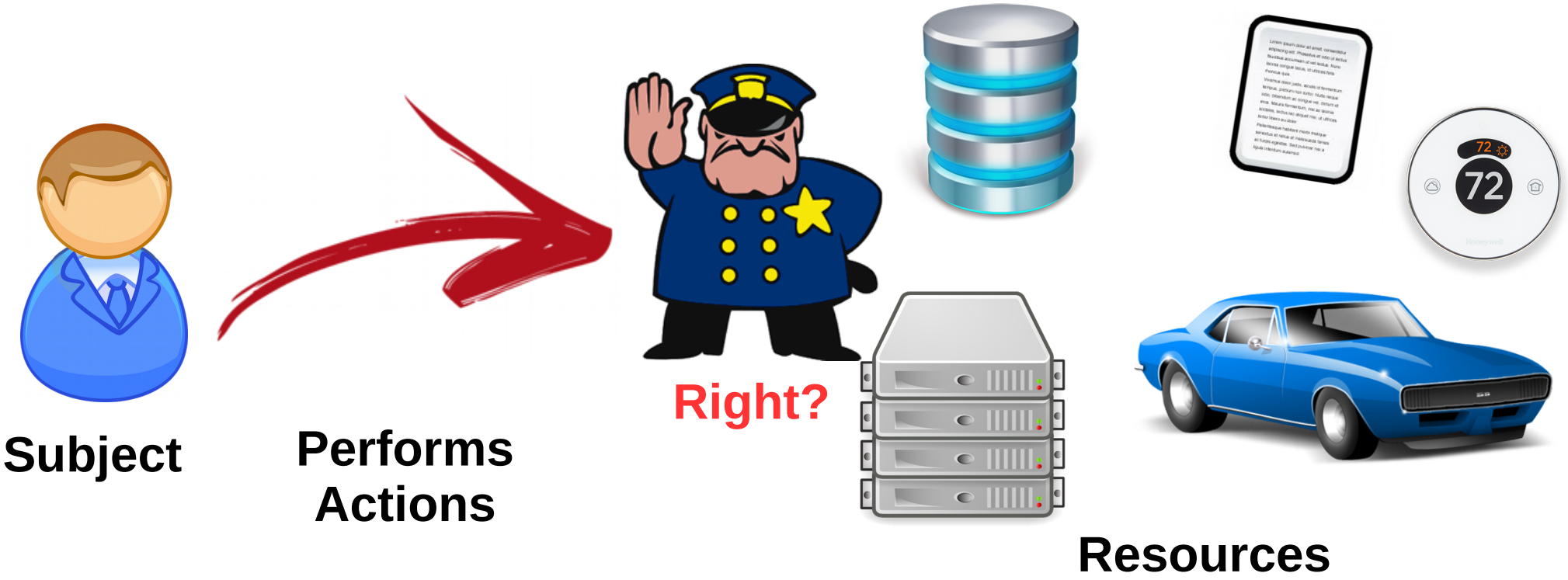
Subject

Performs Actions

Resources

Access Control

Technique to decide whether a Subject requesting to perform an Action on a Resource in a given Context holds the right the perform it



Attribute Based Access Control (ABAC)

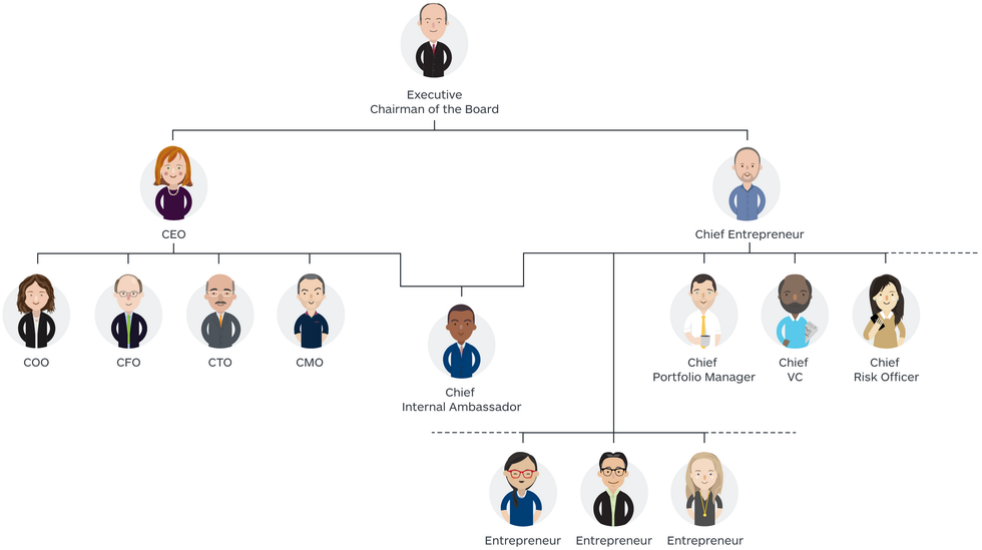
An access control method where subject requests to perform operations on objects are granted or denied based on assigned attributes of the subject, assigned attributes of the object, environment conditions, and a set of policies that are specified in terms of those attributes and conditions

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Guide to Attribute Based Access Control (ABAC) Definition and Considerations. NIST Special Publication 800-162

Attributes

- Attributes represent characteristics of the
  - Subjects
  - Resources
  - Actions
  - Environment
- Examples:
  - Subject



    - Role (e.g., in a company: Worker, Employee, Executive, CEO...)
    - Projects assigned to the subject
    - Physical location
  - Resources
    - Owner/producer
    - Number of copies of a document
    - Project of a document
    - Security classification

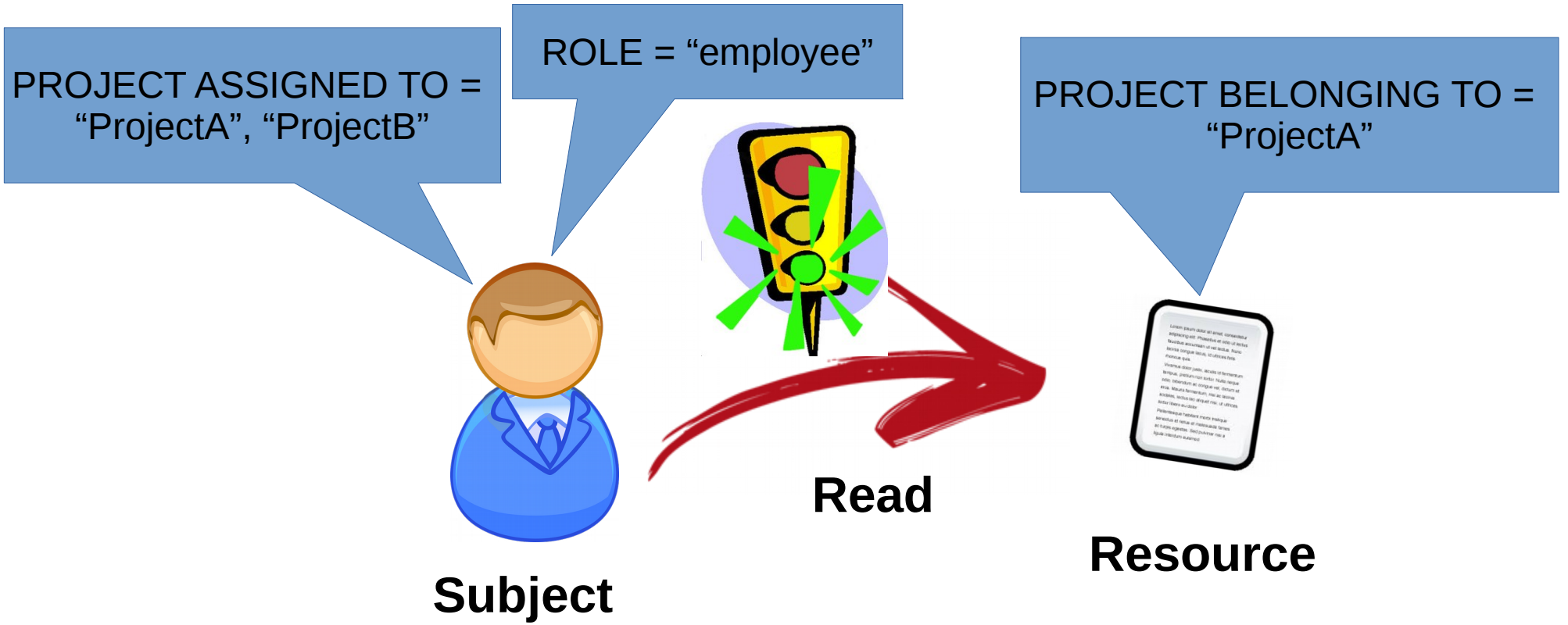
Access Control Policy

- Set of rules defined in terms of conditions on attributes of subjects, resources, actions, and environment
- Combination algorithms to decide the precedence among the rules

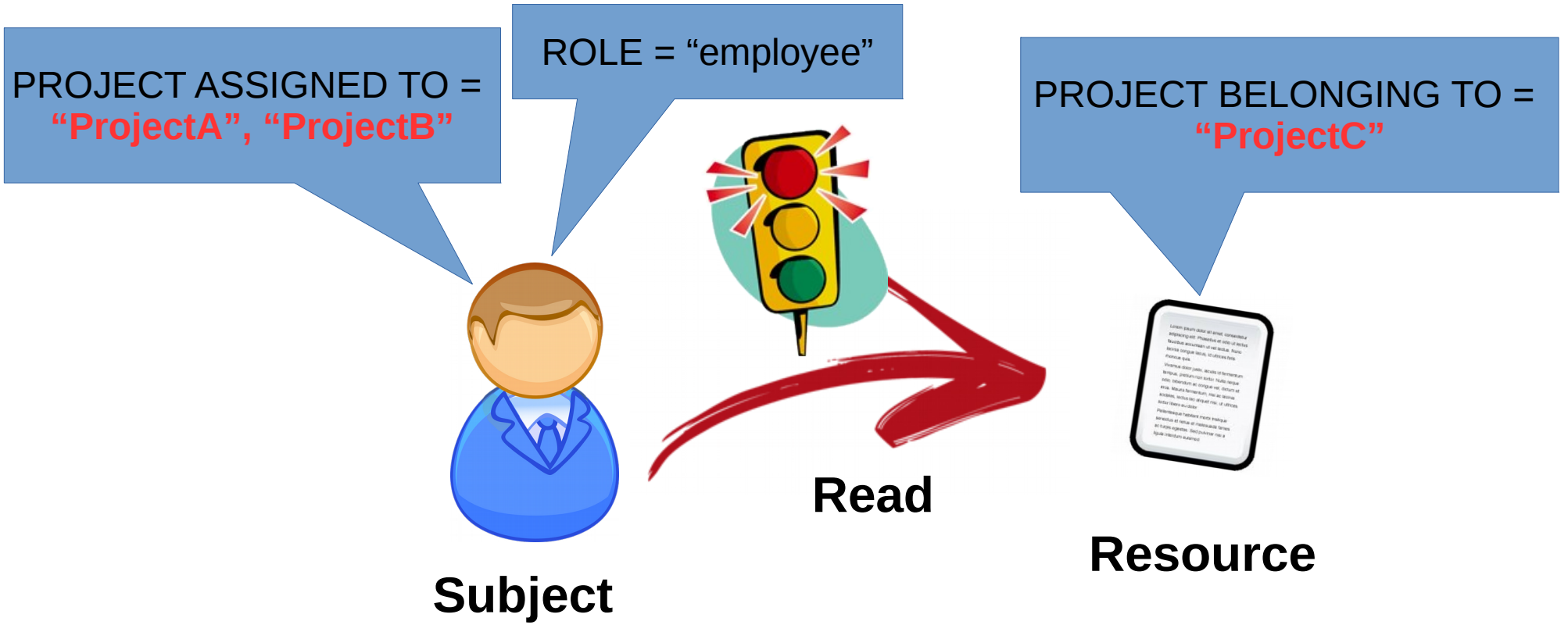
Example of Natural Language Access Control Policy:

Documents can be read if the value of the attribute ROLE of the Subject is “employee” and if the value of the attribute PROJECT BELONGING TO of the Resource is equal to (one of) the value of the attribute PROJECT ASSIGNED TO of the Subject

Access Control Policy



Access Control Policy



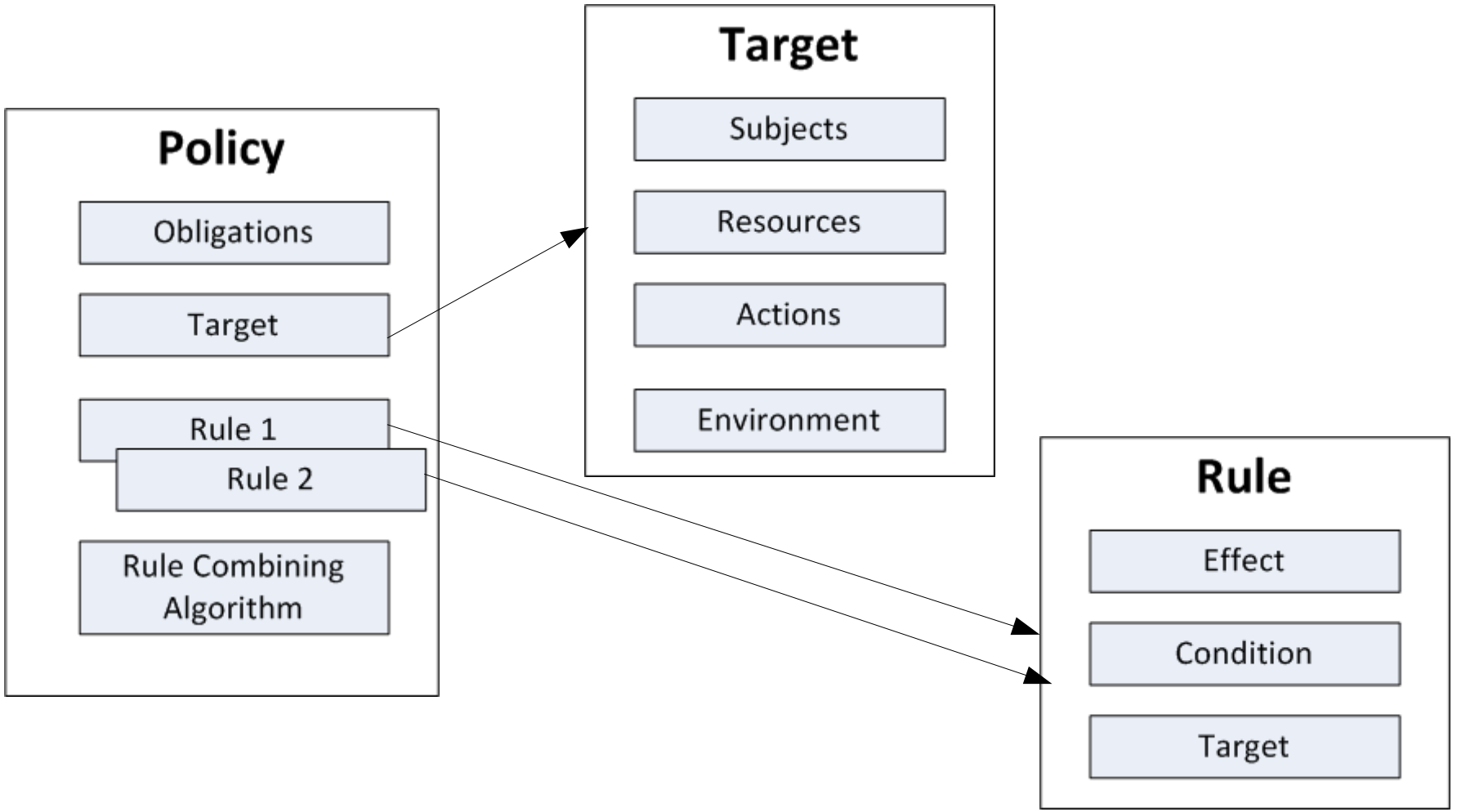
Extensible Access Control Markup Language 3.0 (XACML)

XACML defines:

- A XML-based Language to express Attribute based Access Control Policies
- A reference architecture for the Access Control Framework

eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01. OASIS Standard incorporating Approved Errata. 12 July 2017

Extensible Access Control Markup Language 3.0 (XACML): Policy Language



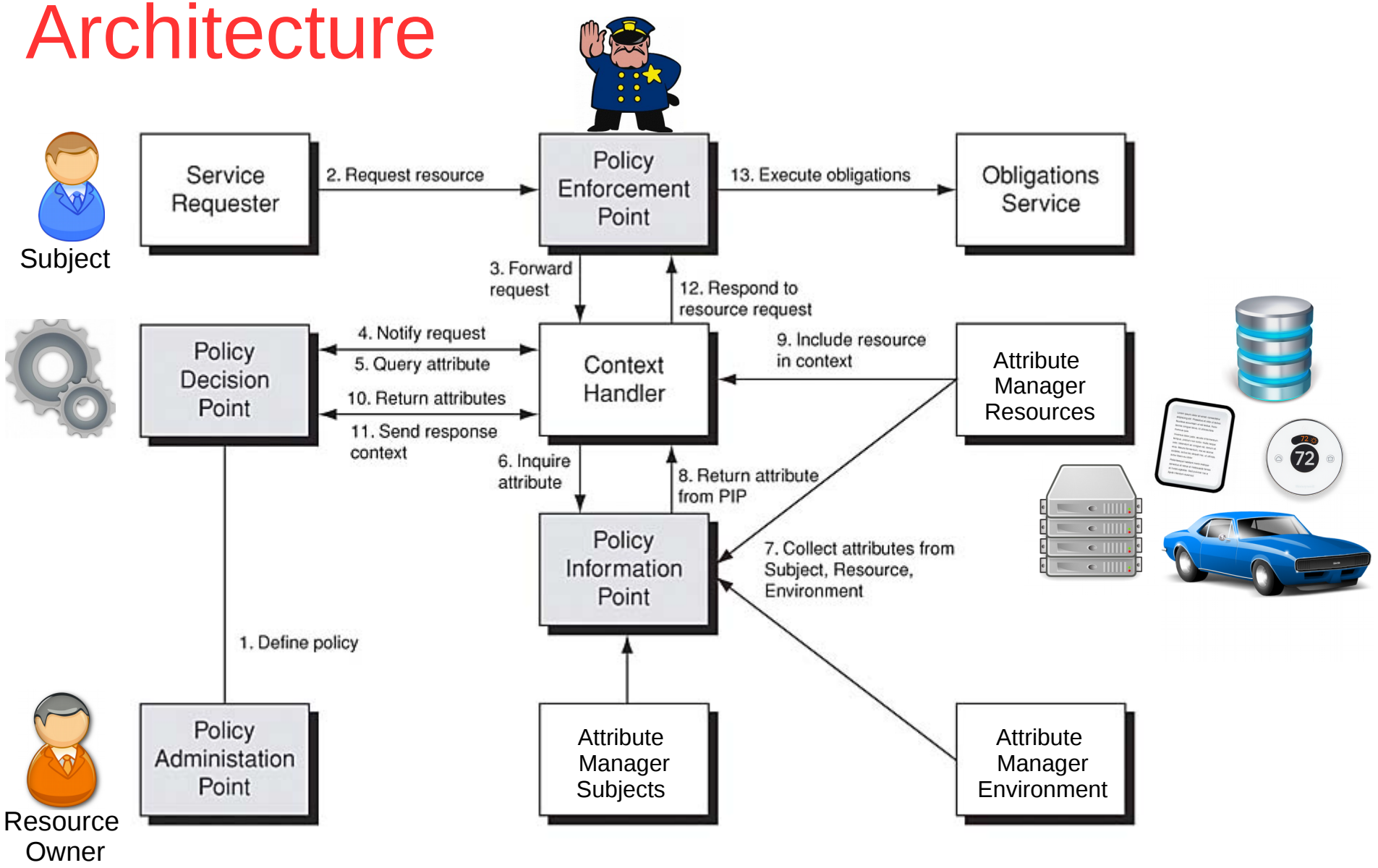
Extensible Access Control Markup Language 3.0 (XACML): Policy Example

```
<Policy PolicyId="PolicyCNR" RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-
combining-algorithm:first-applicable" Version="3.0">
  <Description>PolicyForCNR</Description>
  <Rule Effect="Permit" RuleId="authorization_1">
    <Target>
      <AnyOf> <AllOf>
        <Match MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
          <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            687fffb544f346baf8
          </AttributeValue>
          <AttributeDesignator AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"
            Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
            DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/>
        </Match>
      </AllOf> </AnyOf>
    </Target>
```

Extensible Access Control Markup Language 3.0 (XACML): Policy Example

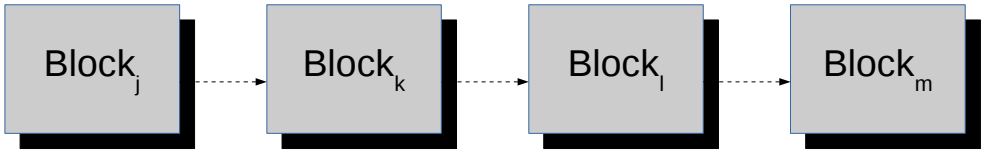
<Condition>
<Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:and"> <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal"> <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only"> <AttributeDesignator AttributeId="**urn:oasis:names:tc:xacml:1.0:subject:subject-location**" Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject" DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/> </Apply> <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">**EUROPA**</AttributeValue> </Apply> <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal"> <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only"> <AttributeDesignator AttributeId="**urn:oasis:names:tc:xacml:1.0:subject:subject-role**" Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject" DataType="http://www.w3.org/2001/XMLSchema#string" MustBePresent="true"/> </Apply> <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">**Executive**</AttributeValue> </Apply></Apply>
</Condition>
</Rule>

Extensible Access Control Markup Language 3.0 (XACML): Reference Architecture



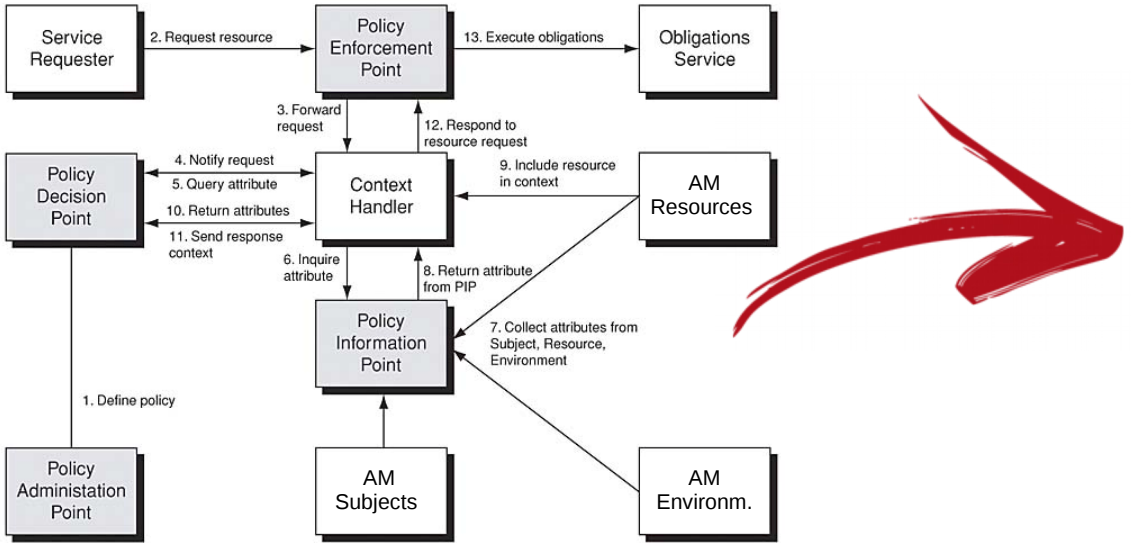
Our Proposal:

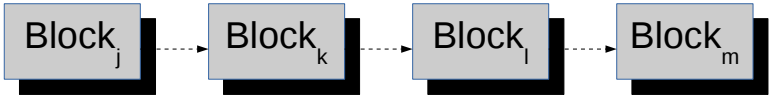
Blockchain based Access Control Service



Our Proposal

Implement a XACML based Access Control Framework exploiting the Blockchain technology

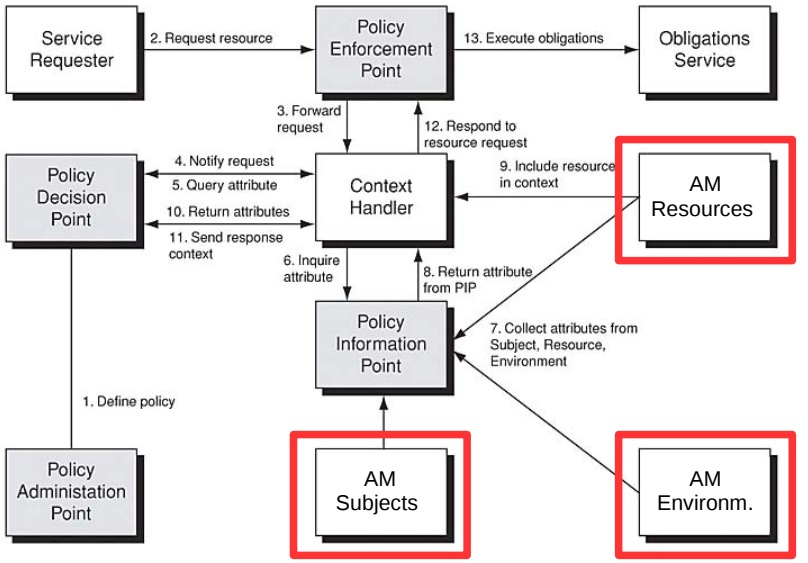




Attributes Management

Attributes Managers (AMs) are implemented as Smart Contracts (**Smart AMs**)

- Smart AMs are stored on the Blockchain
- Each Smart AM has its own address
- Smart AMs provide the interfaces to get the current attribute values and to update them

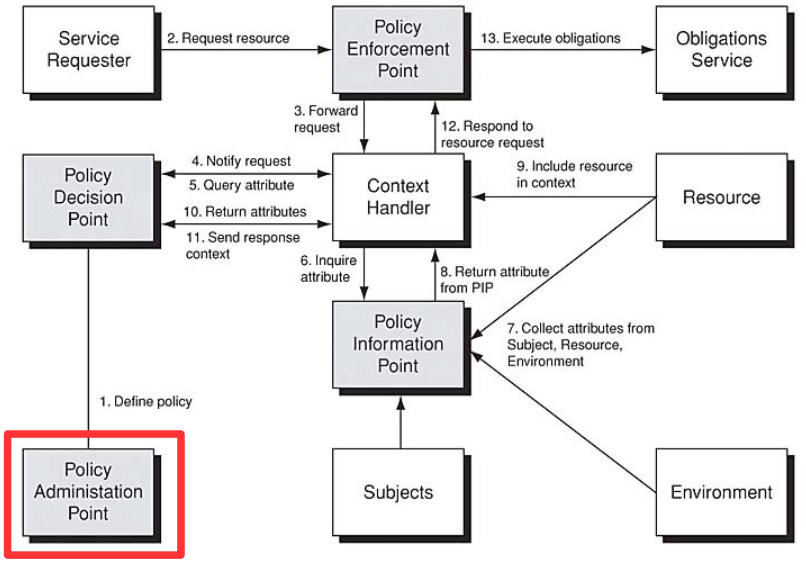


Access Control Policy Representation

Access Control Policies are represented as Smart Contracts (**Smart Policies**)

- XACML policies are translated to Solidity Smart Contracts by a off-chain Parser/Mapper (PAP)
- Smart Policies are stored on the Blockchain (Policy Repository)
- Each Smart Policy has its own address
- The PAP manages a table

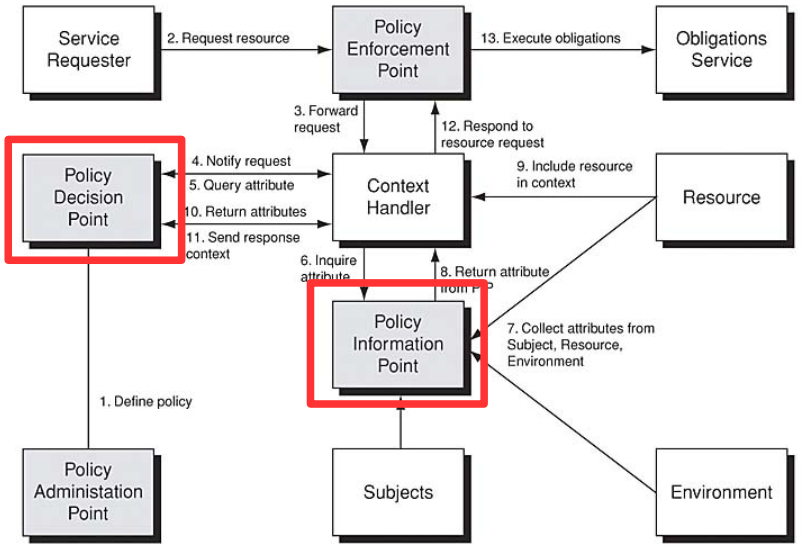
  <Resource ID, Smart Policy Address>



Access Control Policy Representation (II)

- Smart policies are **executable** versions of the XACML polices (PDP)
  - Embed the code to retrieve the Attributes (PIP) from the Smart AMs
  - Embed the code to evaluate the rules
  - Embed the code to combine the results of the rule evaluation
  - They are executed

    “by the Blockchain”



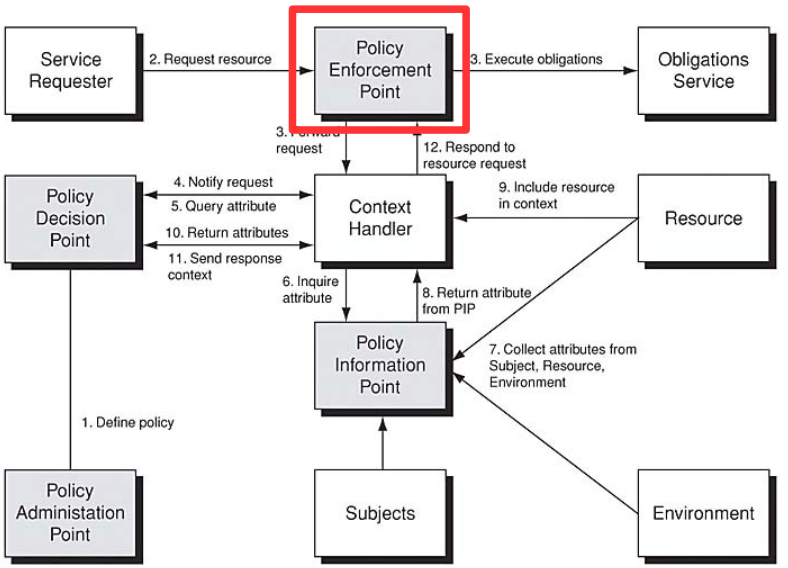
Template of Solidity Smart Policy

```
contract AMContract {
   function get_value() public constant returns (uint32);
}

contract PDP_Policy {
   AMContract private am;
   event Evaluation(
      address indexed _from,
      uint32 _id,
      bool _decision
   );
   address private owner;
   modifier onlyOwner {
        require(msg.sender == owner);
      _;
   }
   function close() public onlyOwner {
      selfdestruct(owner);
   }
   function PDP_Policy() {
      owner = msg.sender;
      am = AMContract(0x56a58e23f8a5efc346910ab0bd950c6ae4333252);
   }
  function evaluate(bytes32 subject, uint32 nonce) returns(bool) {
          attributeValue = am.get_value();
          decision  =  ..... real evaluation of the policy ........
      return decision;
   }
}
```

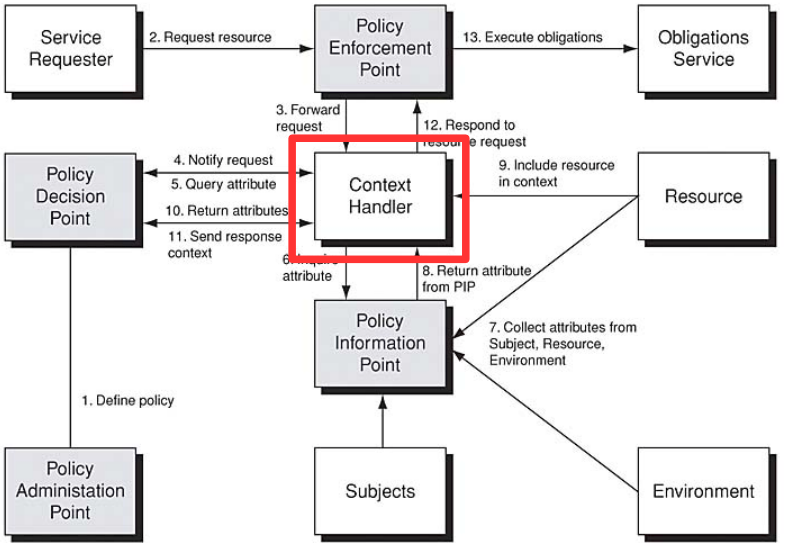
Policy Enforcement Point (PEP)

- Embedded in the code which implements the access to the Resource (off-chain)
- Unaware of the Blockchain. Perfoms its usual tasks:
  - Suspends the execution of the action on the resource
  - Invokes the CH for the evaluation of the Smart Policy
  - Enforces the resulting decision

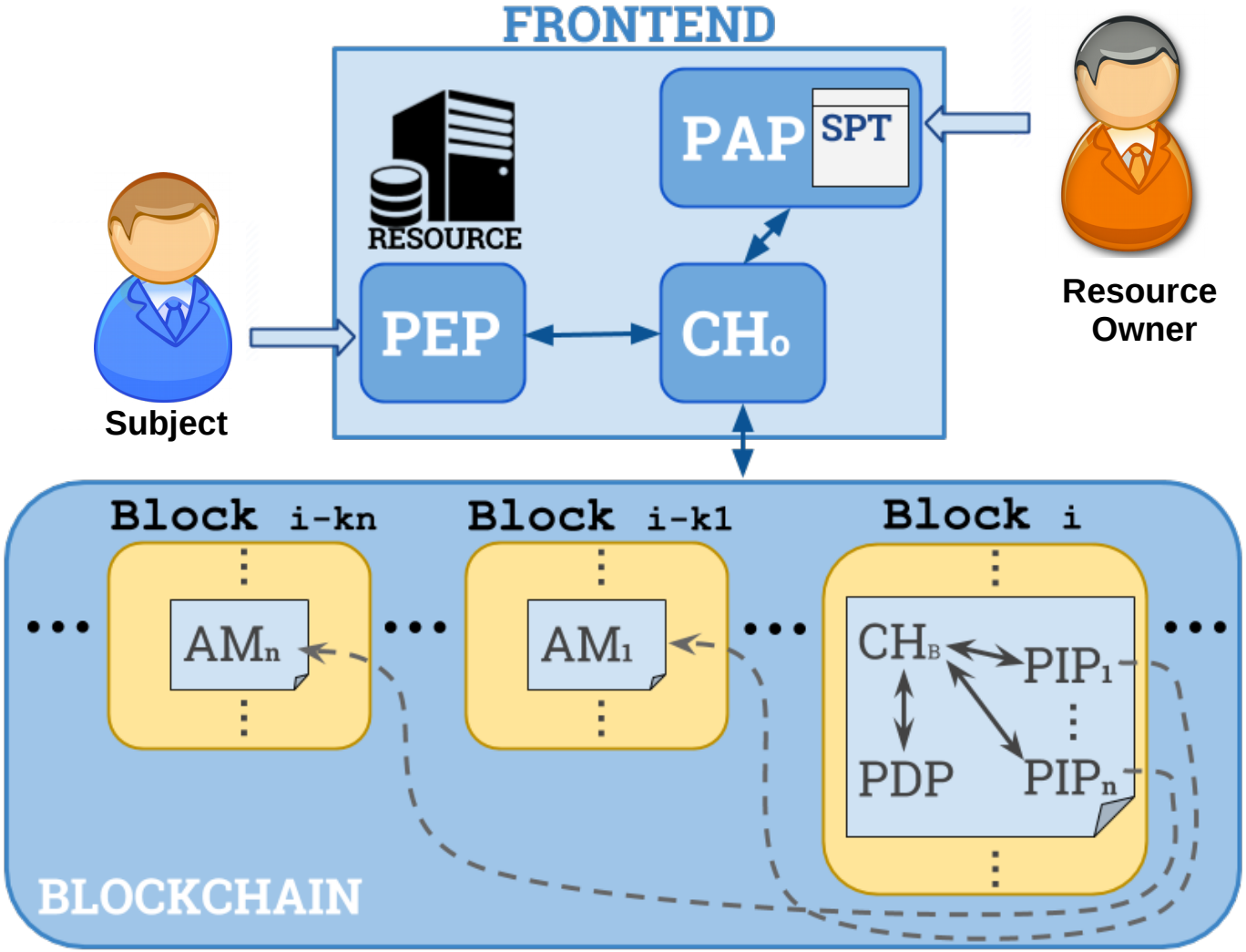


Context Handler (CH)

- Frontend of the Service (off-chain)
- Interacts with the Blockchain:
  - Policy creation:
    - Compiles Solidity code received from the PAP to produce Smart Policies
    - Stores them on the Blockchain and sends to the PAP the resulting address
  - Access Request evaluation:
    - Retrieves from the PAP the address of the Smart Policy to be evaluated for each access request
    - Invokes the Smart Policies for being evaluated

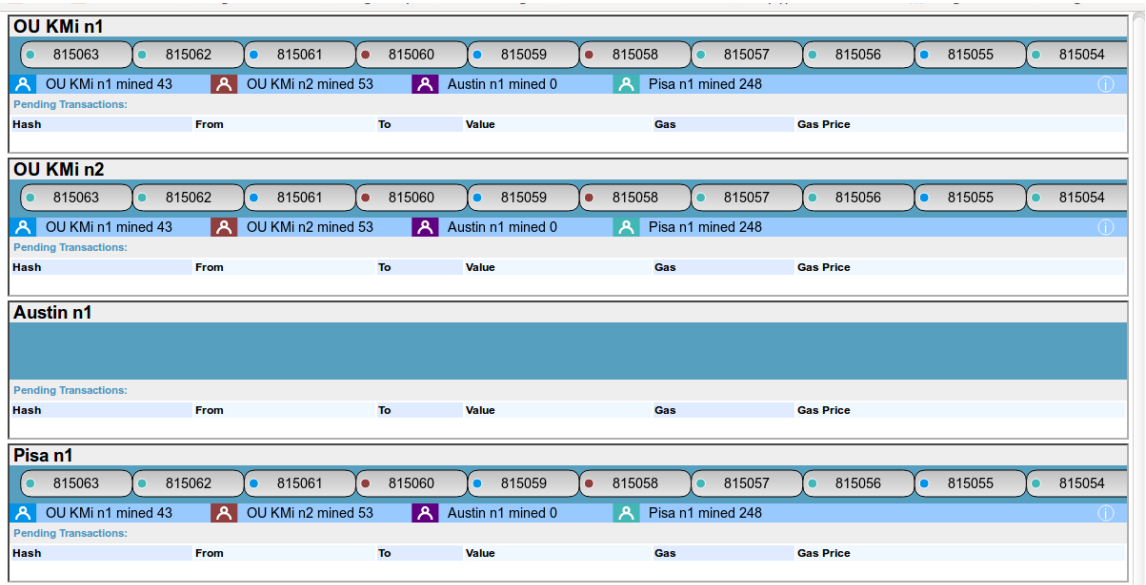


Architecture of the Blockchain based Access Control Service



Some Implementation Details

- International Educational Blockchain academic Testnet (Ethereum based)
  - We are running our own node in Pisa
- PEP: Java + XACML Request/Response
- CH: Java + solc compiler to compile the Solidity code to produce the Smart Policies + web3j to use geth API to interact with Ethereum
- PAP: Java + balana utils for handling XACML
- PDP/PIP: Solidity code



paolo.mori@iit.cnr.it