

MCeTech 2009  
Ottawa, Canada



uOttawa

L'Université canadienne  
Canada's university

# A Non-Technical User-Oriented Display Notation for XACML Conditions

by Bernard Stepien, Amy Felty,  
Stan Matwin

School of Information Technology and  
Engineering

Université d'Ottawa | University of Ottawa



[www.uOttawa.ca](http://www.uOttawa.ca)

# Motivation

- The XACML XML-based language is very precise and allows fine-grained access control policy specification with complex conditions.
- However, it is out of reach for non-technical users for the following reasons:
  - Long XML tags
  - Long and complex domain references
  - Prefix notation for operations
  - List oriented notation for conjunction and disjunction operators

# Example XACML specification

Merchandise is food and DayOfTheWeek is Monday

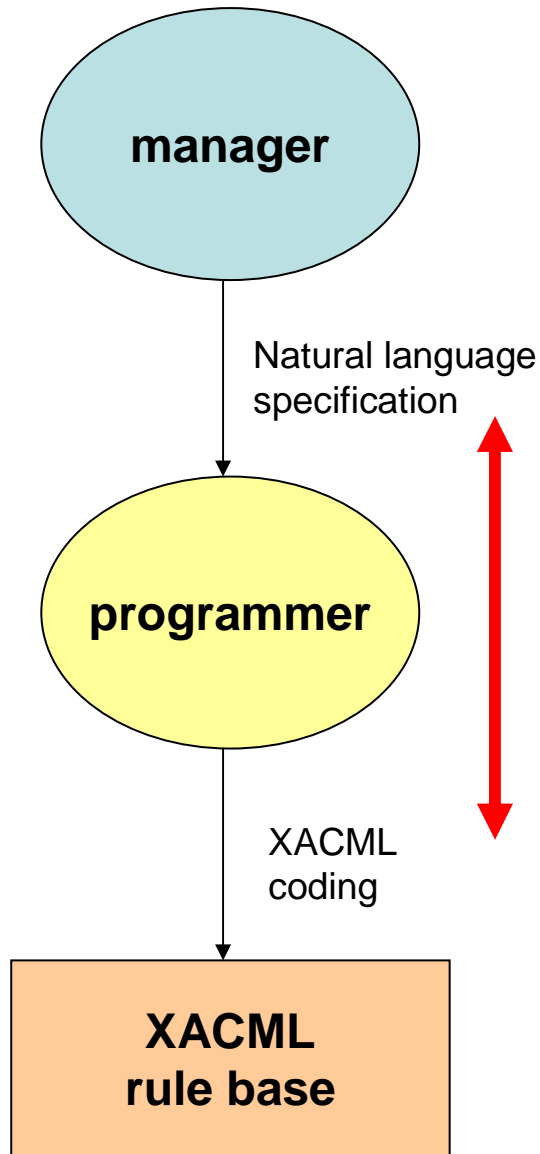
```
<Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:and">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
    <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
      <SubjectAttributeDesignator AttributeId="Merchandise"
        DataType="http://www.w3.org/2001/XMLSchema#string" />
    </Apply>
    <AttributeValue
      DataType="http://www.w3.org/2001/XMLSchema#string">food</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">
      <SubjectAttributeDesignator AttributeId="DayOfTheWeek"
        DataType="http://www.w3.org/2001/XMLSchema#string" />
    </Apply>
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string"
      >Monday</AttributeValue>
  </Apply>
</Condition>
```

# Non-technical users application examples

- Controlled cell-phones use:
  - Restrict the phone numbers that can be called
  - Restrict the hours or days where calls can be placed
  - Restrict roaming zones
- Controlled credit-card use
  - Restrict the merchandise that can be purchased.
  - Restrict the hours and days of use
  - Restrict the stores where is can be used
  - Restrict the geographic zones of use

# Access control management

## traditional approach

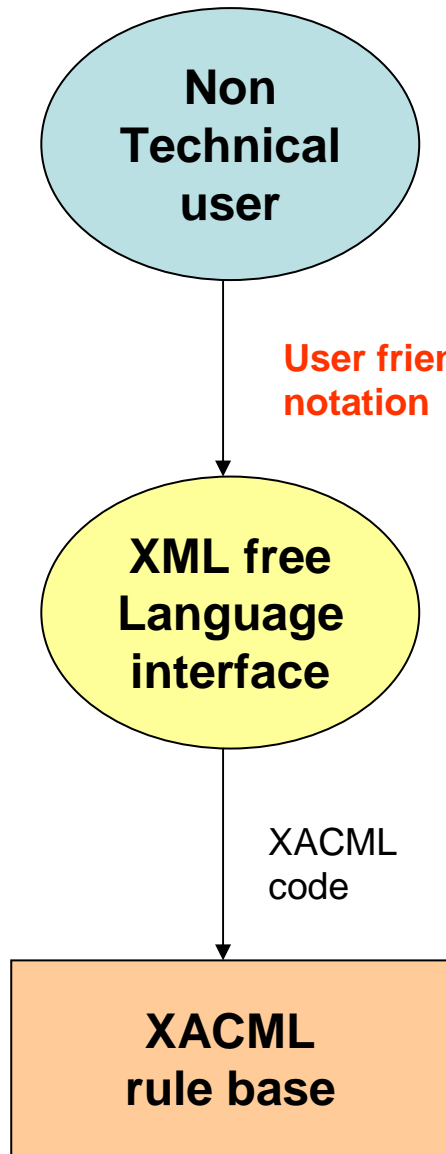


1. Interpretation errors
2. Coding errors
3. Verification barrier

- Who controls?
- Who implements?

# Access control management

## user-centric approach



- Eliminates the programmer.
- Increases user's confidence.
- Does not eliminate XACML

# Viewing or editing a XACML specification

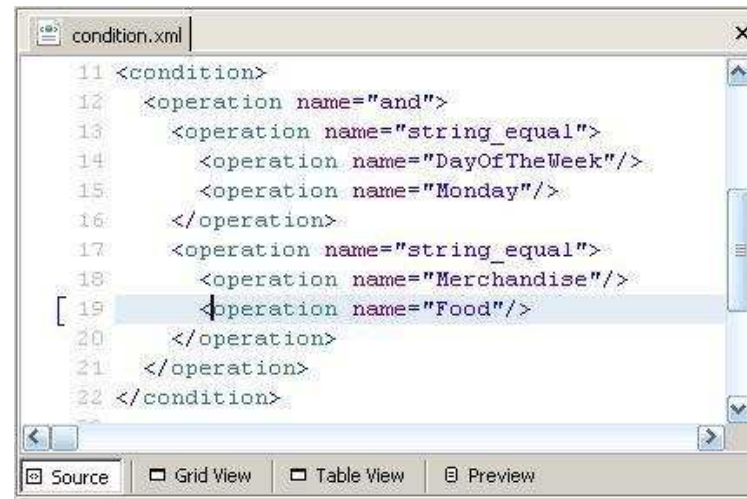
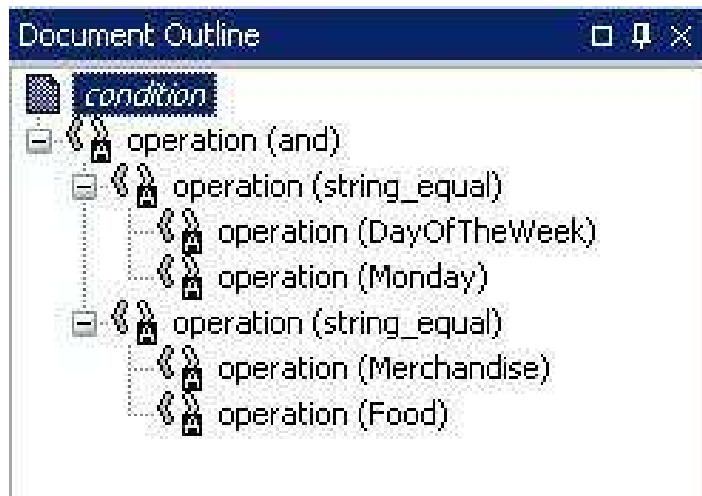
current state

- XACML specifications can be created or edited with two classes of tools:
  - Generic XML editors
  - Specific XACML editors

# Generic XML editors

## basic tree notations

- Representing a simple logical expression in XML using XML-pad





# Generic XML editors

## graphic tree notations

Node	Value
xml	version="1.0" encoding="utf-8"
DOCTYPE	#inline
condition	(operation)
operation	(operation)*
@name	and
operation	(operation)*
@name	string_equal
operation	(operation)*
@name	DayOfTheWeek
operation	(operation)*
@name	Monday
operation	(operation)*
@name	string_equal
operation	(operation)*
@name	Merchandise
operation	(operation)*
@name	Food

xml	version="1.0" encoding="utf-8"
condition	
operation	
name	and
operation (2)	
name	operation
1	string_equal
operation (2)	
name	
1	DayOfTheWeek
2	Monday
2	string_equal
operation (2)	
name	
1	Merchandise
2	Food

# Specific XACML editors

## UMU XACML editor

- XML tags or domains from pull-down menus

UMU-XACML-Editor - policy\_example.xml

File Schema Validator About

Policy Document

- <Policy : SamplePolicy>
  - <Target>
  - <Rule : LoginRule>
    - <Target>
    - <Condition>
      - <Apply : urn:oasis:names:tc:xacml:1.0:function:time-greater-than-or-equal/>
      - <Apply : urn:oasis:names:tc:xacml:1.0:function:time-one-and-only/>
      - <AttributeValue/>
    - <Apply : urn:oasis:names:tc:xacml:1.0:function:time-one-and-only/>
  - <Rule : FinalRule/>

\* FunctionId: urn:oasis:names:tc:xacml:1.0:function:time-greater-than-or-equal

\* Required

List of Applies

<Apply : urn:oasis:names:tc:xacml:1.0:function:time-one-and-only/>

Analizando fichero:C:\BSI\_Projects\ACCEPT\XACML\policy\_example.xml

<AnySubject> Label nonrecognized

<AnyAction> Label nonrecognized

<AnySubject> Label nonrecognized

<AnyResource> Label nonrecognized

<EnvironmentAttributeSelector> Label nonrecognized

<EnvironmentAttributeSelector> Label nonrecognized

# Specific XACML editors

## XACML studio

The screenshot shows the XACML Studio interface within a Microsoft Internet Explorer browser window. The browser's address bar displays `http://localhost:3000/xs/index.html`. The main content area is divided into three panes: Policy Repository, Properties, and Information.

**Policy Repository:** A tree view showing a hierarchy of XACML objects. The selected node is `SubjectMatch(==)` under `Subject` of a `Rule` within a `Target` of a `Policy`. Other nodes include `AttrVal(J. Hibbert)` and `SubjAttrDesignator`.

**Properties:** A table showing the properties of the selected `SubjectMatch(==)` node. The `MatchId` dropdown menu is open, showing options like `match` and `regexp-match`.

Name	Value
created_at	2008/10/28 14:28:12 -0700
id	66
MatchArgType	string
MatchId	[Dropdown menu]
subject_id	==
updated_at	

**Information:** Displays the text "Policy for Conformance Test IIA002."

# Drawbacks of XML or XACML editors

- They all require knowledge of XACML syntax and semantics.
- They all require programming skills.
- They do not reduce the amount of information a user must process.

# Principles of our notation

- Stay as close as possible to the user's natural language:
  - by avoiding any technical terminology for operators
  - by maintaining the overall structure of a natural language.
- Tree representation as an implicit structuring paradigm
- Use infix representation for conjunction and disjunction operators as with natural languages.
- Maintain XACML's natural non-binary nature of conjunction and disjunction operators but eliminate its original list representation.
- Use a casual terminology for conjunction and disjunction operators depending on their position in the tree hierarchy.
- Ensure a full graphical overview of the expression being built at all times regardless of its complexity.

# User-centric notation

## Natural language rule:

*“It is permitted to purchase food on a Monday or a Tuesday or travel on a Friday provided that the balance of the account is over 500”.*

## A typical pseudo-code mathematical like translation:

((Merchandise == food and (day == Monday or day == Tuesday))  
or (Merchandise == travel and day == Friday)) and balanceOfAccount > 500

## Translated in our notation:

Merchandise is one of  
    Food  
    provided that DayOfTheWeek is one of Monday, Tuesday  
Travel  
    provided that DayOfTheWeek is Friday  
and  
BalanceOfAccount over 500

# Benefits of the notation

- The XACML rule could have been translated into straight natural language.
- But natural languages are inherently ambiguous.
- Example: it is not clear if the balance of account > \$ 500 applies to food or travel purchases or even on Fridays only.
- The tree representation eliminates any ambiguity and also the need for parentheses.
- Natural language is difficult to edit and transform into formal languages (XACML or others)

# Four components of our notation

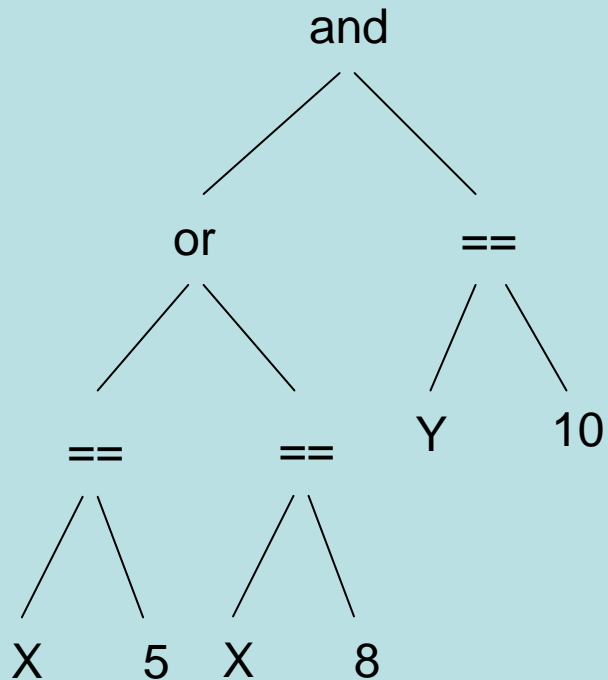
- Non binary tree representation
- horizontal tree representation
- Indentation instead of parentheses
- Variable factorization

In fact the horizontal tree representation is nothing new, it is directly derived from the structure of a XML document.



# Tree concept comparison

## Traditional vertical binary tree



## Natural language representation

X is equal to 5 or to 8 and Y is equal to 10

## Our horizontal tree notation

X is one of 5, 8  
and  
Y is 10

# List representation

- The XACML list concept for conjunction and disjunction operators is maintained but by repeating the operators between elements.

X is 3  
and  
Y is 15  
and  
Z is 34

X is 6  
or  
X is 45  
or  
X is 27

# Sub-constraint terminology

- Conjunction and disjunction operators representation vary depending on their location in the hierarchy
- “and”, “or” notation at the highest level
- “Provided that”, “one of” at lower levels

X is 3  
    provided that K is 4  
and  
    Y one of 15, 24, 46  
and  
    Z is one of  
        blue  
            provided that L is 5  
        red  
            provided that L is 10

# Complex rule example

Merchandise is one of  
clothing

provided that BalanceOfAccount over 2000.00,  
concert

provided that DistanceFromHome under 5

and

DayOfTheWeek is one of  
Tuesday

provided that TimeOfTheDay between 16:00:00 and 18:00:00,  
Wednesday

provided that TimeOfTheDay between 12:00:00 and 14:00:00,  
Friday

provided that TimeOfTheDay before 21:00:00

# Generic notation

- The notation is not limited to the representation of XACML
- Any condition expression from any Access control language can be represented.

Cisco firewall rule

```
access-list 101 deny tcp host 148.22.33.44 host 192.168.0.0 eq 3500
```

```
protocol is tcp  
and  
srcIP is 148.22.33.44  
and  
dstIP is 192.168.0.0  
and  
dstPort is 3500
```

# Application configuration

- The need for the programmer is not fully eliminated.
- The programmer still builds the application.
- Application provider administrates:
  - Data types
  - Enumerated values

```
<Variable name="DayOfTheWeek"
           type="String">
  <Values>
    <Value name="Monday"/>
    <Value name="Tuesday"/>
    <Value name="Wednesday"/>
    <Value name="Thursday"/>
    <Value name="Friday"/>
    <Value name="Saturday"/>
    <Value name="Sunday"/>
  </Values>
</Variable>
```

# Our notation and editors

## Rule condition manipulation



## Value selection



# Conclusion

- With our display notation, anyone can now create and edit XACML rules
- Thus, our notation could result in a larger audience for XACML.



# Contact

- Bernard Stepien – [bernard@site.uottawa.ca](mailto:bernard@site.uottawa.ca)
- Amy Felty – [afelty@site.uottawa.ca](mailto:afelty@site.uottawa.ca)
- Stan Matwin – [stan@site.uottawa.ca](mailto:stan@site.uottawa.ca)