Open Geospatial Consortium Inc.

Date: 2008-02-20

Reference number of this OGC® project document: **OGC 07-026r2**

Version: 1.0

Category: OGC® Implementation Standard

Editor: Andreas Matheus

Co-Editor: Jan Herrmann

Geospatial eXtensible Access Control Markup Language (GeoXACML)

Document type: OGC® Publicly Available Standard
Document subtype: Encoding
Document stage: Approved
Document language: English

Disclaimer

Issues of absolute or relative accuracy of feature geometry and positioning data and the computational stability of finite precision arithmetic as used in all computers will affect results where distance measure, geometry or positional values are compared. For this reason, and the variety of statistically valid but different implementation approaches to these issues, Boolean criteria as used in XACML or GeoXACML policy statements will not always produce uniform results across geographic implementations. For these reasons, users of this technology should not ask for fine gradations of measures that their data or coordinate transformation cannot support, but should alternately allow a "tolerant" approach where feasible which might give access to slightly more data than would be given in a perfect computational, but impossible to implement, environment.

Contents

i. Preface

The Policy Language introduced in this document defines a geo-specific extension to the OASIS standard "eXtensible Access Control Markup Language (XACML), Version 2.0" [1]. Due to the similarity of XACML version 2.0 to previous versions (e.g. 1.0 or 1.1), the extension described in this document also represents a valid extension for earlier XACML versions.

GeoXACML defines an extension to XACML for spatial data types and spatial authorization decision functions. Those data types and functions can be used to define additional spatial constraints for XACML based policies.

Beside a schema to encode policies, XACML includes a context schema that includes a specification of the generic data level interface to the XACML processor (PDP). Since GeoXACML defines extensions for the policy encoding schema only, it does not affect the XACML context schema and therefore does not include an interface specification of any kind.

In addition, XACML includes a model for an access control system. This incorporates stereotype definitions of a Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Administration Point (PAP) and Policy Information Point (PIP) as well as their relations to each other in the context of an access control system. For the purpose of GeoXACML, the access control model is informative.

Attention is drawn to the fact that this document defines a rule based Policy Language suitable to express access rights. By using the GeoXACML Policy Language, an interoperable access control system for geospatial applications, such as Spatial Data Infrastructures, can be implemented. It is important to highlight that GeoXACML is not designed to be a Rights Expression Language.

Therefore, this standard is not an extension of the OGC GeoDRM Reference Model (The OpenGIS® Abstract Specification Topic 18: Geospatial Digital Rights Management Reference Model [9]). However, this geo-specific extension to the existing OASIS standard can be seen as the baseline for the development of an implementation specification in the context of [9].

Submitting organizations

The following organizations submitted this Implementation Specification to the Open Geospatial Consortium Inc. as a Request For Comment (RFC):

Universität der Bundeswehr München
Andreas Matheus
Werner-Heisenberg-Weg 39
D-85579 Neubiberg
Germany
andreas.matheus(at)unibw.de

Galdos Systems Inc.
Ron Lake
1300-409 Granville St
Vancouver V6C 1T2
Canada
Rlake(at)galdosinc.com

ii. Submission and contribution contact points

All questions regarding this submission should be directed to the editor, the submitters or the contributors:

CONTACT	COMPANY	EMAIL
Andreas Matheus	Universität der Bundeswehr München	Andreas.Matheus(at)unibw.de
Ron Lake	Galdos Systems Inc.	rlake(at)galdosinc.com
Jan Herrmann	Ludwig-Maximilians-Universität München	jan.herrmann(at)lmu.de
John Herring	Oracle USA	john.herring(at)oracle.com
Martin Kyle	Sierra Systems Group Inc.	MartinKyle(at)SierraSystems.com

iii. Revision history

Date	Release	Author	Paragraph modified	Description
2007-01-22	0.1.0	Andreas Matheus	All	Initial Writing
2007-02-28	0.2.0	Andreas Matheus	All	Formatting
2007-03-19	0.3.0	Michael Mendonca	All	GML corrections
2007-08-06 – 2007-08-14	0.4.0	Jan Herrmann Andreas Matheus		Incorporation of comments received during the 30 day period for public comment Adding more geo-specific definitions.
2007-08-16	0.5.0	John Herring	2, Annex A	Description conformance classes
2007-08-22	0.6.0	Jan Herrmann Andreas Matheus		Clarification of test cases, geometry collections and bag functions
2007-08-27	0.7.0	Jan Herrmann Andreas Matheus	throughout	Editorial corrections
2007-09-06	0.7.1	Martin Kyle	throughout	More editorial corrections
2007-9-21 – 2007-10-25	0.8.0	Jan Herrmann Andreas Matheus	7,8,9, Annex A	Incorporation of comments and decisions from the TC meeting 17.-21.9.2007
2007-10-29	0.8.1	Martin Kyle	7,8, Annex A,B	Minor corrections, add Round Caps
2007-10-30	0.8.2	Jan Herrmann Andreas Matheus Martin Kyle	7,8,A,B	Incorporation of comments from Martin Kyle and adding the Unit of Measure "metre" for functions `Buffer, Area, Distance, IsWithinDistance, Length`
2007-11-5	0.8.3	John R. Herring		Editing for English and Simple Features.
2007-11-8	0.8.4	Jan Herrmann Andreas Matheus		Update for Unit of Measure and convert function, URN corrections
2007-11-9	0.8.5	John R. Herring		
2007-11-12	0.8.6	Jan Herrmann Andreas Matheus		Minor corrections
2007-11-13	0.8.7	Andreas Matheus	8.2.1 8.5 9.6 page ii	Correction of Buffer function Correct title to "… conversion functions" Insert of conversion function URNs Insert of additional disclaimer
2007-11-16	0.9	Andreas Matheus	throughout	Final formatting
2008-2-20	1.0	Carl Reed	Throughout	Fix cover page, redo preface, introduction, forward, and other edits as required.

iv. Changes to the OGC® Abstract Specification

The OpenGIS® Abstract Specification does not require changes to accommodate this standard.

Even though this document does not require the change of OGC specifications, there are potential change requests to the "OpenGIS® Web Service Common Implementation Specification" [11]. This is because GeoXACML can be used to establish an Access Control Mechanism to protect the access to OpenGIS Web Services (OWS). In that sense, an OWS can reply with error code exceptions, currently not defined in [11].

A Change Request to [11] shall be developed to specify exceptions, relevant in term of Access Control.

A Change Request to "Definition identifier URNs in OGC namespace" [15] is required to register the GeoXACML URN identifiers in the OGC context. In this Change Request, the following base identifiers are to be defined: "urn:ogc:def:dataType:geoxacml:1.0" and "urn:ogc:def:function:geoxacml:1.0". This document and the existing extensions define all GeoXACML identifier URNs.

v. Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

This document is not comprehensive in and of itself. Since this document defines an extension to an existing OASIS standard, as defined in [1]: "eXtensible Access Control Markup Language (XACML) Version 2.0". Therefore, the OGC recommends that using this document ONLY together with [1].

This document is also the core specification of a multi-part specification. Therefore, it is only meaningful if used with AT LEAST one extension.

Normative Annex A defines the conformance tables.

Normative Annex B defines the conformance definitions and tests.

Informative Annex C provides an example how to protect a WMS and WFS with geo-specific rights.

Please note that definitions taken from other specifications are shaded in 5% grey. They are included in the specification for the purpose of easy reading. However, the reader is urged to follow the references to the normative definitions.

vi. Introduction

Even though much geographic information is freely accessible, there are strong requirements for its protection and access control, such as for revisions or protection of confidential information. The need for managed access to geographic information is the result of many different driving forces including legal requirements and commercial interests.

In contrast to a "persistent protection mechanism," defined by the Geospatial Digital Rights Management Reference Model (GeoDRM-RM) to be a mechanism that "*remains in force regardless of where the content of the original resource is located or reproduced*" (p. 25, [9]), an Access Control System manages access to information only until it is obtained by the user and stored on a local computer system. Thinking of a service oriented Spatial Data Infrastructure, a "persistent protection mechanism" (e.g. GeoDRM System) would manage access at the time (i) the user requests geographic information from the service and (ii) afterwards, when the geographic information is stored somewhere else. The limitations of an Access Control System are such that access is typically used for online control at the time the user initiates a request to the service or before the corresponding response is submitted to the user. After the (geographic) information has left the service, it is no longer managed by the access control system.

In the real world, one can find numerous use cases where the establishment of access control is sufficient. In these cases, the complexity of the security system can be reduced significantly by merely using an Access Control System. This is in contrast to the case for a Security System implementing a comprehensive rights management policy. That said, GeoXACML is complementary to GeoRM.

The eXtensible Access Control Markup Language (XACML) allows establishing an Access Control System that can be used to manage access for Service Oriented Architectures (SOA). In that sense, XACML can be used to protect information by the declaration of rights through the specified Policy Language. Even though XACML is designed to be used as a multi-purpose language, it does not have the capabilities to express geo-specific constraints on access rights.

This document defines the Geospatial eXtensible Access Control Markup Language (GeoXACML) as a standardized geo-specific extension to the existing XACML Policy

Language. GeoXACML allows managing access to geographic information and services in an interoperable way across jurisdictions.

Arguments for standardization from this perspective include:

- The need for interoperability of rules and policies, respectively, if harmonization of distributed or hierarchical policy sets is required (e.g. Geospatial Data Infrastructure rules, INSPIRE rules, NSDI rules each refining the access rights for globally available web services)

- A common Policy Language as an important requirement to facilitate the development of sophisticated tools for access control systems

- The re-use of highly reliable software components, implemented in compliance to the defined Policy Language

OpenGIS® Geospatial eXtensible Access Control Markup Language (GeoXACML)
Implementation Specification – Version 1.0

1 Scope

The Geospatial eXtensible Access Control Markup Language (GeoXACML) defines an extension to the XACML Policy Language that supports the declaration and enforcement of access restrictions on geographic information.

The geospatial extension to XACML defined in this OGC standard is based on the extensibility points as introduced in section 8 (p. 89, [1]). In short, GeoXACML defines:

- the geometry model on which the geometric data types in access rules have to be based on,
- the different encoding languages for geometric data types (which are provided in the extensions to this core specification),
- the testing functions for topological relationships between geometries, and
- the geometric functions.

2 Conformance

Because GeoXACML uses the existing extension points of XACML, a GeoXACML Policy Language is based on the same XML Policy Schema as XACML. However, a GeoXACML Policy may include specific attribute data types and condition functions, which are defined in GeoXACML only. Therefore a PDP, supporting GeoXACML Policies SHALL be capable of performing authorization decisions on XACML policies; the opposite is not true.

In order to claim conformance with the GeoXACML specification, an implementation of a **PDP** SHALL be conform to:

(i) the XACML specification as stated in [1], section 10 AND

(ii) the normative sections of this document, in particular conformance tables, defined in Annex A and tests defined in Annex B AND

(iii) **at least** one geometry encoding as specified in an extension to this document and with the geometry model as defined in chapter 7 of this document[1].

The implementation must pass conformance tests from different categories, as defined in Annex B.

3 Normative references

[1] Remark: The geometry model defined in chapter 7 is based on the simple feature geometry types as defined in [4].

[1] OASIS, *eXtensible Access Control Markup Language (XACML) Version 2.0, 1 Feb 2005*, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

[2] OASIS, *eXtensible Access Control Markup Language (XACML) Version 2.0 - Policy XML Schema: access_control-xacml-2.0-policy-schema-os.xsd*

[3] OASIS, *eXtensible Access Control Markup Language (XACML) Version 2.0 - Context XML Schema: access_control-xacml-2.0-context-schema-os.xsd*

[4] OGC, *Open Geospatial Consortium Inc.: OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture, Version: 1.2.0, Date: 2006-10-05,* http://portal.opengeospatial.org/files/?artifact_id=18241

[5] National Institute of Standards and Technology (NIST), *NIST Guide to SI Units*: *B. 9 Factors for units listed by kind of quantity or field of science*, http://physics.nist.gov/Pubs/SP811/appenB9.html

4 Terms and definitions

For the purposes of this document, the following terms and definitions apply. Please note that terms and definitions taken from other specifications are informative and shaded 5% grey. They are included here for easy reading. For the normative definition, please follow the reference.

4.1 Terms and definitions from eXtensible Access Control Markup Language (XACML) Version 2.0

For the normative definitions, please see [1].

Access - Performing an action

Access control - Controlling access in accordance with a policy

Action - An operation on a resource

Applicable policy - The set of policies and policy sets that governs access for a specific decision request

Attribute - Characteristic of a subject, resource, action or environment that may be referenced in a predicate or target (see also – named attribute)

Authorization decision - The result of evaluating applicable policy, returned by the PDP to the PEP. A function that evaluates to “Permit”, “Deny”, “Indeterminate” or “NotApplicable", and (optionally) a set of obligations

Bag – An unordered collection of values, in which there may be duplicate values

Condition - An expression of predicates. A function that evaluates to "True", "False" or “Indeterminate”

Conjunctive sequence - a sequence of predicates combined using the logical ‘AND’ operation

Context - The canonical representation of a decision request and an authorization decision

Context handler - The system entity that converts decision requests in the native request format to the XACML canonical form and converts authorization decisions in the XACML canonical form to the native response format

Decision – The result of evaluating a rule, policy or policy set

Decision request - The request by a PEP to a PDP to render an authorization decision

Disjunctive sequence - a sequence of predicates combined using the logical 'OR' operation

Effect - The intended consequence of a satisfied rule (either "Permit" or "Deny")

Environment - The set of attributes that are relevant to an authorization decision and are independent of a particular subject, resource or action

Named attribute – A specific instance of an attribute, determined by the attribute name and type, the identity of the attribute holder (which may be of type: subject, resource, action or environment) and (optionally) the identity of the issuing authority

Obligation - An operation specified in a policy or policy set that should be performed by the PEP in conjunction with the enforcement of an authorization decision

Policy - A set of rules, an identifier for the rule-combining algorithm and (optionally) a set of obligations. May be a component of a policy set

Policy administration point (PAP) - The system entity that creates a policy or policy set

Policy-combining algorithm - The procedure for combining the decision and obligations from multiple policies

Policy decision point (PDP) - The system entity that evaluates applicable policy and renders an authorization decision. This term is defined in a joint effort by the IETF Policy Framework Working Group and the Distributed Management Task Force (DMTF)/Common Information Model (CIM) in [RFC3198]. This term corresponds to "Access Decision Function" (ADF) in [ISO10181-3].

Policy enforcement point (PEP) - The system entity that performs access control, by making decision requests and enforcing authorization decisions. This term is defined in a joint effort by the IETF Policy Framework Working Group and the Distributed Management Task Force (DMTF)/Common Information Model (CIM) in [RFC3198]. This term corresponds to "Access Enforcement Function" (AEF) in [ISO10181-3].

Policy information point (PIP) - The system entity that acts as a source of attribute values

Policy set - A set of policies, other policy sets, a policy-combining algorithm and (optionally) a set of obligations. May be a component of another policy set

Predicate - A statement about attributes whose truth can be evaluated

Resource - Data, service or system component

Rule - A target, an effect and a condition. A component of a policy

Rule-combining algorithm - The procedure for combining decisions from multiple rules

Subject - An actor whose attributes may be referenced by a predicate

Target - The set of decision requests, identified by definitions for resource, subject and action, that a rule, policy or policy set is intended to evaluate

5 Conventions

5.1 Symbols (and abbreviated terms)

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
GeoXACML	Geospatial eXtensible Access Control Markup Language
GeoDRM	Geospatial Digital Rights Management
GeoDRM-RM	Geospatial Digital Rights Management – Reference Model
GML	Geography Markup Language
ISO	International Organization for Standardization
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
OWS	OpenGIS Web Services
PAP	Policy Administration Point
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PIP	Policy Information Point
SAML	Security Assertion Markup Language
SOA	Service Oriented Architecture
UML	Unified Modeling Language
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language

5.2 UML Notation

The diagrams that appear in this standard are presented using the Unified Modelling Language (UML) static structure diagram. The UML notations used in this standard are described in the diagram below.

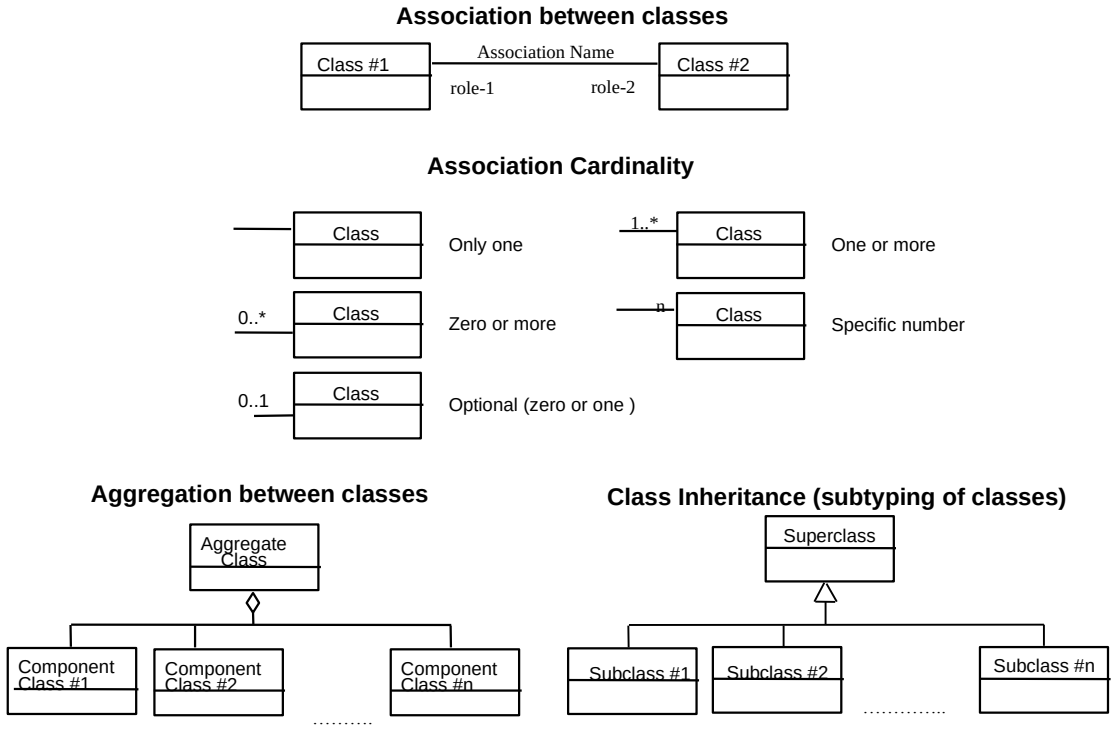

Figure 1 — UML notation

6 Brief Introduction to XACML (informative)

This chapter provides a brief informative introduction of XACML.

A short primer on XACML is available as a Technology Report at the OASIS Cover Pages; see [8] for more information.

The XACML standard can be separated into two main sections, which are introduced in more detail in the following sections: (i) Policy Language and Authorization Model as and (ii) Information Flow Model.

6.1 Policy Language Model and Authorization

The XACML Policy Language Model defines an XML encoding for expressing general purpose access restrictions and extension points to define your own Attribute Values, Functions, etc. The entire set of access restrictions defines an XACML Policy. The Policy is structured, according to the following UML diagram (see Figure 2).

The top level element is the <PolicySet>. It can host zero or more <PolicySet> elements, which can be included inline or by reference. This powerful feature allows the reuse of pre-defined policy segments as well as the integration of multiple policies.

Each <PolicySet> element can host one or more <Policy> elements, which is the container for a set of <Rule> elements. Inside the <Rule> element, conditions can be formed to express complex access restrictions, using the <Condition> element.

Each <PolicySet>, <Policy> and <Rule> element have a <Target> element, which can be used to define simple matching conditions for the Subject, Action, Resource and Environment. This allows the effective structuring of a policy into sub-trees, which eases the maintenance of rights defined in a policy. On the other hand, the simple matching in a <Target> element ensures fast decision making, when it comes to deriving an authorization decision.

The flexible matching of Subjects in the <Target> element supports direct association of access rights to subjects or roles, as defined in the RBAC profile of XACML (“Core and hierarchical role based access control (RBAC) profile of XACML v2.0”,[6]).

In order to derive an authorization decision (i.e. XACML authorization decision) for a given request, the XACML policy is traversed from the top (i.e. <PolicySet> element) to the leaves (i.e. <Rule> elements). For all matching <Rule> elements, their Effect (i.e. Permit or Deny) is taken as the most basic driver for the authorization decision. By traversing up the policy the effects of all Rules – associated to a <Policy> element – are combined using the RuleCombiningAlgorithm. The resulting effects of all <Policy> elements are matched on the next highest level, until reaching the top <PolicySet> element; the PolicyCombiningAlgorithm creates the final effect of the entire policy, which represents the authorization decision.

The XACML Policy Language defines four different results for the authorization decision: (i) Permit, (ii) Deny, (iii) Indeterminate and (iv) NotApplicable. Finally, the process of deriving an authorization decision can result in an error, which is documented as additional information in the <Decision> element.

In addition, the decision can be "Permit with Obligation", which can be expressed in the <Obligation> element, attached to the <Policy> or <PolicySet> element.

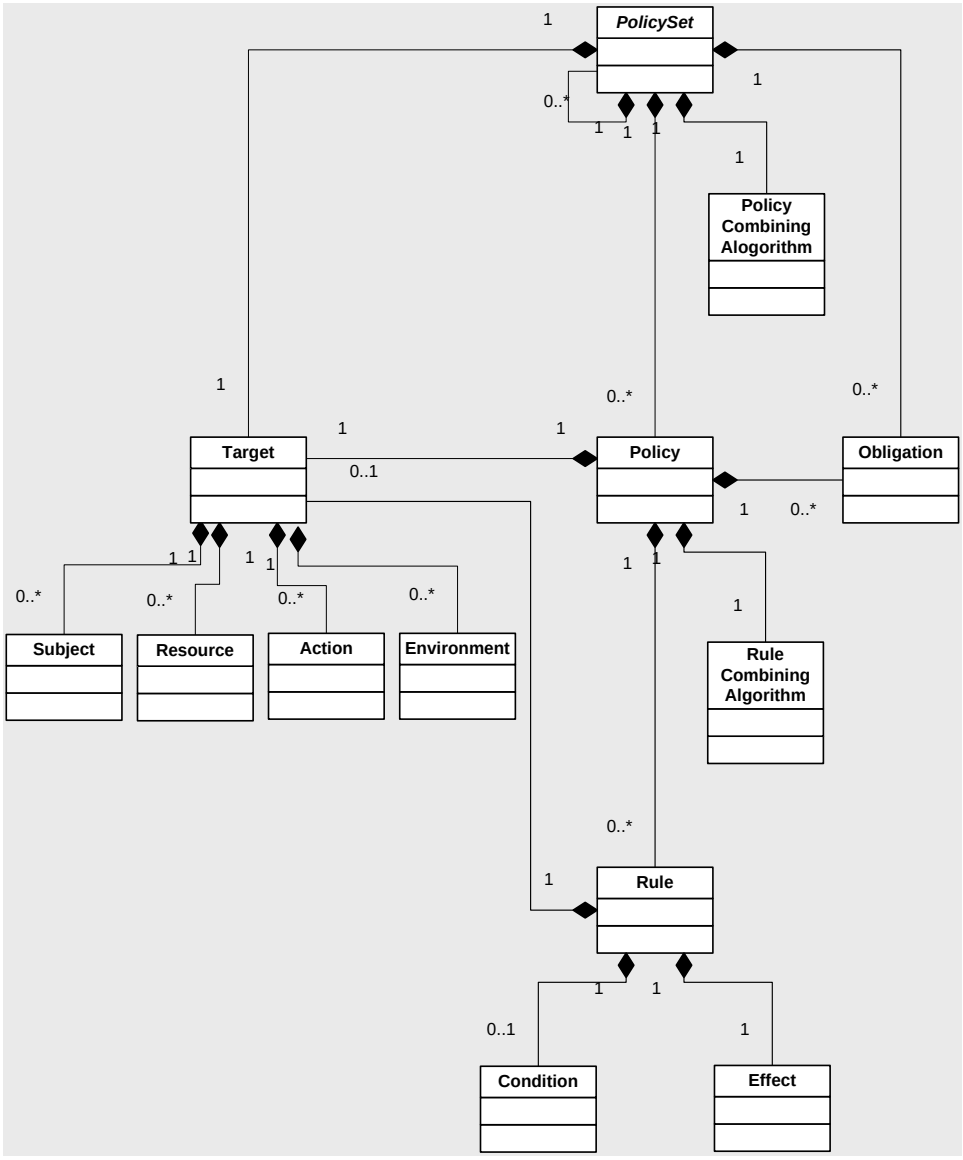

Figure 2 — XACML Policy Language Model

6.2 Information Flow Model

The XACML Information Flow Model defines the architecture of a modular and distributed access control system. In addition, it defines the exchange of messages between the components and the structure of the messages. The following figure illustrates the informative architecture and the sequence of messages, sent between the components of the access control system.

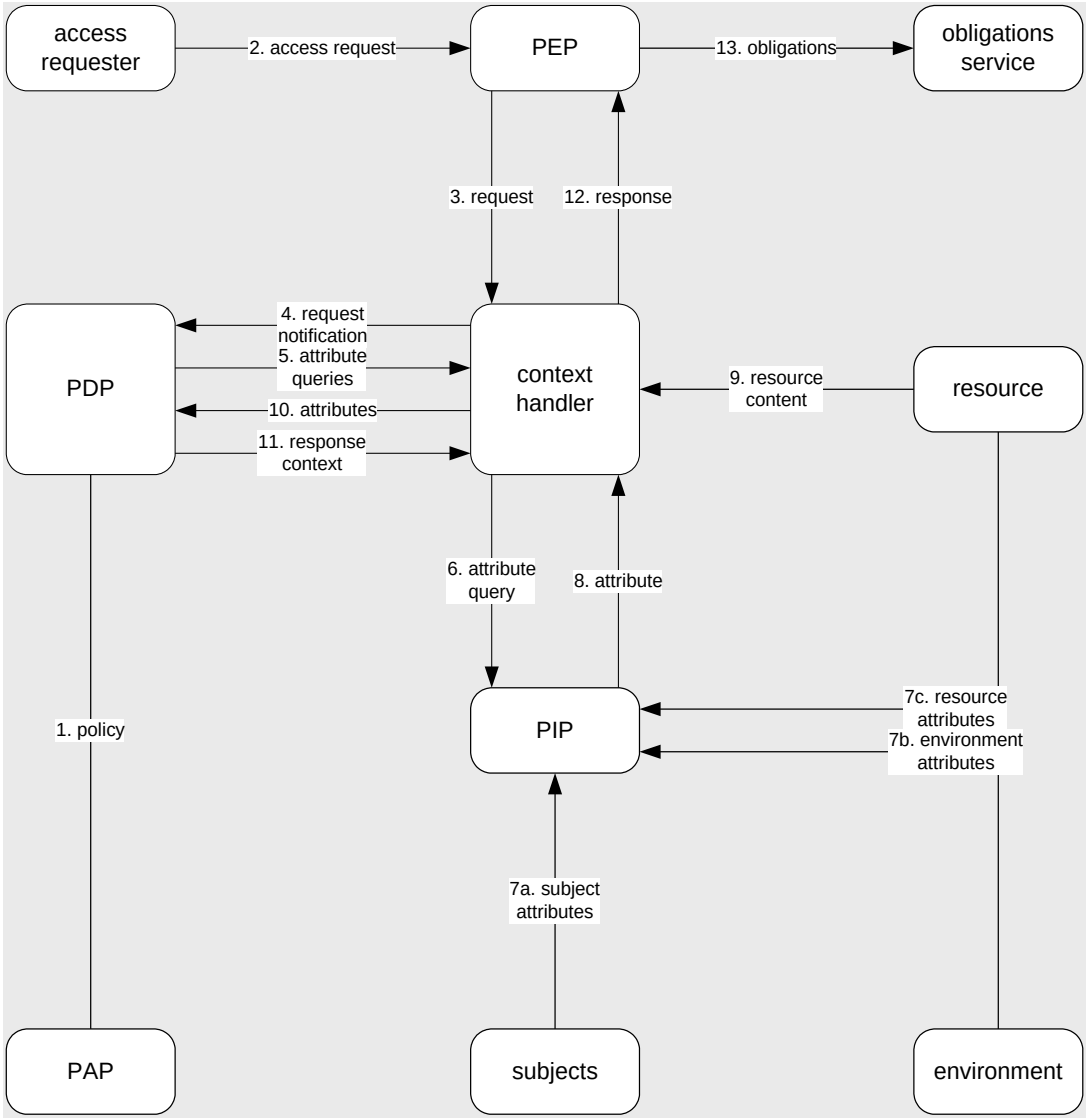

Figure 3 — XACML Information Flow Model

The Policy Administration Point (PAP) is the component that allows one or multiple policy administrators to maintain access rights in a set of policies. In addition, the PAP might provide an interface for requesting policies, as it is defined in the "SAML 2.0 profile of XACML v2.0" of the XACML standard (see [7]).

The Policy Decision Point (PDP) is the component that derives an authorization decision based on a request, received from one or multiple Policy Enforcement Point(s) (PEP). A PDP may request policies from the PAP or use a policy repository on file or in a database.

The Policy Enforcement Point (PEP) can be characterized as a binary switch that either forwards the intercepted request from the client to the service (and the response from the service to the client respectively) or replies with an adequate error message. The decision if the request or response is to be forwarded or blocked depends on the authorization decisions, received from the PDP. Because the PEP must request authorization decisions in a particular XACML message format, it is the duty of the context handler to collect all relevant information and prepare the authorization decision request message. The information, collected by the context handler can include the identity information about the user, the action to be taken on the resource, information about the resource itself, the IP address of the client, the time of the request, certificate information, etc… In order to collect all relevant information, it can be required to request such information from the Policy Information Point (PIP).

6.3 Extension capabilities of XACML

The XACML specification defines the non-normative extensibility points (section 8, [1]). For this specification, it is important to note that the `DataType, FunctionId` and `AttributeId` can be extended.

Please see the XACML schema definitions in access_control-xacml-2.0-policy-schema-os.xsd ([2]) for the XML format of the elements.

6.3.1 Defining new Data Types

An `<AttributeValue>` element has an attribute named DataType, which is of type xs:anyURI. According to the extension capabilities of XACML, additional attribute values can be defined by associating a unique DataType identifier to it.

Section 8.2 of the XACML specification states that *"<xacml:AttributeValue> and <xacml-context:AttributeValue> elements MAY contain an instance of a structured XML data-type."*.

This capability allows the definition of geometry data types, as defined by GeoXACML.

```
<xs:element name="AttributeValue" type="xacml:AttributeValueType"
substitutionGroup="xacml:Expression"/>
<xs:complexType name="AttributeValueType" mixed="true">
```

```
  <xs:complexContent mixed="true">
    <xs:extension base="xacml:ExpressionType">
      <xs:sequence>
        <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
      <xs:anyAttribute namespace="##any" processContents="lax"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Figure 4 — XACML schema definition of the <AttributeValue> element

6.3.2 Defining new Functions

A <Function> element has an attribute named FunctionId, which is of type xs:anyURI. According to the extension capabilities of XACML, additional functions can be defined by associating a unique FunctionId to it.

This capability allows the definition of geo-specific functions, as defined by GeoXACML.

```
<xs:element name="Function" type="xacml:FunctionType"/>
<xs:complexType name="FunctionType">
  <xs:attribute name="FunctionId" type="xs:anyURI" use="required"/>
</xs:complexType>
```

Figure 5 — XACML schema definition of the <Function> element

6.3.3 Defining new Identifiers / Attributes

An <AttributeDesignator> element allows fetching information from an XACML AuthorizationDecisionRequest based on named attributes. In order to specify new names for attributes, the <AttributeDesignatorType> has an attribute named AttributeId. In order to use GeoXACML specific data types, the value of the attribute DataType SHALL be used according to the specified data types. According to the extension capabilities of XACML, additional identifier-names or attribute-names can be defined by associating a unique AttributeId to it.

An <AttributeSelector> element allows fetching information from an XACML AuthorizationDecisionRequest based on the XML encoded information as it can be inserted into the <ResourceContent> element. The value of the attribute named DataType SHALL be used according to the specified data types.

These capabilities allow the fetching of geo-specific information using either the XACML AttributeDesignator or the AttributeSelector without modifying any XACML schema (policy or authorization decision request schema).

The following two figures highlight the extension points in the XACML schema that are used by GeoXACML.

```
<xs:complexType name="AttributeDesignatorType">
  <xs:complexContent>
    <xs:extension base="xacml:ExpressionType">
      <xs:attribute name="AttributeId" type="xs:anyURI" use="required"/>
      <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
      <xs:attribute name="Issuer" type="xs:string" use="optional"/>
      <xs:attribute name="MustBePresent" type="xs:boolean" use="optional"
default="false"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Figure 6 — XACML schema definition of the `<AttributeDesignatorType>` element

```
<xs:complexType name="AttributeSelectorType">
  <xs:complexContent>
    <xs:extension base="xacml:ExpressionType">
      <xs:attribute name="RequestContextPath" type="xs:string" use="required"/>
      <xs:attribute name="DataType" type="xs:anyURI" use="required"/>
      <xs:attribute name="MustBePresent" type="xs:boolean" use="optional"
default="false"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Figure 7 — XACML schema definition of the `<AttributeSelectorType>` element

7 GeoXACML Geometry Model (normative)

This chapter describes the GeoXACML core geometry model. It is separated from the encoding for geometries that are used in GeoXACML policy declarations. Please note that the actual XML encoding of these geometries can vary. Therefore, this standard provides multiple extension parts in which a particular encoding is defined.

Any conformant implementation for GeoXACML SHALL support at least one geometry schema encoding (introduced in the extension parts of this document) supporting the geometry types listed below.

In this standard, the underlying geometry models of all geometry schema encodings in the extension parts are based on the geometry definitions as specified in the OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture, Version: 1.2.0 ([4]).

In order to support a flexible and straight forward solution for allowing geometric data types that easily comply with the base XACML specification, GeoXACML extends XACML by only one new data type that is named "urn:ogc:def:dataType:geoxacml:1.0:geometry".

This implies that the data type of every geometric attribute value, bag of geometric values or pointers to geometric data (i.e. AttributeSelector or AttributeDesignator) in a GeoXACML policy SHALL always be "urn:ogc:def:dataType:geoxacml:1.0:geometry".

By using a super class for all geometric data types (see [4]) as the only spatial data type in GeoXACML, this standard remains "stable" even if new extensions parts are added. Additionally, the use of the geometric super class as the only new data type in GeoXACML simplifies the declaration of the function signatures as well as the implementation of GeoXACML conformant access control systems.

As all extensions are based on the underlying geometry model of [4], the specific values for a geometry <AttributeValue> of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" must be represented by of one of the following basic types:

Remark: The definitions are listed here for easy reading. For the normative definitions, please see [4].

Point - A Point is a 0-dimensional geometric object and represents a single location in coordinate space. A Point has an x-coordinate value, a y-coordinate value. If called for by the associated Spatial Reference System, it may also have coordinate values for z and m.

LineString - A LineString is a Curve with linear interpolation between Points. Each consecutive pair of Points defines a Line segment.

Polygon - A Polygon is a planar Surface defined by 1 exterior boundary and 0 or more interior boundaries. Each interior boundary defines a hole in the Polygon.

MultiPoint - A MultiPoint is a 0-dimensional GeometryCollection. The elements of a MultiPoint are restricted to Points.

MultiLineString - A MultiLineString is a MultiCurve whose elements are LineStrings.

MultiPolygon - A MultiPolygon is a MultiSurface whose elements are Polygons.

Geometries as listed above SHALL be represented as a child node of an <AttributeValue> element. Regardless of the type of the enclosed geometry, the "DataType" attribute of the <AttributeValue> element SHALL be set to "urn:ogc:def:dataType:geoxacml:1.0:geometry". The encoding of the enclosed geometry SHALL conform to the syntax as specified in the chosen encoding extension.

As defined in [4], a geometry might be empty. The encoding of an empty geometry SHALL be compliant with the definition in the relevant extension. If necessary, an empty geometry can also be represented as an empty <AttributeValue> of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry":

<AttributeValue DataType="urn:ogc:def:dataType:geoxacml:1.0:geometry"/>.

Please note that there is a difference between the geometric data type (as shown above) in GeoXACML and the XML encoded data type as represented in the extension(s). Like in the geometry model of SFS, the geometric data type in this standard is an abstract super class. The concrete instantiations of this super class are the XML encoded data types found in the extension(s).

8 GeoXACML Spatial Function Definitions (normative)

Each of the following relational and geometric operators is supported by GeoXACML, in at least one of the conformance classes defined in Annex A. Their definition is based on the OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture, Version: 1.2.0 ([4]).

In case a GeoXACML defined function returns a bag of geometries, the implementation SHALL provide this as a Bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry". The same applies if one explicitly defines a bag of geometries in a policy. Note that all common Bag and Set Functions (see 8.3 and 8.4) can be used on Bags of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry".

If the geometric operator returns an empty Bag, this is to be represented as an empty Bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry".

All geometries, explicitly defined in a <Rule> element of a GeoXACML policy SHALL use only one encoding extension to this standard.

In GeoXACML, the internal unit of length is metres and the internal unit of area is square metres. Accordingly, length or area values that are either a function parameter or the result of a function, are based on GeoXACML's internal unit of length or area. In use cases, where the distance / area is specified in a different unit of measure, a conversion SHALL be achieved using the functions as defined in 8.5.

All defined functions SHALL return a XACML ProcessingError with the state INDETERMINATE if

- the XML encoded data type of the argument g is invalid or not suitable for the function according to the definition in [4], or
- the processing of the underlying functionality results in an error.

In addition, all defined functions with two arguments g1 and g2 of any geometric data type SHALL return a XACML ProcessingError with the state INDETERMINATE if

- the combination of data types for g1 and g2 is invalid according to the definition in [4].
- the Coordinate Reference System definitions for g1 and g2 are both explicitly given but their values are different and the implementation does not support a CRS-transformation at runtime.

The Set and Bag Functions as defined in 8.3 and 8.4 are not spatial functions. Only spatial functions from 8.1 and 8.2 should be used to test spatial conditions.

8.1 Definition of Topological Functions

All functions defined in this section SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a value of data type "http://www.w3.org/2001/XMLSchema#boolean". In the following definitions, a geometry is treated as a representation of a possibly infinite set of points. The representations obviously are not infinite, but use a variety of interpolation algorithms.

Contains(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometry g2 lies in the closure (boundary union interior) of geometry g1 – the inverse of Within(g1: Geometry, g2: Geometry).

Crosses(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometries g1 and g2 share some but neither is contained in the other, and the dimension of the intersection is less than that of both of the geometries.

Disjoint(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometries g1 and g2 have no point in common.

Equals(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometries g1 and g2 are equal (geometrically contain exactly the same points).

Intersects(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometries g1 and g2 have at least one point in common (the inverse of Disjoint).

Overlaps(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometries g1 and g2 share some but not all points in common, and the intersection has the same dimension as the geometries themselves.

Touches(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometries g1 and g2 have at least one boundary point in common, but no interior points.

Within(g1:Geometry, g2:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometry g1 is spatially within geometry g2; that is if every point on g1 is also on g2.

8.2 Definition of Geometric Functions

8.2.1 Definition of Constructive Geometric Functions

Buffer(g:Geometry, d:Double) : Bag
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and "http://www.w3.org/2001/XMLSchema#double.

This function SHALL return a bag of geometry values representing the buffer of geometry g at distance d. The buffer of a geometry at distance d is the Polygon or MultiPolygon which contains all points within a distance d of the geometry. The unit of measure for the argument d SHALL be metre.

Boundary(g:Geometry) : Bag
This function SHALL take one argument of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a Bag of geometries.

This function SHALL return a bag of geometry values representing the combinatorial boundary of geometry g.

ConvexHull(g:Geometry) : Geometry
This function SHALL take one argument of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a value of the same data type.

This function SHALL return a geometric object that represents the convex hull of geometry g. The convex hull is the smallest convex polygon that contains all points of the geometry. The convex Hull for a geometry of encoding data type Point is the point itself.

Centroid(g:Geometry) : Geometry
This function SHALL take one argument of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a value of the same data type. This function SHALL return the point that is the geometric centre of gravity of the geometry g.

Difference(g1:Geometry, g2:Geometry) : Bag
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a Bag of geometries.

This function SHALL return a bag of geometry values representing the closure of the set difference between the two geometries g1 and g2. The difference is the set of all points which lie on g1 but not on g2.

SymDifference(g1:Geometry, g2:Geometry) : Bag
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a Bag of geometries.

This function SHALL return a bag of geometry values representing the closure of the symmetric difference of two Geometries. The symmetric difference is the set of points which lie on either g1 or g2 but not in both.

Intersection(g1:Geometry, g2:Geometry) : Bag
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a Bag of geometries.

This function SHALL return a bag of geometry values representing the Point set intersection of geometry g1 and geometry g2.

Union(g1:Geometry, g2:Geometry) : Bag
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return a Bag of geometries.

This function SHALL return a bag of geometry values representing a Point set union of geometry g1 with geometry g2.

8.2.2 Definition of Scalar Geometric Functions

Area(g:Geometry) : Double
This function SHALL take one argument of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return value of data type"http://www.w3.org/2001/XMLSchema# double".

This function SHALL return a value representing the area of geometry g The unit of measure for the return value SHALL be square metre. This function SHALL return zero for a geometry of encoding data type Point or LineString.

Distance(g1:Geometry, g2:Geometry) : Double
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return value of data type "http://www.w3.org/2001/XMLSchema# double".

This function SHALL return a value representing the shortest distance in metre between any two points in the two geometries g1 and g2.
Because the geometries are closed, it is possible to find a point on each geometric object involved, such that the distance between these 2 points is the returned distance between their geometric objects.

IsWithinDistance(g1:Geometry, g2:Geometry, d:Double) : Boolean
This function SHALL take two arguments of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry", an argument of data type "http://www.w3.org/2001/XMLSchema# double" and return value of data type"http://www.w3.org/2001/XMLSchema#boolean".

This function SHALL return a TRUE value if and only if the shortest distance between the geometries g1 and g2 is less than or equal to a specified distance d in metre.

Length(g:Geometry) : Double
This function SHALL take one argument of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return value of data type "http://www.w3.org/2001/XMLSchema# double".

This function SHALL return a value representing the length of geometry g. The unit of measure for the return value SHALL be metre. This function SHALL return zero for a geometry of encoding data type Point. This function SHALL return the perimeter for a geometry of encoding data type Polygon.

8.2.3 Definition of Functions to check special characteristics

All functions defined in this section SHALL take one argument of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and return value of data type "http://www.w3.org/2001/XMLSchema#boolean".

IsSimple(g:Geometry) : Boolean
This function SHALL return a TRUE value if and only if geometry g has no anomalous geometric points, such as self intersection or self tangency. Simple curves intersect themselves only at their endpoints (only if they are also closed, see next function).

IsClosed(g:Geometry) : Boolean
This function SHALL return a TRUE value for a curve g if and only if the start and end point of the curve g are identical. For a point, a finite set of points or an empty geometry this function SHALL return TRUE. Surfaces require a 3D space to close. In general, a closed geometry has an empty boundary.

IsValid(g:Geometry) : Boolean
This function SHALL return a TRUE value if and only if the geometry g is valid as specified in [4].

Remark: Even though it is implicit that geometries (to be processed by the PDP) are valid, this operator allows explicit checking similar to Exception handling in the Java programming language. Java defines checked exceptions that must be handled by the programmer and unchecked exceptions, where handling in the source code is optional. The intended use of the IsValid operator is similar to unchecked exceptions.

8.3 Definition of Bag Functions

GeometryOneAndOnly(b: Bag) : Geometry

This function SHALL take a bag with values of data type Geometry as an argument and SHALL return a value of data type “urn:ogc:def:dataType:geoxacml:1.0:geometry”.

This function SHALL return the value of the only element in the bag b.

This function SHALL return a XACML ProcessingError with state INDETERMINATE if the geometry bag b does not have exactly one value.

GeometryBagSize(b: Bag) : Integer
This function SHALL take a bag with values of data type “urn:ogc:def:dataType:geoxacml:1.0:geometry”as an argument and SHALL return a value of data type “http://www.w3.org/2001/XMLSchema#integer”.

This function SHALL return a value indicating the number of elements in the bag.

GeometryIsIn(g:Geometry, b: Bag) : Boolean
This function SHALL take as argument a value of data type “urn:ogc:def:dataType:geoxacml:1.0:geometry”as the first argument, a bag with values of data type “urn:ogc:def:dataType:geoxacml:1.0:geometry”as the second argument and SHALL return an “http://www.w3.org/2001/XMLSchema#boolean”.

The function SHALL return a TRUE value if and only if the first argument matches by the function “urn:ogc:def:function:geoxacml:1.0:geometry-equals**”** any value in the bag.

GeometryBag(g*:Geometry) : Bag
This function SHALL take any number of arguments whose values are all of data type “urn:ogc:def:dataType:geoxacml:1.0:geometry”and return a bag of geometries containing the values of the arguments**.**

An application of this function to zero arguments SHALL produce an empty bag of data-type “urn:ogc:def:dataType:geoxacml:1.0:geometry**”**.

Remark: The Bag Functions as defined by GeoXACML can be used in conjunction with XACML higher-order bag functions as defined in [1], A.3.12.

8.4 Definition of Set Functions

These functions operate on bags of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" mimicking sets by eliminating duplicate elements from a geometry bag.

GeometryBagIntersection(b1: Bag, g2: Bag) : Bag
This function SHALL take two arguments that are both a bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry".

This function SHALL return a bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" such that it contains only elements that are common between the bags b1 and b2, which is determined by "urn:ogc:def:function:geoxacml:1.0:geometry-equals". No duplicates, as determined by "urn:ogc:def:function:geoxacml:1.0:geometry-equals", SHALL exist in the result.

GeometryBagAtLeastOneMemberOf(b1: Bag, b2: Bag) : Boolean
This function SHALL take two arguments that are both a bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and SHALL return a "http://www.w3.org/2001/XMLSchema#boolean".

The function SHALL evaluate to TRUE if and only if at least one element of the bag b1 is contained in the second bag b2 as determined by "urn:ogc:def:function:geoxacml:1.0:geometry-is-in".

GeometryBagUnion(b1: Bag, b2: Bag) : Bag
This function SHALL take two arguments that are both a bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry".

This function SHALL return a bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" such that it contains all elements of both bags b1 and b2. No duplicates, as determined by "urn:ogc:def:function:geoxacml:1.0:geometry-equals", SHALL exist in the result.

GeometryBagSubset(b1: Bag, b2: Bag) : Boolean
This function SHALL take two arguments that are both a bag of data type "urn:ogc:def:dataType:geoxacml:1.0:geometry" and SHALL return a "http://www.w3.org/2001/XMLSchema#boolean".

This function SHALL return a TRUE value if and only if the bag b1 is a subset of the bag b2. Each argument SHALL be considered to have had its duplicates removed, as determined by "urn:ogc:def:function:geoxacml:1.0:geometry-equals", before the subset calculation.

GeometrySetEquals(b1: Bag, b2: Bag) : Boolean
This function SHALL take two arguments that are both a bag of data

type“urn:ogc:def:dataType:geoxacml:1.0:geometry” and SHALL return a “http://www.w3.org/2001/XMLSchema#boolean”.

This function SHALL return the result of applying "urn:oasis:names:tc:xacml:1.0:function:and" to the application of "urn:ogc:def:function:geoxacml:1.0:geometry-bag-subset" to bag b1 and bag b2 and the application of "urn:ogc:def:function:geoxacml:1.0:geometry-bag-subset" to bag b2 and bag b1.

Remark: The Set Functions as defined by GeoXACML can be used in conjunction with XACML higher-order bag functions as defined in [1], A.3.12.

8.5 Definition of Conversion Functions

As stated before, in GeoXACML the only internal unit of length is metre and the only internal unit of area is square metre. This implies that all length / area arguments in rules that are based on different units of measure have to be converted into metre / square metre, before they are processed by the corresponding functions (e.g. IsWithinDistance or Buffer function). This can be achieved by using the appropriate conversion function as defined below.

ConvertToMetre(d:Double, u:String) : Double
This function SHALL take one argument of data type "http://www.w3.org/2001/XML-Schema#double" and one argument of data type http://www.w3.org/2001/XML-Schema#string, representing a length unit of measure. This function SHALL return a value of data type "http://www.w3.org/2001/XMLSchema# double".

This function converts the distance value d given in the unit of measure u into the corresponding distance value based on the basic unit metre.

This function SHALL return a XACML ProcessingError with state INDETERMINATE if the unit of measure, as given by the parameter u, cannot be converted to metre.

ConvertToSquareMetre(a:Double, u:String) : Double
This function SHALL take one argument of data type "http://www.w3.org/2001/XMLSchema# double" and one argument of data type http://www.w3.org/2001/XMLSchema#string, representing an area unit of measure. This function SHALL return a value of data type "http://www.w3.org/2001/XMLSchema# double".

This function converts the area value a given in the unit of measure u into the corresponding area value based on the basic unit square metre.

This function SHALL return a XACML ProcessingError with state INDETERMINATE if the unit of measure, as given by the parameter u, cannot be converted to square metre.

Remark: In order to avoid unnecessary conversions at runtime, it is highly recommended to state distance and area literals in rules directly in metre and square metre.

9 GeoXACML Identifier Definitions (normative)

The implementation MUST use the attributes associated with the following identifiers in the way GeoXACML has defined. The second column in the tables refers to the name of the functions as used in Chapter 8 of this specification or in [1].

9.1 Identifier for Geometry

The following identifier SHALL be used to reference the geometric data type Geometry.

URN	Data Type
urn:ogc:def:dataType:geoxacml:1.0:geometry	Geometry

Table 1 — GeoXACML Geometry URN

In order to construct geometric data types using a specific encoding scheme, GeoXACML defines more URNs in the existing extensions.

9.2 Identifiers for Topological Functions

In order to define functions for testing topological relations, GeoXACML defines the following URNs.

URN	Function
urn:ogc:def:function:geoxacml:1.0:geometry-equals	Equals
urn:ogc:def:function:geoxacml:1.0:geometry-disjoint	Disjoint
urn:ogc:def:function:geoxacml:1.0:geometry-touches	Touches
urn:ogc:def:function:geoxacml:1.0:geometry-crosses	Crosses
urn:ogc:def:function:geoxacml:1.0:geometry-within	Within
urn:ogc:def:function:geoxacml:1.0:geometry-contains	Contains
urn:ogc:def:function:geoxacml:1.0:geometry-overlaps	Overlaps
urn:ogc:def:function:geoxacml:1.0:geometry-intersects	Intersects

Table 2 — Topological Function URNs

9.3 Identifiers for Geometric Functions

In order to express and enforce more complex access restrictions based on spatial analysis, GeoXACML defines the following geometric function URNs.

9.3.1 Identifiers for Constructive Geometric Functions

URN	Geometric Function
urn:ogc:def:function:geoxacml:1.0:geometry-buffer	Buffer
urn:ogc:def:function:geoxacml:1.0:geometry-boundary	Boundary
urn:ogc:def:function:geoxacml:1.0:geometry-union	Union
urn:ogc:def:function:geoxacml:1.0:geometry-intersection	Intersection
urn:ogc:def:function:geoxacml:1.0:geometry-difference	Difference
urn:ogc:def:function:geoxacml:1.0:geometry-sym-difference	SymDifference
urn:ogc:def:function:geoxacml:1.0:geometry-centroid	Centroid
urn:ogc:def:function:geoxacml:1.0:geometry-convex-hull	ConvexHull

Table 3 — Constructive geometric function URNs

9.3.2 Identifiers for Scalar Geometric Functions

URN	Geometric Function
urn:ogc:def:function:geoxacml:1.0:geometry-distance	Distance
urn:ogc:def:function:geoxacml:1.0:geometry-is-within-distance	IsWithinDistance
urn:ogc:def:function:geoxacml:1.0:geometry-length	Length
urn:ogc:def:function:geoxacml:1.0:geometry-area	Area

Table 4 — Scalar geometric function URNs

9.3.3 Identifiers for Miscellaneous Geometric Functions

URN	Geometric Function
urn:ogc:def:function:geoxacml:1.0:geometry-is-simple	IsSimple
urn:ogc:def:function:geoxacml:1.0:geometry-is-closed	IsClosed
urn:ogc:def:function:geoxacml:1.0:geometry-is-valid	IsValid

Table 5 — Miscellaneous geometric function URNs

9.4 Identifiers for Bag Functions for XACML Bags containing Geometries

In order to operate on bags containing geometries, GeoXACML defines the following geometric bag function URNs.

Remark: GeoXACML does not define new Bag Functions but leverages the Bag Functions for geometries as defined in the XACML specification (see [1], A.3.10).

URN	Bag Function
urn:ogc:def:function:geoxacml:1.0:geometry-one-and-only	GeometryOneAndOnly
urn:ogc:def:function:geoxacml:1.0:geometry-bag-size	GeometryBagSize
urn:ogc:def:function:geoxacml:1.0:geometry-is-in	GeometryIsIn
urn:ogc:def:function:geoxacml:1.0:geometry-bag	GeometryBag

Table 6 — Geometric Bag Function URNs

9.5 Identifiers for geometric Set Functions

In order to analyse geometric bags, GeoXACML defines the following geometric set function URNs.

URN	Set Function
urn:ogc:def:function:geoxacml:1.0:geometry-bag-intersection	GeometryBagIntersection
urn:ogc:def:function:geoxacml:1.0:geometry-at-least-one-member-of	GeometryAtLeastOneMemberOf
urn:ogc:def:function:geoxacml:1.0:geometry-bag-union	GeometryBagUnion
urn:ogc:def:function:geoxacml:1.0:geometry-bag-subset	GeometryBagSubset
urn:ogc:def:function:geoxacml:1.0:geometry-set-equals	GeometrySetEquals

Table 7 — Geometric Set Function URNs

9.6 Identifiers for conversion Functions

In order to convert units of measure, GeoXACML defines the following conversion function URNs.

URN	Set Function
urn:ogc:def:function:geoxacml:1.0:convert-to-metre	ConvertToMetre
urn:ogc:def:function:geoxacml:1.0:convert-to-square-metre	ConvertToSquareMetre

Table 8 — Conversion Function URNs

Annex A GeoXACML Conformance Tables (normative)

This Annex defines which GeoXACML attribute, identifier, data type and function definitions are mandatory to be implemented toward BASIC or STANDARD conformance.

*Note: "**I**" means to be implemented toward BASIC conformance. "**II**" means to be implemented in addition to "**I**" towards STANDARD conformance.*

A.1 Schema elements

GeoXACML does not define any new schema elements. Any conformant GeoXACML process MUST be able to prove XACML conformance

A.2 Identifier Prefixes

The following identifier prefixes are reserved by GeoXACML.

Note: Because they are part of the GeoXACML URN definitions, they MUST therefore be implemented in the BASIC conformance class.

Identifier
urn:ogc:def:dataType:geoxacml:1.0:geometry
urn:ogc:def:function:geoxacml:1.0

Any conformant GeoXACML process MUST be able to pass tests as defined in Annex B which include these identifier prefixes.

A.3 Algorithms

GeoXACML does not define any new algorithms.

A.4 Status Codes

GeoXACML does not define any new status codes.

A.5 Data Type

Data Type	Conformance Class
urn:ogc:def:dataType:geoxacml:1.0:geometry	I

A.6 Functions

In order to pass the BASIC conformance level, an implementation MUST process ALL function URNs marked “I” for ALL test cases defined in Annex B.3.

In order to pass the STANDARD conformance level, an implementation MUST process ALL function URNs marked “II” for ALL test cases defined in Annex B.3.

A.6.1 Topological Functions

Functions	Conformance Class
urn:ogc:def:function:geoxacml:1.0:geometry-equals	I
urn:ogc:def:function:geoxacml:1.0:geometry-disjoint	I
urn:ogc:def:function:geoxacml:1.0:geometry-touches	I
urn:ogc:def:function:geoxacml:1.0:geometry-crosses	I
urn:ogc:def:function:geoxacml:1.0:geometry-within	I
urn:ogc:def:function:geoxacml:1.0:geometry-contains	I
urn:ogc:def:function:geoxacml:1.0:geometry-overlaps	I
urn:ogc:def:function:geoxacml:1.0:geometry-intersects	I

Any BASIC level conformant GeoXACML process MUST be able to pass tests as defined in B.5 which include all of the functions above.

A.6.2 Bag Functions

Functions	Conformance Class
urn:ogc:def:function:geoxacml:1.0:geometry-one-and-only	I
urn:ogc:def:function:geoxacml:1.0:geometry-bag-size	I
urn:ogc:def:function:geoxacml:1.0:geometry-is-in	I
urn:ogc:def:function:geoxacml:1.0:geometry-bag	I

Any BASIC level conformant GeoXACML process MUST be able to pass tests as defined in B.5 which include all of the above functions.

A.6.3 Set Functions

Functions	Conformance Class
urn:ogc:def:function:geoxacml:1.0:geometry-bag-intersection	I
urn:ogc:def:function:geoxacml:1.0:geometry-bag-at-least-one-member-of	I
urn:ogc:def:function:geoxacml:1.0:geometry-bag-union	I
urn:ogc:def:function:geoxacml:1.0:geometry-bag-subset	I
urn:ogc:def:function:geoxacml:1.0:geometry-set-equals	I

Any BASIC level conformant GeoXACML process MUST be able to pass tests as defined in B.5 which include all of the functions above.

A.6.4 Geometric Functions

Functions	Conformance Class
urn:ogc:def:function:geoxacml:1.0:geometry-buffer	II
urn:ogc:def:function:geoxacml:1.0:geometry-boundary	II
urn:ogc:def:function:geoxacml:1.0:geometry-convex-hull	II
urn:ogc:def:function:geoxacml:1.0:geometry-centroid	II
urn:ogc:def:function:geoxacml:1.0:geometry-difference	II
urn:ogc:def:function:geoxacml:1.0:geometry-sym-difference	II
urn:ogc:def:function:geoxacml:1.0:geometry-intersection	II
urn:ogc:def:function:geoxacml:1.0:geometry-union	II
urn:ogc:def:function:geoxacml:1.0:geometry-area	II
urn:ogc:def:function:geoxacml:1.0:geometry-distance	II
urn:ogc:def:function:geoxacml:1.0:geometry-is-within-distance	II
urn:ogc:def:function:geoxacml:1.0:geometry-length	II
urn:ogc:def:function:geoxacml:1.0:geometry-is-simple	II
urn:ogc:def:function:geoxacml:1.0:geometry-is-closed	II
urn:ogc:def:function:geoxacml:1.0:geometry-is-valid	II

Any STANDARD level conformant GeoXACML process MUST be able to pass tests as defined in B.5 which include all of the functions above.

A.6.5 Conversion Functions

Functions	Conformance Class
urn:ogc:def:function:geoxacml:1.0:convert-to-metre	I
urn:ogc:def:function:geoxacml:1.0:convert-to-square-metre	I

For the function “urn:ogc:def:function:geoxacml:1.0:convert-to-metre” any BASIC level conformance GeoXACML process SHALL support all possible conversions to metre, as listed in [5], table “LENGTH”.

For the function “urn:ogc:def:function:geoxacml:1.0:convert-to-square-metre” any BASIC level conformance GeoXACML process SHALL support all possible conversions to square metre, as listed in [5], table “AREA AND SECOND MOMENT OF AREA”.

Any BASIC level conformant GeoXACML process MUST be able to pass tests as defined in B.5 which include all of the functions above.

Annex B GeoXACML Conformance Definition and Tests (normative)

This section lists those portions of the specification that SHALL be satisfied in an implementation of a PDP that claims conformance with GeoXACML v1.0.

Because the conformance to GeoXACML relies on the conformance to XACML, it is strongly recommended that the implementation satisfies all tests defined in [14].

B.1 Conformance Class Definition

GeoXACML has two conformance classes:

a) The **BASIC (I)** conformance class, which MUST be satisfied by all PDP implementations, consists of ALL tests for ALL definitions that are label "**I**".
b) The **STANDARD (II)** conformance class MUST satisfy the BASIC class tests and MUST further satisfy ALL tests for ALL definitions that are label "**II**".

B.2 Aspect Definitions

GeoXACML has two different aspect groups for testing:

- **Aspect group A**: Data Type
- **Aspect group B**: Function

The individual aspects for each group are defined in section 9 of this specification and in the used extension(s).

B.3 Conformance Test Case Definitions

Each test in the test suite will be associated to a particular requirement or constraint in this standard (the aspect), and the test format will follow the following pattern:

1. Test Purpose: To determine if the candidate process satisfies a particular aspect of this standard. For BASIC conformance, only BASIC aspects will be included in the test sample instance For STANDARD conformance, any aspects from BASIC AND STANDARD conformance SHALL be included in the test sample instance.

2. Test Method: A GeoXACML instance document using that particular aspect(s) will be present for processing, and the process should respond properly

3. Reference: Each test will reference the aspect to be tested

Aspects to be tested in this specification are GeoXACML data type and function definitions. For each aspect, conformance test cases from the following three different categories MUST be processed:

- Category "**Policy Language Encoding**" (**PLE**) defines conformance test cases that focus on the capability of the implementation being able to properly process the XACML conformant XML encoding for GeoXACML defined geometric data types and functions. These test cases are fundamental and have to be run first.
- Category "**Logical and Syntactical Processing**" (**LSP**) defines conformance test cases to ensure the capability of the implementation to properly understand the syntax of GeoXACML defined functions and geometric data types, as defined in the used extension(s). The test cases also ensure that implementations properly process defined geometric data types and functions according to their logic. These test cases assume that tests from the PLE category have completed with no error.
- Category "**Exception Behaviour and Processing**" (**EBP**) defines conformance test cases that focus on the capability of the implementation being able to properly process error situations of GeoXACML defined functions and geometric data types as defined in the used extension(s). One important aspect of these test cases is to ensure proper and interoperable handling of invalid geometry attribute values or geometric function results. These test cases are to be run at last because they rely on the error free completion of PLE and LSP tests.

A set of test cases for each conformance class and category has been created to assist in the process of testing an implementation towards conformance. These test cases are hosted by OGC and can be located from the GeoXACML Specification page. The site hosting the test cases contains a full description of the test cases and how to execute them.

B.4 PLE Conformance Tests

Test Case	Description	Result
PLE.1	Validate GeoXACML policy file against the XACML policy schema as defined in [2] by using a XML schema validation engine.	Validation succeeds with no errors
PLE.2	PDP to process GeoXACML policy file	PDP does not report any unknown identifiers
PLE.3	PDP to process authorization decision request using AttributeDesignator	PDP does not report any unknown identifiers
PLE.4	PDP to process authorization decision request using AttributeSelector	PDP does not report any unknown identifiers

B.4.1 Towards BASIC Conformance

Create a GeoXACML policy file containing all aspects from group A and group B, marked "**I**".

- Perform PLE.1

- Perform PLE.2

Create a XACML authorization decision request where each request is containing at least one aspect from Group A, marked "**I**".

- Perform PLE.3
- Perform PLE.4

B.4.2 Towards STANDARD Conformance

Create a GeoXACML policy file containing all aspects from group A and group B, marked "**II**".

- Perform PLE.1
- Perform PLE.2

Create a XACML authorization decision request where each request is containing at least one aspect from Group A, marked "**II**".

- Perform PLE.3
- Perform PLE.4

B.5 LSP Conformance Tests

Test Case	Description	Result
LSP.1	Have PDP to process GeoXACML policy file and validate Function result for error-free case	As defined

B.5.1 Towards BASIC Conformance

B.5.1.1 Testing Aspects from Group A

N/A

B.5.1.2 Testing Aspects from Group B

Execute test LSP.1 on each aspect for each valid combination on aspects from Group A, marked as "**I**".

B.5.2 Towards STANDARD Conformance

B.5.2.1 Testing Aspects from Group A

N/A

B.5.2.2 Testing Aspects from Group B

Execute test LSP.1 on each aspect for each valid combination on aspects from Group A, marked as "**II**".

B.6 EBP Conformance Tests

Test Case	Description	Result
EBP.1	PDP to process authorization decision request with **valid** aspect	PDP MUST report no error
EBP.2	PDP to process authorization decision request with **invalid** aspect	PDP MUST report ProcessingError
EBP.3	PDP to process policy file with **valid** aspect	PDP MUST report no error
EBP.4	PDP to process policy file with **invalid** aspect	PDP MUST report ProcessingError

B.6.1 Tests toward BASIC Conformance

B.6.1.1 Testing Aspects from Group A

Execute test EBP.1, EBP.2, EBP.3 and EBP.4 on each aspect, marked as "**I**".

B.6.1.2 Testing Aspects from Group B

Execute test EBP.3 and EBP.4 on each aspect, marked as "**I**".

B.6.2 Tests toward STANDARD Conformance

B.6.2.1 Testing Aspects from Group A

Execute test EBP.1, EBP.2, EBP.3 and EBP.4 on each aspect, marked as "**II**".

B.6.2.2 Testing Aspects from Group B

Execute test EBP.3 and EBP.4 on each aspect, marked as "**II**".

Annex C Example for protecting a OpenGIS® Web Service (informative)

C.1 Example Infrastructure

For the scope of this example let's assume that a Web Map Service (see [12]) or a Web Feature Service (see [13]) exists that shall be protected by access control. The example infrastructure, as illustrated in figure C.1, is based on the distributed access control system as introduced in the XACML standard and is applicable to both services.

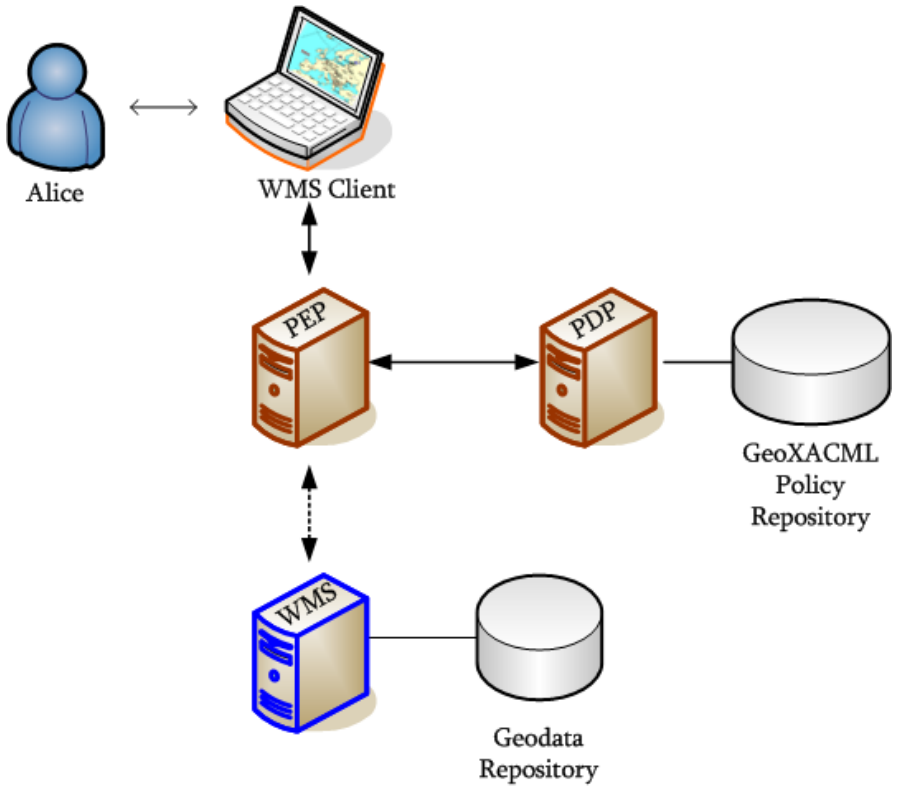

Figure C.1 — Example Infrastructure to protect a OpenGIS® Web Service

C.2 Managing Access to a WMS

The WMS provides different operations of which two are subject for access control: The GetMap operation allows the retrieval of maps and the GetFeatureInfo operation allows the retrieval of additional information, displayed on the map.

C.2.1 Protect access to a particular layer for GetMap operations

In this scenario, Alice can request maps from layer Capitals only.

```
<Rule xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
access_control-xacml-2.0-policy-schema-os.xsd" Effect="Permit"
RuleId="Protection Scenario C.2.1">
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Alice</AttributeValue>
                    <SubjectAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#string"
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Actions>
            <Action>
                <ActionMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">map</AttributeValue>
                    <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
    <Condition>
        <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:all-of">
            <Function
FunctionId="http://www.geoxacml.org/1.0/function#equals"/>
                <AttributeSelector
RequestContextPath="//wms:GetMap/*/sld:NamedLayer/sld:Name"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Capitals</AttributeValue>
        </Apply>
    </Condition>
</Rule>
```

C.2.2 Protect access to a particular area using the GetMap operation

In this scenario, Alice is allowed to obtain maps around the world but not for the area of North and South America. See figure C.2 for the protected area.

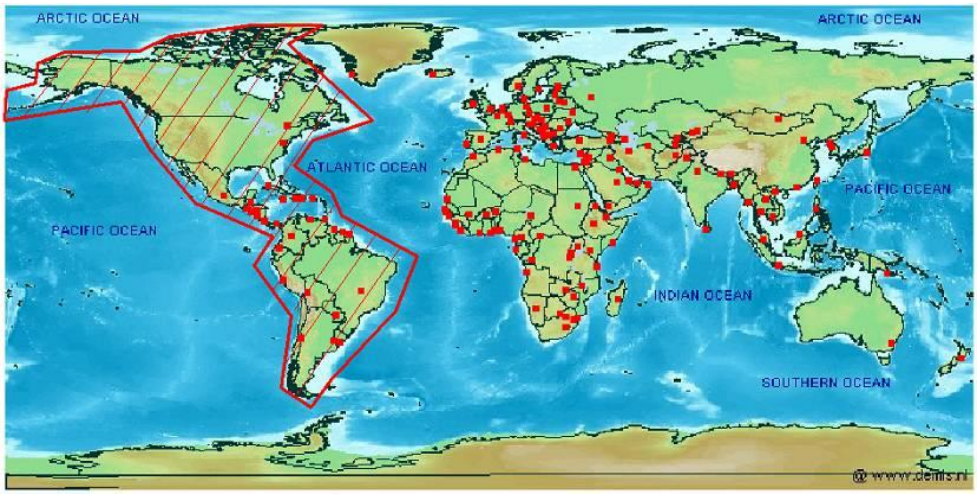

Figure C.2 — Example restricted Area

```
<Rule xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
access_control-xacml-2.0-policy-schema-os.xsd" Effect="Permit"
RuleId="Protection Scenario C.2.2">
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Alice</AttributeValue>
                    <SubjectAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#string"
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Actions>
            <Action>
                <ActionMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">map</AttributeValue>
                    <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
    <Condition>
```

```
        Because the encoding of the condition, using GeoXACML definitions
        depended on the geometry encoding to be used, please
        see Figure C.1 of the associated extension!
    </Condition>
</Rule>
```

C.2.3 Protect access to a particular area using the GetFeatureInfo operation

In this scenario, Alice is allowed to obtain feature information from layer Capitals for parts of Europe only. See figure C.3 for the allowed area.

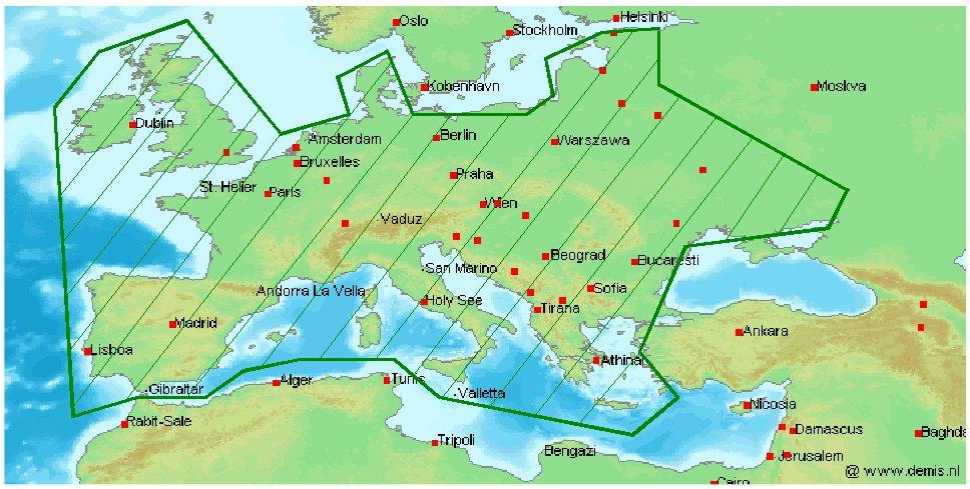

Figure C.3 — Example permitted Area

```
<Rule xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
access_control-xacml-2.0-policy-schema-os.xsd" Effect="Permit"
RuleId="Protection Scenario C.2.3">
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Alice</AttributeValue>
                    <SubjectAttributeDesignator
DataType="http://www.w3.org/2001/XMLSchema#string"
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Actions>
            <Action>
                <ActionMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
```

```
                    <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>

    </Target>
    <Condition>
        Because the encoding of the condition, using GeoXACML definitions
        depended on the geometry encoding to be used, please
        see Figure C.2 of the associated extension!
    </Condition>
</Rule>
```

C.3 Managing Access to a WFS

The (transactional) WFS provides multiple operations which are candidates for applying access control: The GetFeature operation allows retrieval of feature information encoded in GML; the Transaction operation allows for the insertion, modification or deletion of features through the WFS interface.

C.3.1 Protect access to particular feature types using the GetFeature operation

In this scenario, Alice is allowed to obtain feature instances only of type ows4:Road_L and ows4:River_L.

```
<Rule xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
access_control-xacml-2.0-policy-schema-os.xsd" Effect="Permit" RuleId="
Protection Scenario C.3.1">
    <Description>Alice can make GetFeature requests for ows4:Road_L and
ows4:River_L</Description>
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Alice</AttributeValue>
                    <SubjectAttributeDesignator
SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-
subject" AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Resources>
            <Resource>
                <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:integer-less-than">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">0</AttributeValue>
                    <AttributeSelector
RequestContextPath="count(//wfs:Query[@typeName='ows4:Road_L'])"
DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                </ResourceMatch>
            </Resource>
            <Resource>
                <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:integer-less-than">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">0</AttributeValue>
                    <AttributeSelector
RequestContextPath="count(//wfs:Query[@typeName='ows4:River_L'])"
DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                </ResourceMatch>
            </Resource>
        </Resources>
        <Actions>
            <Action>
                <ActionMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
```

```
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">read</AttributeValue>
                    <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
</Rule>
```

C.3.2 Protect access to a particular area and feature types using the Transaction operation

In this scenario, Alice is allowed to insert features for the area around the airport (see figure C.4) for features of type ows4:Helipad_P
only.

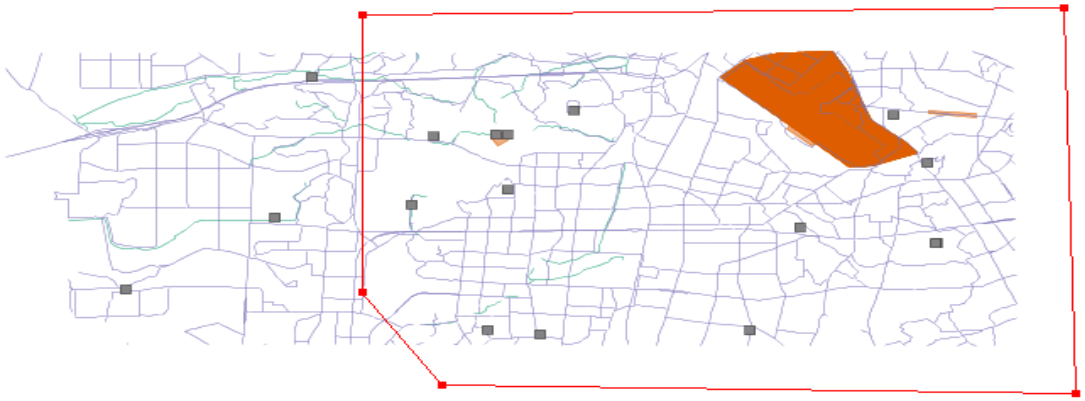

Figure C.4: Access Control Area[2] for inserting helipad feature information

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
access_control-xacml-2.0-policy-schema-os.xsd" Effect="Permit"
RuleId="Protection Scenario C.3.2">
```

[2] (-74.96789132745889, 39.383275615837945), (-74.96789132745889, 39.296675134185634), (-74.94733464747071, 39.268245683138154), (-74.78331858373527, 39.265621426118386), (-74.78638021692498, 39.38546249668775), (-74.96789132745889, 39.383275615837945), (-74.96789132745889, 39.383275615837945), (-74.96789132745889, 39.383275615837945)

```
    <Description>Alice can Insert features of type ows4:HeliPad_P2 WITHIN area
around the airport</Description>
    <Target>
        <Subjects>
            <Subject>
                <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">Alice</AttributeValue>
                    <SubjectAttributeDesignator
SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </SubjectMatch>
            </Subject>
        </Subjects>
        <Resources>
            <Resource>
                <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:integer-less-than">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#integer">0</AttributeValue>
                    <AttributeSelector
RequestContextPath="count(//wfs:Insert/ows4:HeliPad_P2)"
DataType="http://www.w3.org/2001/XMLSchema#integer"/>
                </ResourceMatch>
            </Resource>
        </Resources>
        <Actions>
            <Action>
                <ActionMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                    <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">insert</AttributeValue>
                    <ActionAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
DataType="http://www.w3.org/2001/XMLSchema#string"/>
                </ActionMatch>
            </Action>
        </Actions>
    </Target>
    <Condition>
        Because the encoding of the condition, using GeoXACML definitions
        depended on the geometry encoding to be used, please
        see Figure D.3 of the associated extension!
    </Condition>
</Rule>
```

Bibliography

[6] OASIS, *Core and hierarchical role based access control (RBAC) profile of XACML v2.0*, 2005-02-01, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-rbac-profile1-spec-os.pdf

[7] OASIS, *SAML 2.0 profile of XACML v2.0*, 2005-02-01, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-saml-profile-spec-os.pdf

[8] Cover Pages, Extensible Access Control Markup Language (XACML), http://xml.coverpages.org/xacml.html

[9] OGC, *The OpenGIS® Abstract Specification Topic 18: Geospatial Digital Rights Management Reference Model (GeoDRM RM), Version: 1.0.0, 2006-12-29*, http://portal.opengeospatial.org/files/?artifact_id=17802

[10]Vivid Solutions, *JTS Topology Suite Technical Specifications, Version 1.4*, http://www.vividsolutions.com/JTS/bin/JTS%20Technical%20Specs.pdf

[11] OGC, *OpenGIS® Web Service Common Implementation Specification, Version 1.0.0, 2005-11-22*, http://portal.opengeospatial.org/files/?artifact_id=8798

[12] OGC, *OpenGIS® Web Map Service Implementation Specification, Version 1.3.0, 2004-01-20*, http://portal.opengeospatial.org/files/?artifact_id=4756

[13] OGC, *OpenGIS® Web Feature Service Implementation Specification, Version 1.1.0, 2005-05-03*, http://portal.opengeospatial.org/files/?artifact_id=8339

[14] OASIS, *Conformance Tests for XACML 1.0 and 1.1, 25 March 2004*, http://www.oasis-open.org/committees/download.php/6076/ConformanceTests.zip

[15] OGC, *Definition identifier URNs in OGC namespace, Version 1.1.0, 2006-08-08*, http://portal.opengeospatial.org/files/?artifact_id=16339