XACML

eXtensible Access Control Markup Language

Author

Francisco Alexandre de Gouveia

Coordinator

Doutor Diogo Gomes

Collaborator

Engenheiro Ricardo Azevedo

Universidade de Aveiro

Instituto de Telecomunicações

Portugal Telecom Inovação

XACML – What is it?

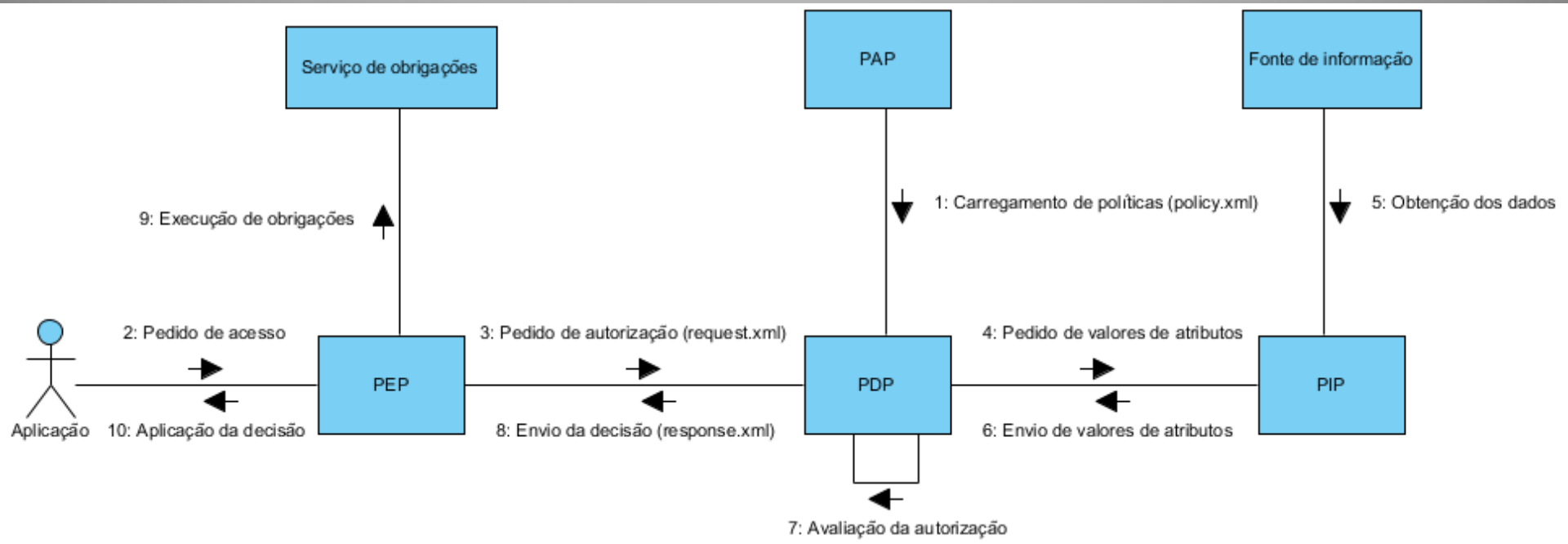
- Standard defined by OASIS for extensible and generic access control
- Consists on:
  - Extensible policy language in XML
  - Extensible request-response language in XML
  - Distributed architecture based on:
    - Policy Enforcement Point
    - Policy Decision Point
    - Policy Information Point
    - Policy Administration Point

XACML – What is it used for?

- Access control
- As it is generic and extensible, can be applied in any context:
  - Door access control
  - Web page access control
  - Service access control
- Only takes decisions!
  - Doesn't tell you which accesses do you have

XACML – How it works?

- Request-response system
- Distributed architecture



XACML – Request

- Can a **subject** make an **action** on the **resource** in some **environment**?
- Keywords:
  - **Subject** – The one who wants to interact
  - **Action** – Kind of interaction
  - **Resource** – Interaction's destiny
  - **Environment** – Anything that cannot be included in the last three categories

XACML – Request

- Can a **subject** make an **action** on the **resource** in some **environment**?

XACMLv2 example

```
<Request>
    <Subject>
        <Attribute AttributeId="user" DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Anonymous</AttributeValue>
        </Attribute>
    </Subject>
    <Action>
        <Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
        DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Read</AttributeValue>
        </Attribute>
    </Action>
    <Resource>
        <Attribute AttributeId="type" DataType="http://www.w3.org/2001/XMLSchema#string">
            <AttributeValue>Topic</AttributeValue>
        </Attribute>
    </Resource>
</Request>
```

Subject

Action

Resource

XACML – Request

- Can a **subject** make an **action** on the **resource** in some **environment**?

```
<Subject>
    <Attribute AttributeId="user" DataType="http://www.w3.org/2001/XMLSchema#string">
        <AttributeValue>Anonymous</AttributeValue>
    </Attribute>
</Subject>
```

XACMLv2 example

```
<xacml:Attributes Category="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
    <xacml:Attribute AttributeId="user" IncludeInResult="false">
        <xacml:AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
            Anonymous
        </xacml:AttributeValue>
    </xacml:Attribute>
</xacml:Attributes>
```

XACMLv3 example

XACML – Response

- Response types:
  - Access granted (Permit)
  - Access denied (Deny)
  - Indeterminate decision (Indeterminate)
  - No policies applicable (Not applicable)
- Together with:
  - Tasks to be run before granting access *

XACML – Response

- * In the 2nd version of XACML, there is an element *Obligation*
- Policy Enforcement Point must do all the requested tasks described in *Obligations*

- *So, what happens when a task is not relevant for the decision but Policy Enforcement Point cannot do it?*

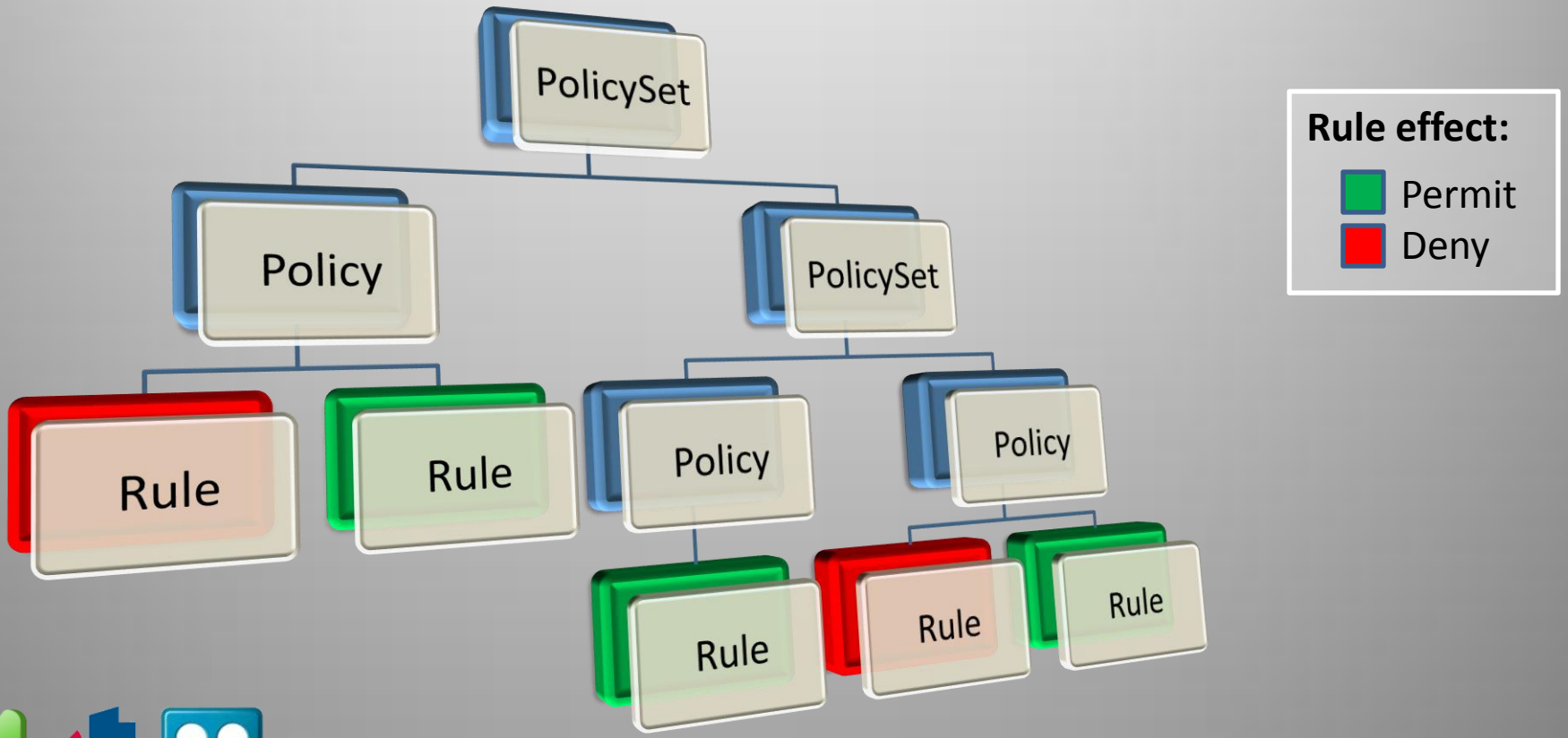
XACML – Response

- * In the 3rd version of XACML, there are *Obligation* and *Advice* elements
- Policy Enforcement Point:
  - **Must** run all the tasks described in *Obligations*
  - **Should** run all the tasks described in *Advices*

- If PEP fails to do a task from Advice elements, the final decision is not changed

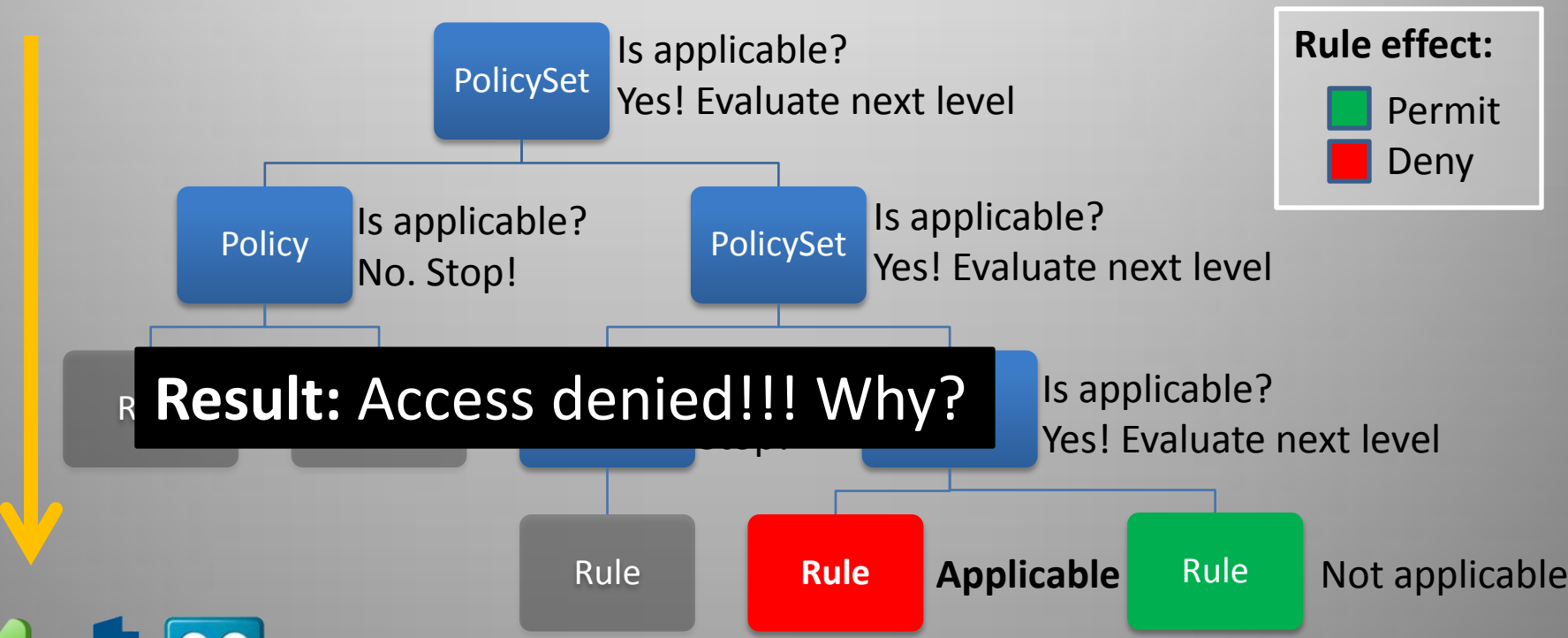
XACML – Evaluation

- How are policies structured?
- There are 3 main elements



XACML – Evaluation

- How are elements evaluated?
- Each element has a "*Target*"
- Evaluation is made from the top

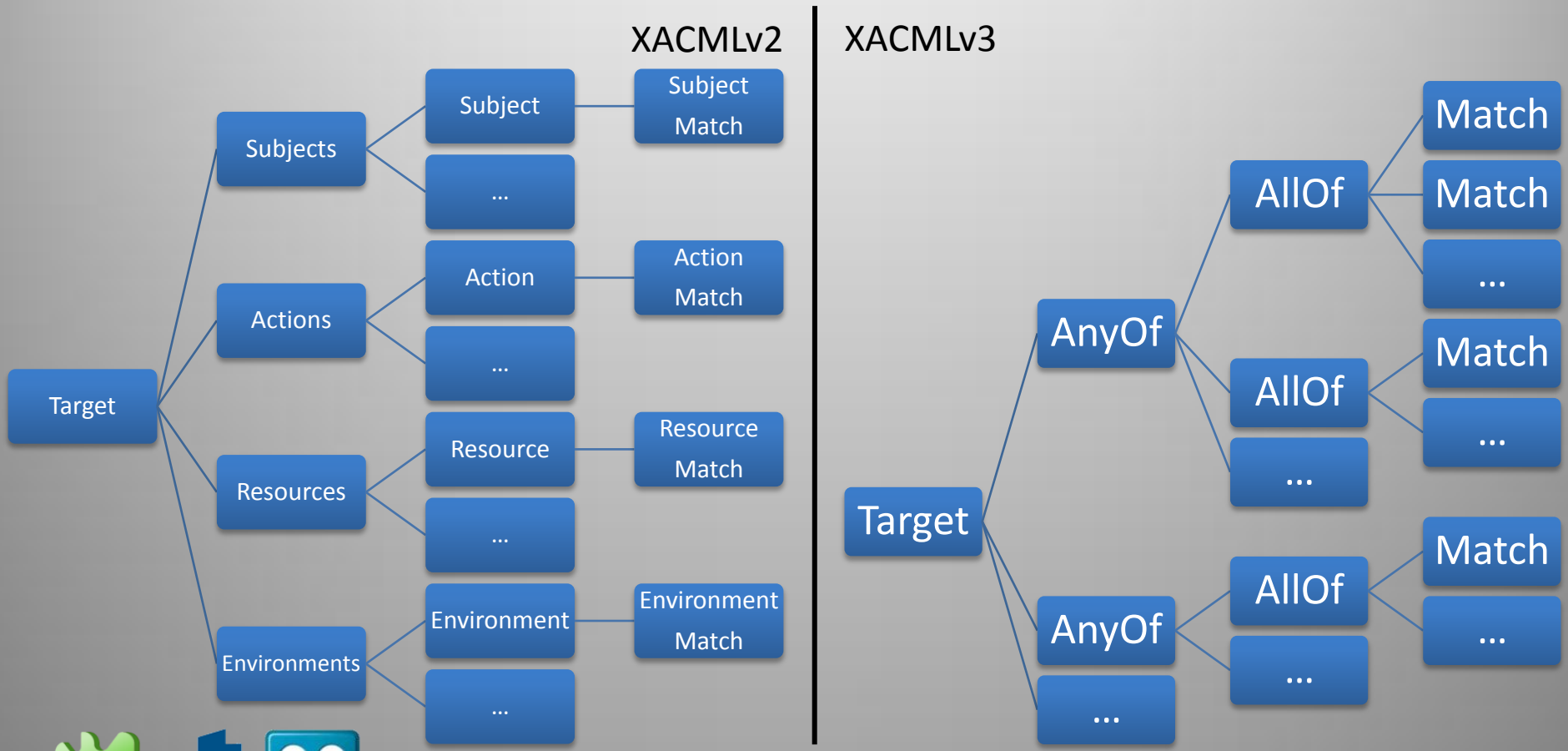


XACML – Evaluation

- What if more than one rule or policy is applicable?
- Answer: Combining algorithms
  - Permit-overrides
  - Deny-overrides
  - Only-one-applicable
  - First-applicable
  - … (more can be created)

XACML – Evaluation

- How is a *Target* element made?



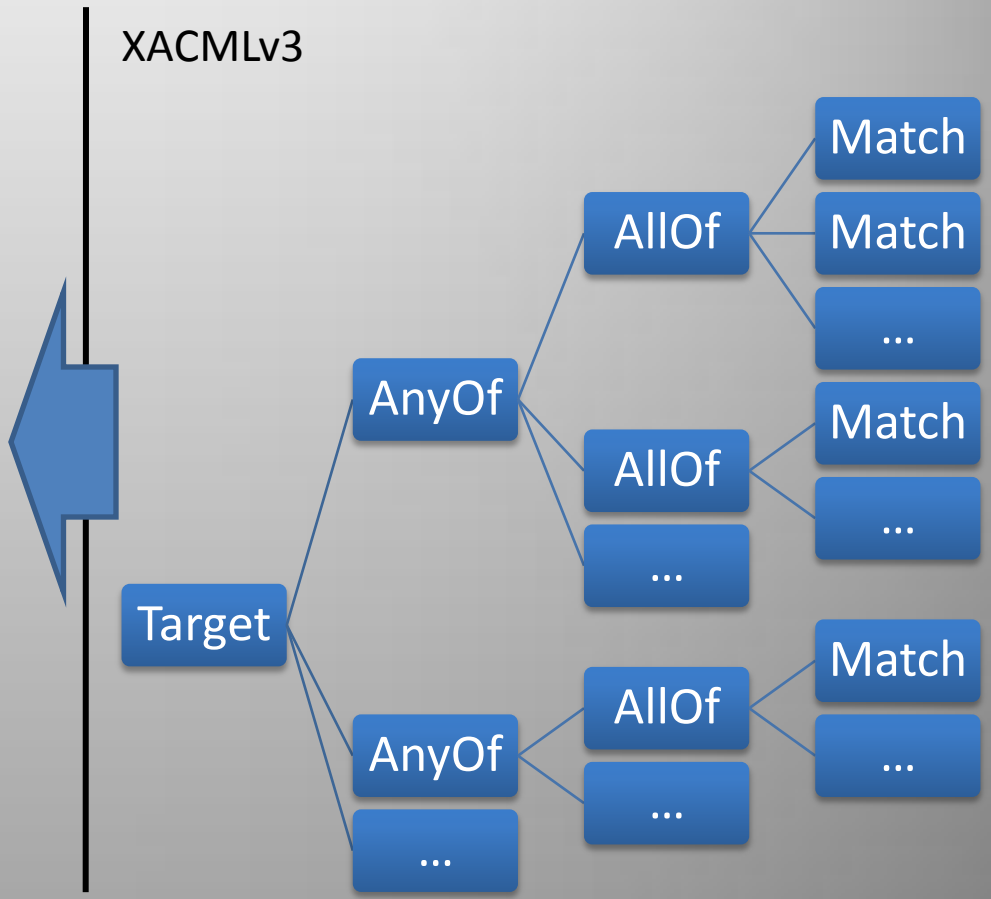
XACML – Evaluation

- How is a *Target* element made?

•Uniform model for all the categories
•Allows to define intersections and unions of Matches

But…

•AnyOf and AllOf elements are not identified. Managing policies with such elements turns out to be a problem that can be solved by:
  - •Analysing deeply element values to know where to modify
  - •Or recreating the Target element each time a change is made



PAP XACMLv3

Policy Administration Point

Author

Francisco Alexandre de Gouveia

Coordinator

Doutor Diogo Gomes

Collaborator

Engenheiro Ricardo Azevedo

Universidade de Aveiro
Instituto de Telecomunicações
Portugal Telecom Inovação

Policy Administration Point

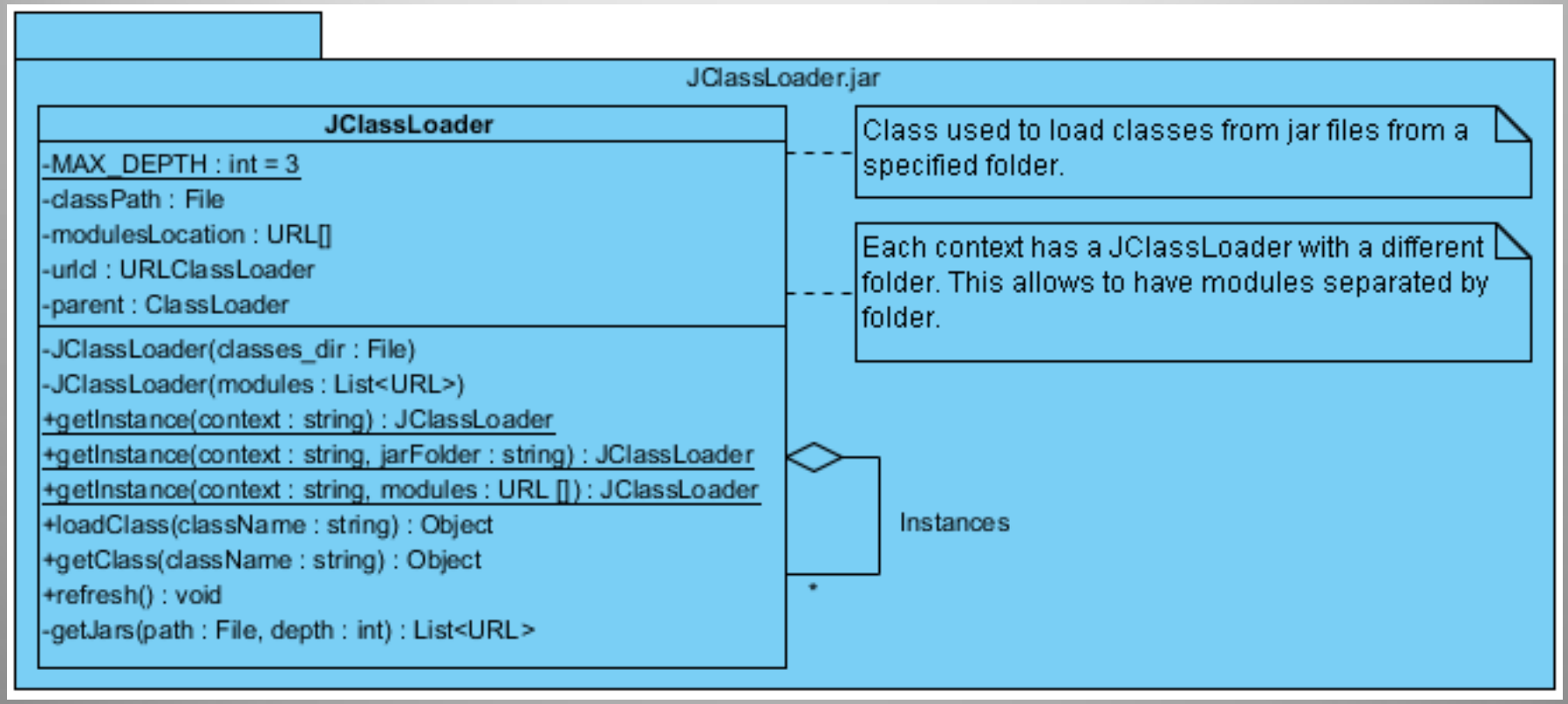
- Project objectives
  - Extensible information system (able to import modules without recompilation)
  - User interface that abstracts the complexity of XACMLv3
  - Creation of policies that respect the standard

Policy Administration Point

- Extensibility
  - Allows importation of classes implementing the defined interfaces for:
    - Retrieving information from Policy Information Point
    - Persistence and retrieving policies
- Used solution:
  - URLClassLoader - loads classes from **.jar* files in run-time

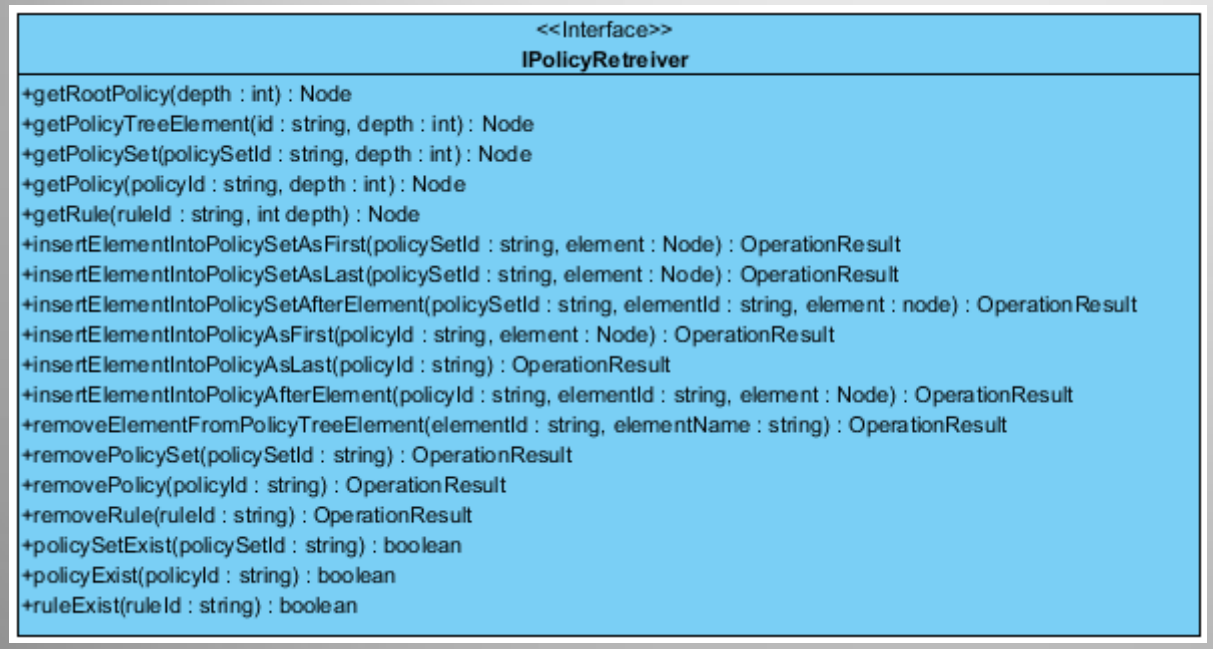
Policy Administration Point

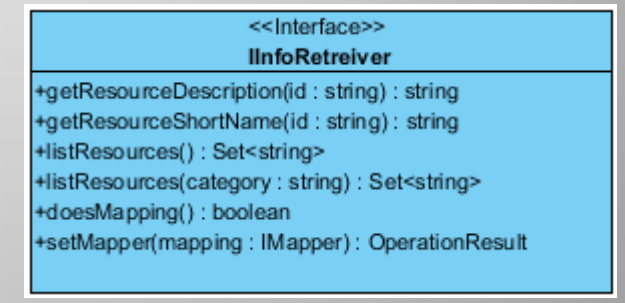
- Extensibility (Class loader)



Policy Administration Point

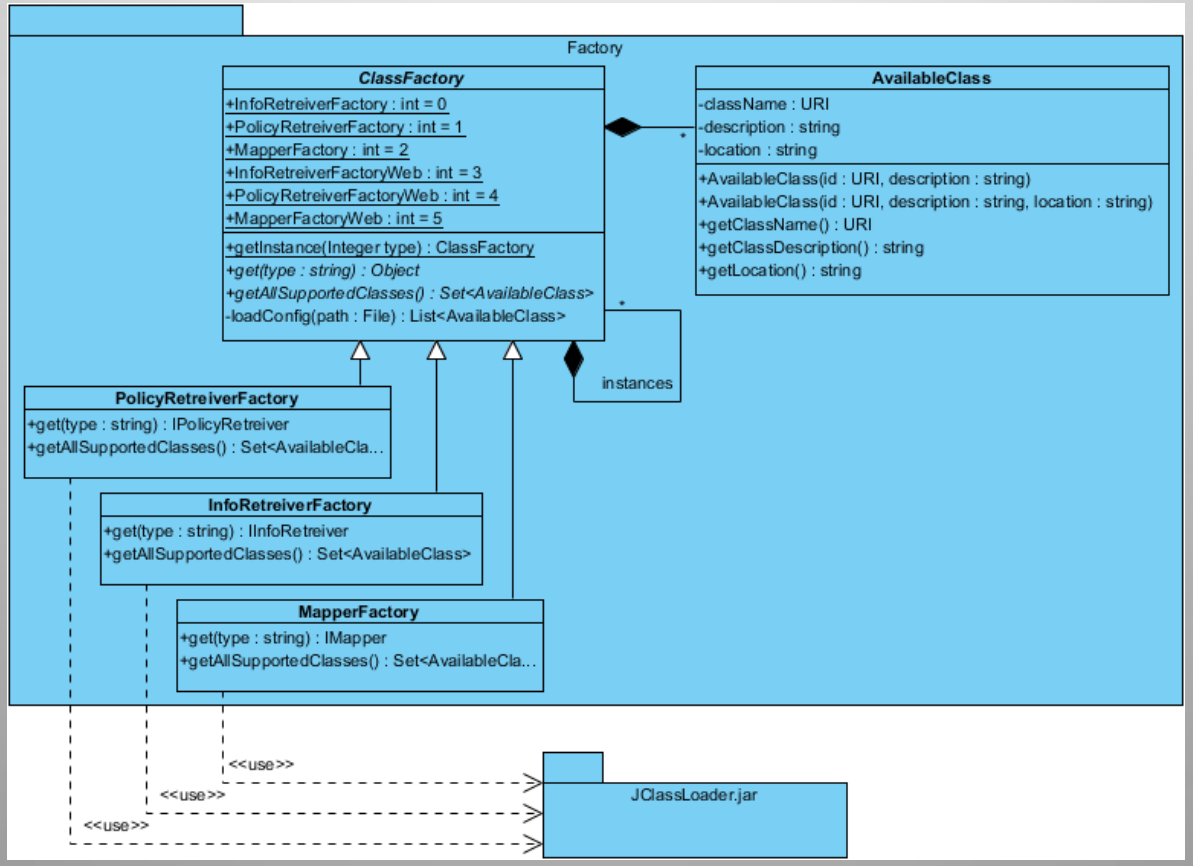
- Extensibility (Interfaces)





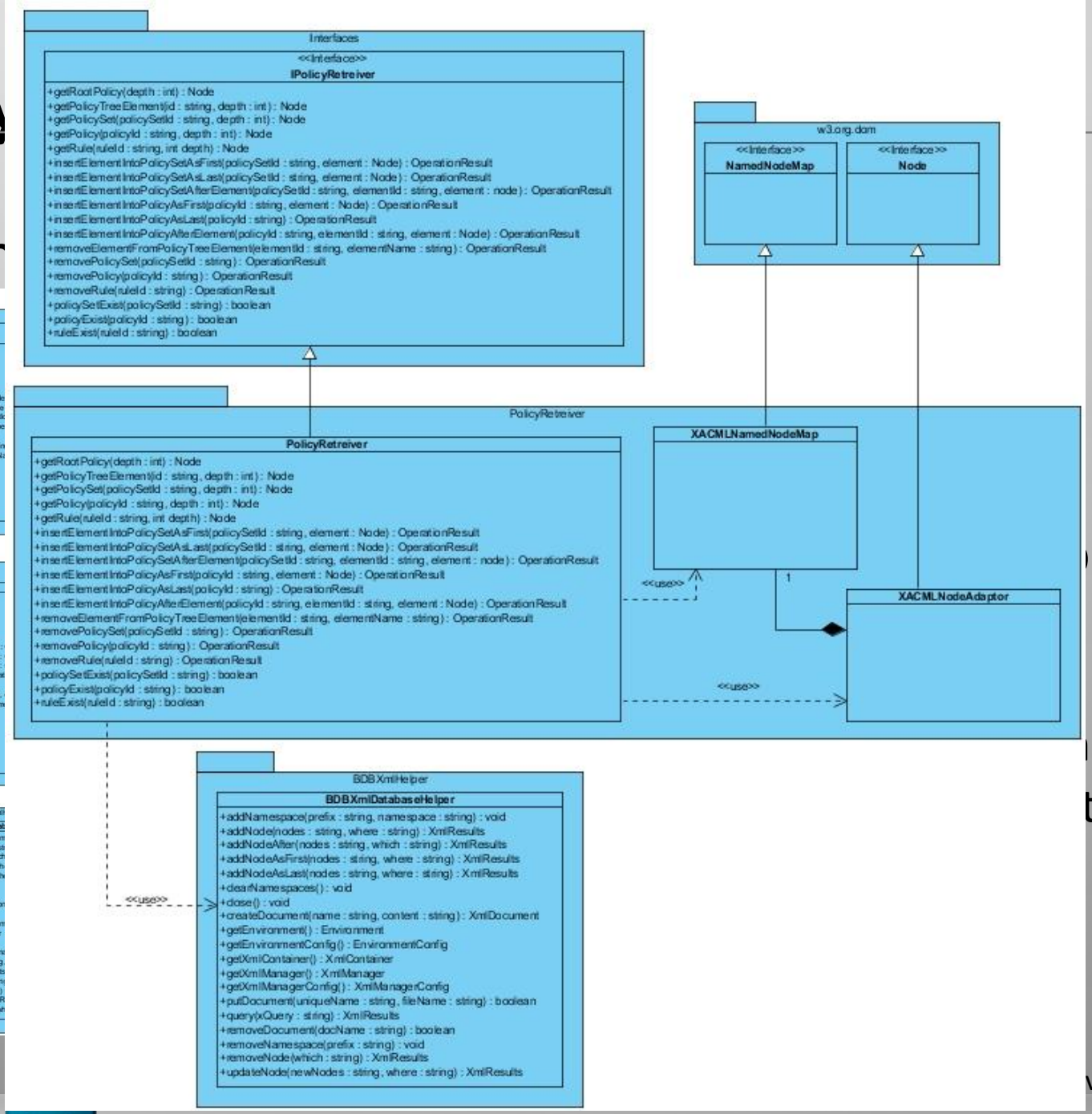
Policy Administration Point

- Extensibility (Factory)



Policy

- Exter



B XML

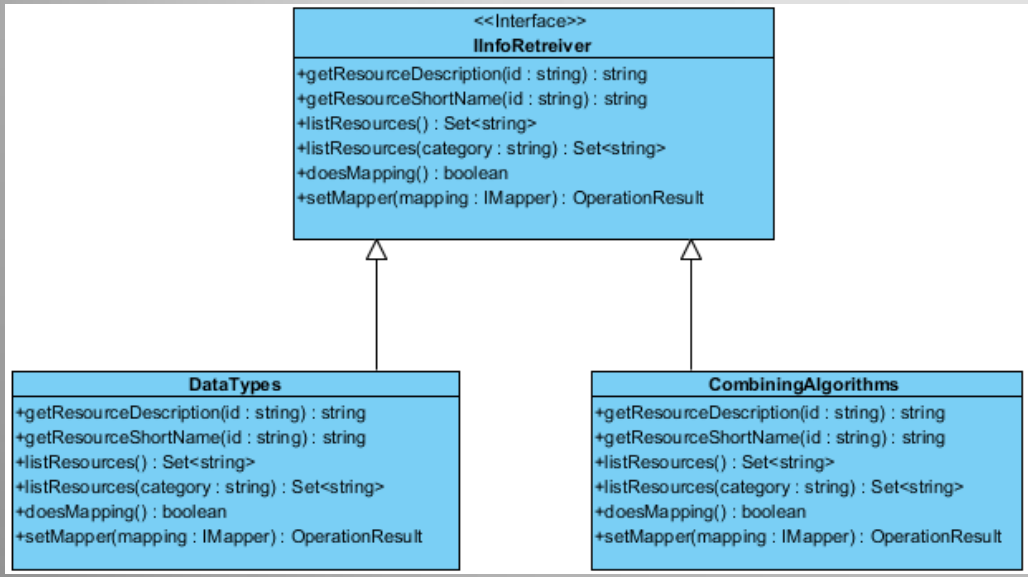
dards

tions

vação

Policy Administration Point

- Extensibility (Implemented modules)



- Info Retriever
  - Reads xml file
- Returns:
  - Data types
  - Functions
  - Combining algorithms

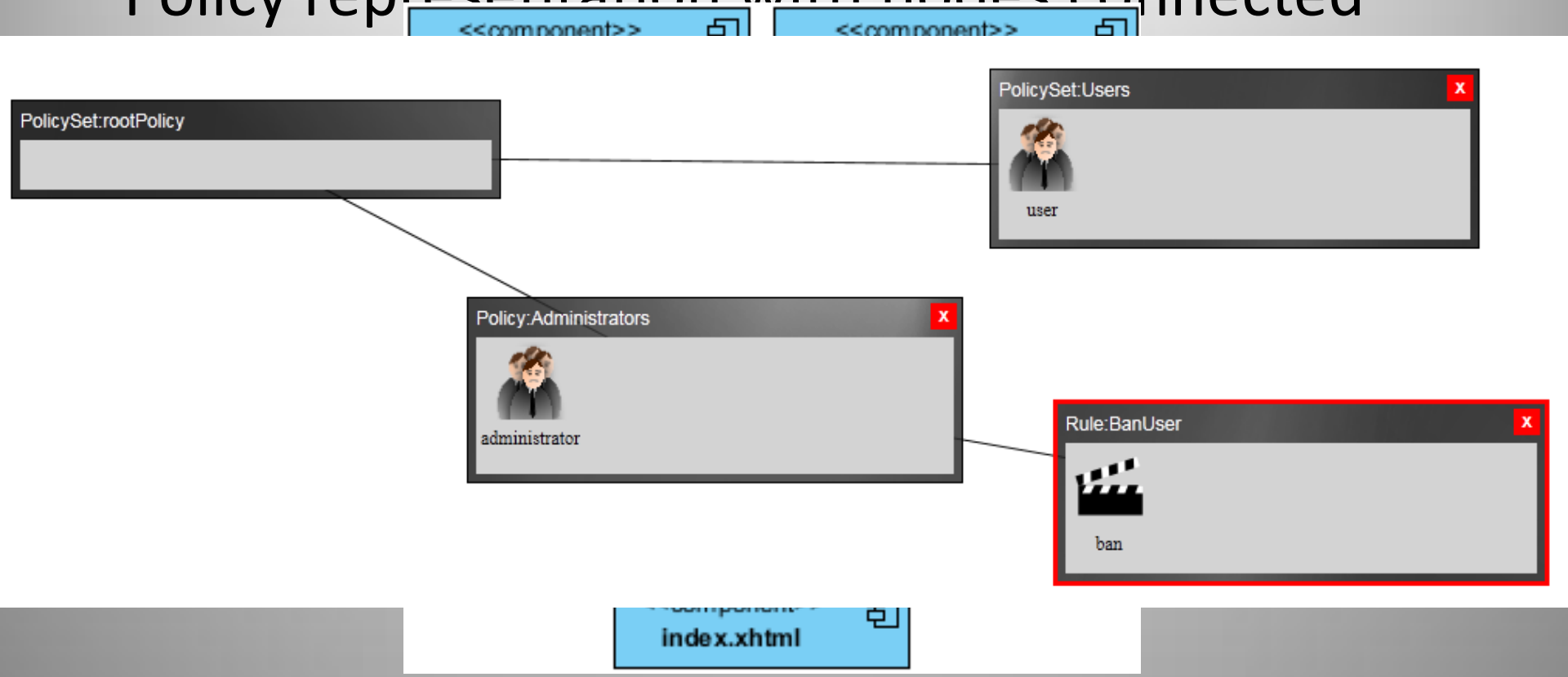
Policy Administration Point

- XACML complexity abstraction on the user interface

- Used solution:
  - Web interface with:
    - Policy representation in nodes with connections between them
    - Complex name abstractions with images and simplified names
    - XACML standard abstraction, showing only the possible options per element

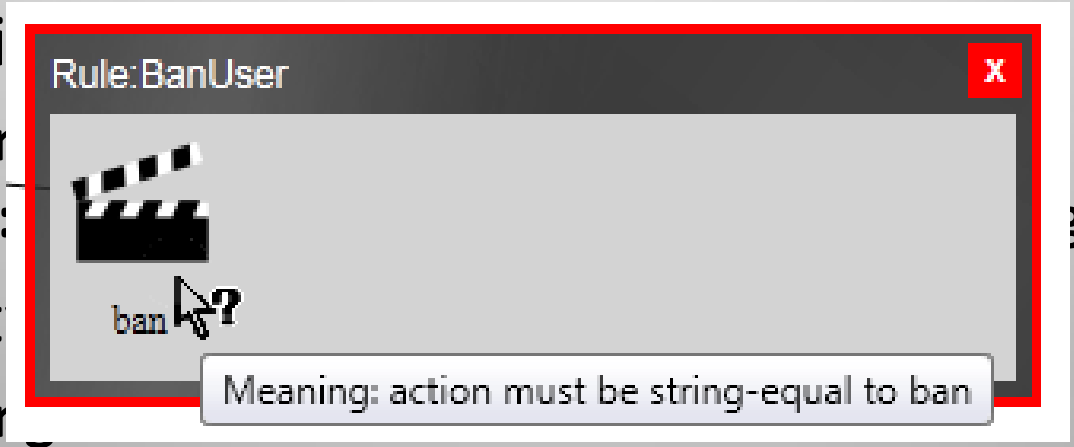
Policy Administration Point

- Policy representation with nodes connected



Policy Administration Point

- Complex names abstraction with images and simplified names
  - Functi
    - **strin**
      urn: **g-equal**
  - Data t
    - **strin**
      http://www.w3.org/2001/XMLSchema#string



Policy Administration Point

- Creation of policies that obey to standard rules
  - Behind all abstraction, there is an effort to save the policies in respect to the standard
  - Options change in relation to the element selected, accordingly to the possibilities of this element
    - E.g.: There is a toolbox whose buttons are displayed depending on the element selected

Policy Administration Point

- Implementation: J2EE (problems occurred)
  - ClassLoader didn't work as expected when hosted in an application server
  - Single instance classes weren't single instance

Policy Administration Point

- Java Class Loader
  - Works hierarchically

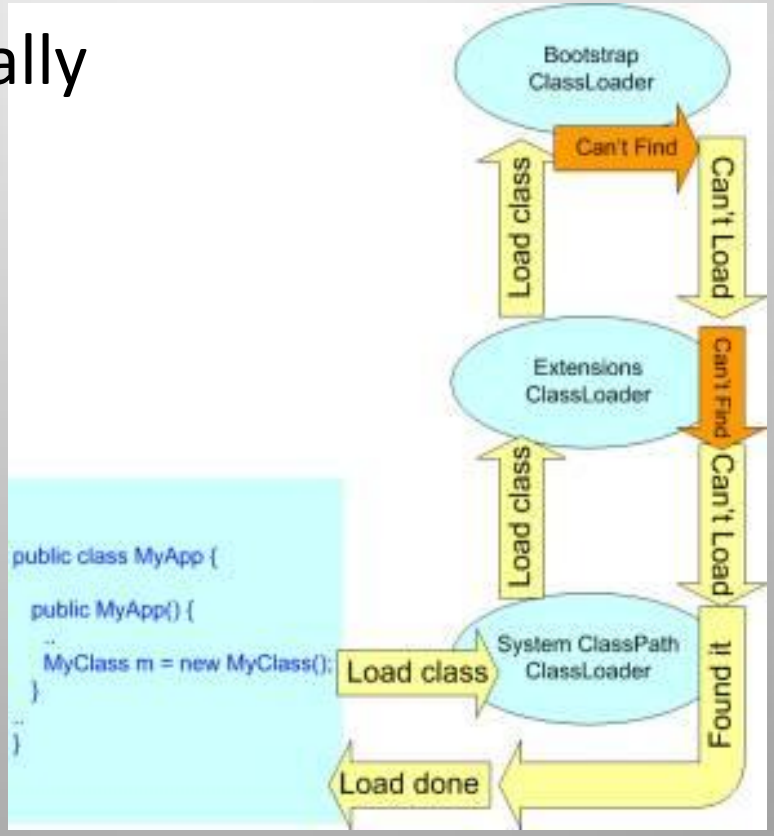


Image taken from http://www.objectsource.com/j2eechapters/Ch21-ClassLoaders_and_J2EE.htm

Policy Administration Point

- Web server instances
  - Application is stored in more than one container
  - For performance purposes, load is balanced between containers

- For single instance classes, a Session Bean Singleton was used

Policy Administration Point

- Conclusions
  - XACML allows you to make fine-grained and generic access control
  - 3rd version brought improvements in relation to the 2nd version
    - Added advice element
    - Unions and intersections of targets
    - Multi-request
    - But it's still a draft...
  - There are not many implementations
  - Hard to administrate Targets

Policy Administration Point

Questions?