XACML v3.0 Time Extensions Version 1.0

Committee Specification 01

13 February 2020

This stage:
https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/cs01/xacml-3.0-time-extensions-v1.0-cs01.docx (Authoritative)
https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/cs01/xacml-3.0-time-extensions-v1.0-cs01.html
https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/cs01/xacml-3.0-time-extensions-v1.0-cs01.pdf

Previous stage:
N/A

Latest stage:
https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/xacml-3.0-time-extensions-v1.0.docx (Authoritative)
https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/xacml-3.0-time-extensions-v1.0.html
https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/xacml-3.0-time-extensions-v1.0.pdf

Technical Committee:

OASIS eXtensible Access Control Markup Language (XACML) TC

Chairs:
Hal Lochhart (harold.w.lochhart@gmail.com), Individual member
Bill Parducci (bill@parducci.net), Individual member

Editor:
Steven Legg (steven.legg@viewds.com), ViewDS Identity Solutions

Related work:
This specification is related to:
- *eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01.* Edited by Erik Rissanen. 12 July 2017. OASIS Standard incorporating Approved Errata. http://docs.oasis-open.org/xacml/3.0/errata01/os/xacml-3.0-core-spec-errata01-os-complete.html. Latest stage: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-en.html.

Abstract:
This profile defines XACML functions for comparing time values that are not sensitive to the time zone chosen for those values, defines functions for performing arithmetic on date and time values and defines a data-type for representing the day of the week along with functions to operate on values of the data-type.

Status:

Citation format:

When referencing this specification the following citation format should be used:

[xacml-time-ext-v1.0]

XACML v3.0 Time Extensions Version 1.0. Edited by Steven Legg. 13 February 2020. OASIS Committee Specification 01. https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/cs01/xacml-3.0-time-extensions-v1.0-cs01.html. Latest stage: https://docs.oasis-open.org/xacml/xacml-3.0-time-extensions/v1.0/xacml-3.0-time-extensions-v1.0.html.

Notices

Table of Contents

1 Introduction

[All text is normative unless otherwise labeled]

1.1 Overview

{Non-normative}

The time functions defined by the XACML core specification [XACML3] have limited utility when used in widely distributed and replicated environments where times are presented with various, different time zones. This may be because the current time is generated according to the time zone in which an XACML component is running and the components are distributed and replicated across various time zones. In the most general case, the location of the XACML service that evaluates a request is unpredictable and uncontrollable by clients and changes from one request to the next.

This document demonstrates the difficulties in using the previously defined time functions with varying time zones and defines new functions that are not sensitive to the choice of time zone.

The core specification defines functions for performing arithmetic on dateTime values, but not for time and date values. This document defines such functions for time and date.

In addition to controlling access according to the time of day, it is not unreasonable for a policy writer to want to control access according to the day of the week. This document defines a new data-type to represent the day of the week with an optional time zone, and new functions to operate on values of the new data-type.

1.2 Glossary

Context handler

The system component that, among other things, may add attribute values to an authorization request, in particular, attribute values for the current date and time of day.

dayOfWeek

An XACML data-type defined in this document for representing a day of the week with an optional time zone.

Policy administration point (PAP)

The system component that creates authorization policies.

Policy decision point (PDP)

The system component that evaluates an authorization request and renders the authorization decision.

Policy enforcement point (PEP)

The system component that makes an authorization request and enforces the authorization decision.

Policy information point (PIP)

The system component that acts as a source of attribute values.

Reference date

The date of an arbitrarily chosen Sunday to be used in converting timeOfDay values into dateTime values. An implementation is free to choose any Sunday. The examples in this document use `2017-01-15` as the ***reference date***.

Resource

The entity being accessed.

Subject

The entity requesting access.

1.3 IPR Policy

This specification is provided under the RF on Limited Terms Mode of the OASIS IPR Policy, the mode chosen when the Technical Committee was established. For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (https://www.oasis-open.org/committees/xacml/ipr.php).

1.4 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.5 Normative References

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.
[RFC5234]	Crocker, D., Ed., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>
[RFC8174]	Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <http://www.rfc-editor.org/info/rfc8174>.
[XACML3]	*eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01.* Edited by Erik Rissanen. 12 July 2017. OASIS Standard incorporating Approved Errata. http://docs.oasis-open.org/xacml/3.0/errata01/os/xacml-3.0-core-spec-errata01-os-complete.html. Latest stage: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-en.html.
[XACMLJSON]	*JSON Profile of XACML 3.0 Version 1.1.* Edited by David Brossard and Steven Legg. 20 June 2019. OASIS Standard. https://docs.oasis-open.org/xacml/xacml-json-http/v1.1/os/xacml-json-http-v1.1-os.html. Latest stage: https://docs.oasis-open.org/xacml/xacml-json-http/v1.1/xacml-json-http-v1.1.html.
[XML]	Extensible Markup Language (XML) 1.0 (Fifth Edition), T. Bray, J. Paoli, M. Sperberg-McQueen, E. Maler, F. Yergeau, Editors, W3C Recommendation, November 26, 2008, http://www.w3.org/TR/2008/REC-xml-20081126/. Latest version available at http://www.w3.org/TR/xml/.
[XSD2]	XML Schema Part 2: Datatypes Second Edition, Paul V. Biron, A. Malhotra, Editors, W3C Recommendation, October 28, 2004, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/. Latest version available at http://www.w3.org/TR/xmlschema-2/.
[INFOSET]	XML Information Set, J. Cowan, R. Tobin, Editors, W3C Recommendation, October 24, 2001, http://www.w3.org/TR/2001/REC-xml-infoset-20011024/. Latest version available at http://www.w3.org/TR/xml-infoset/.

1.6 Non-Normative References

[ISO8601]	ISO 8601:2004, Data elements and interchange formats – Information interchange – Representation of dates and times.
[DOOMSDAY]	Doomsday rule. https://en.wikipedia.org/wiki/Doomsday_rule.
[ENTITIES]	*XACML v3.0 Related and Nested Entities Profile Version 1.0*, Edited by Steven Legg. 25 October 2015. Committee Specification 01. http://docs.oasis-

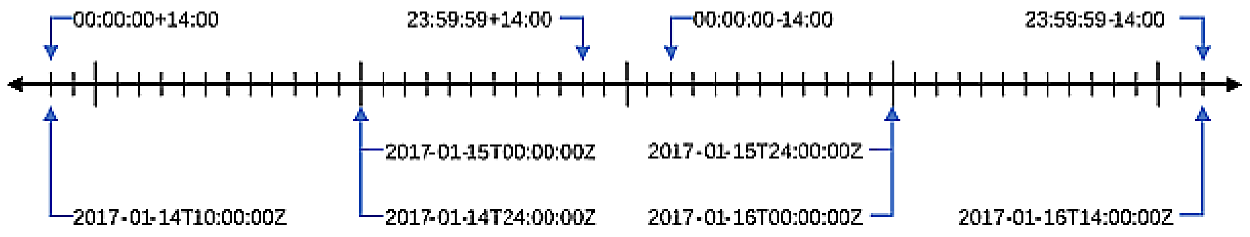
open.org/xacml/xacml-3.0-related-entities/v1.0/cs01/xacml-3.0-related-entities-v1.0-cs01.html. Latest stage: http://docs.oasis-open.org/xacml/xacml-3.0-related-entities/v1.0/xacml-3.0-related-entities-v1.0.html.

2 Background

{Non-normative}

The existing XACML time functions [XACML3] compare time values by first converting the time values to dateTime values using an arbitrarily chosen ***reference date***, then normalizing the dateTime values to UTC and comparing them (the conversion and normalization is the same as that described in Section 3.1). The effect of the conversion and normalization of time values is to map the time values into a 52 hour range of dateTime values centered on the ***reference date***, as illustrated in Figure 1.

Figure 1 - Range for converted and normalized time values.



The least possible time value is `00:00:00+14:00`. Conversion to a dateTime value using the ***reference date*** gives `2017-01-15T00:00:00+14:00`, which normalizes to `2017-01-14T10:00:00Z`. The greatest possible time value is within a second of `23:59:59-14:00`. Conversion to a dateTime value using the ***reference date*** gives `2017-01-15T23:59:59-14:00`, which normalizes to `2017-01-16T13:59:59Z`.

Note that the time `24:00:00` in a dateTime value represents the first instant of the next day. Thus `2017-01-15T24:00:00Z` is the same instant as `2017-01-16T00:00:00Z`. However, the time values `00:00:00` and `24:00:00` are different lexical representations for the same value in the value space for time values, i.e., `00:00:00`. The examples in this document use the time value `23:59:59` to stand in for an instant infinitesimally close to midnight at the end of a day.

Observe that the 24 hour interval beginning at `00:00:00+14:00` and the 24 hour interval ending at `23:59:59-14:00` do not overlap on the dateTime time line.

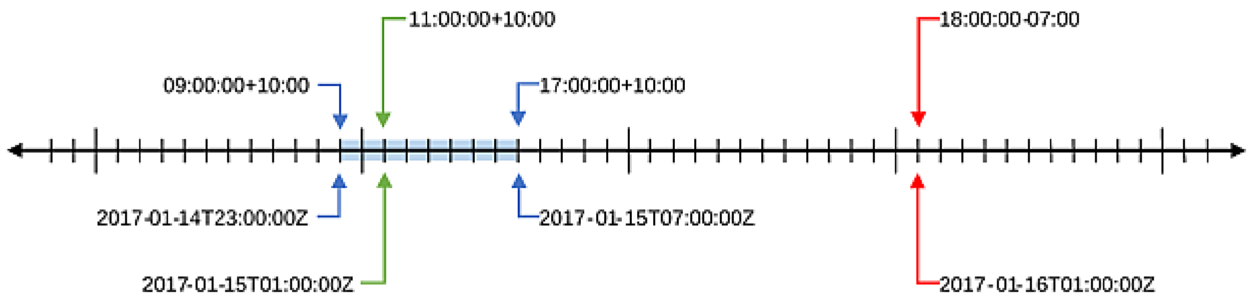
The mapping of time values into an extended range allows for sensible comparisons of times that are specified in the same time zone, regardless of what that might be, but presents difficulties in writing XACML policies that attempt to compare times that may be specified using different time zones. This situation may arise, for example, in a cloud-based authorization service (or a cloud-based service that uses XACML for authorization) where there are multiple instances of ***PDPs*** and their associated ***context handlers*** running in different data centers possibly in different time zones. It is possible for ***PEPs*** to supply explicit values for the current time environment variable and the applications containing the ***PEPs*** may also be hosted in the cloud and be similarly dispersed across different data centers in different time zones. Even ***context handlers*** or ***PEPs*** operating in the same time zone might reasonably choose to use either the local time zone or UTC.

To illustrate the potential problems, consider the following XACML expression to test whether the current time is within the range `09:00:00+10:00` to `17:00:00+10:00`, i.e., “business hours” in Australian Eastern Standard Time.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId="urn:oasis:names:tc:xacml:2.0:function:time-in-range">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"
      DataType="http://www.w3.org/2001/XMLSchema#time"
      MustBePresent="false"/>
  </Apply>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >09:00:00+10:00</AttributeValue>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >17:00:00+10:00</AttributeValue>
</Apply>
```

The time value `09:00:00+10:00` maps to the dateTime value `2017-01-14T23:00:00Z` and the time value `17:00:00+10:00` maps to the dateTime value `2017-01-15T07:00:00Z` using the chosen ***reference date***. Suppose that the current time of day generated by the ***context handler*** is `11:00:00+10:00`, which could also be expressed as `18:00:00-07:00` (Pacific Daylight Time), among many other possibilities. Critically, the result of the XACML expression is sensitive to which way the current time is expressed. The time value `11:00:00+10:00` maps to the dateTime value `2017-01-15T01:00:00Z`, which is clearly between `2017-01-14T23:00:00Z` and `2017-01-15T07:00:00Z`. However, the time value `18:00:00-07:00` maps to the dateTime value `2017-01-16T01:00:00Z`, which is greater than the end point of the range. See Figure 2.

Figure 2 - Effect of time zone choice



One solution to this problem of equivalent time values giving different results would be to insist that all implementations of ***context handlers*** and ***PEPs*** and all time values in policies use the same time zone, e.g., UTC. However, existing implementations have not been so constrained, nor have they been required to be configurable as to which time zone they should use, so rather than retrospectively imposing such requirements this document defines new functions for comparing time values such that values representing the same time of day, though using different time zones, produce consistent results.

3 Time Functions

This section defines functions for comparing, and performing arithmetic on, time values. The functions are defined using concepts and procedures referenced by the definitions of the pre-existing time functions [**XACML3**], however this is not necessarily the optimal way to implement them. Implementations are free to use any method that produces the same results.

3.1 Converting time to dateTime

This section defines a common procedure for converting a given time value into a dateTime value normalized to UTC.

Follow these steps:

1. Create a new dateTime value where the date components, i.e., year, month and day, are set to the values of the same components in the ***reference date***, and the time components, i.e., hour, minute, second and fractional seconds, are set to the values of the same components in the given time value. Set the time zone to UTC.
2. Convert the time zone of the given time value into a dayTimeDuration value with the opposite sign and the same number of hours and minutes. Zero-valued components may be omitted. For example, the time zone `+10:00` becomes `-PT10H`, `+09:30` becomes `-PT9H30M` and `-07:00` becomes `PT7H`.
3. Add the dayTimeDuration value from step 2 to the dateTime value from step 1 according to the specification for adding durations to dateTime values, [**XSD2**] Appendix E, and return the result.

3.2 The time-in-recurring-range Function

The time-in-recurring-range function tests whether one time value falls within a range, given by two other time values, that repeats daily. It is identified by the URI `urn:oasis:names:tc:xacml:3.0:function:time-in-recurring-range`.

This function SHALL take three arguments of data-type `http://www.w3.org/2001/XMLSchema#time` and SHALL return an `http://www.w3.org/2001/XMLSchema#boolean`. If no time zone is provided for the first argument, it SHALL use the default time zone at the ***context handler***. If no time zone is provided for the second argument, then it SHALL use the same time zone as the first argument. If no time zone is provided for the third argument, then it SHALL use the same time zone as the first argument. Each of the three arguments is then converted to a dateTime value according to the procedure in Section 3.1.

The second argument converted to a dateTime value defines a series of dateTime start points for recurring ranges where the start points have the same time of day and every possible date (in practice it is only necessary to consider two days either side of the ***reference date***).

The third argument converted to a dateTime value defines a series of dateTime end points for the recurring ranges where the end points have the same time of day and every possible date.

If any argument evaluates to “Indeterminate”, then the function evaluates to “Indeterminate”; otherwise, the function returns “True” if the first argument converted to a dateTime value is greater than or equal to one of the start points and less than or equal to the end point that is greater than or equal to that start point by less than 24 hours (i.e., the closest end point greater than or equal to the start point); otherwise, the function returns “False”. The dateTime values are compared according to the algorithm defined in [**XSD2**], section 3.2.7.4.

3.2.1 Example 1

{Non-normative}

Consider the following XACML expression for testing whether the current time is in the range `09:00:00+10:00` to `17:00:00+10:00`.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId="urn:oasis:names:tc:xacml:3.0:function:time-in-recurring-range">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"
      DataType="http://www.w3.org/2001/XMLSchema#time"
      MustBePresent="false"/>
  </Apply>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >09:00:00+10:00</AttributeValue>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >17:00:00+10:00</AttributeValue>
</Apply>
```

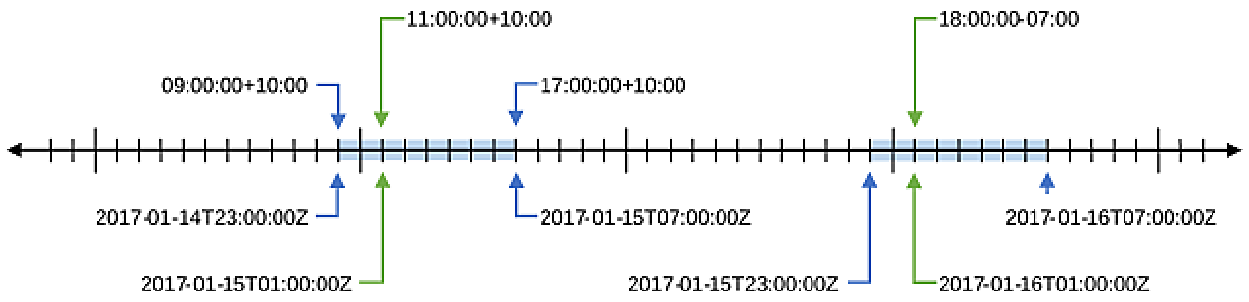
The start point of the range is `09:00:00+10:00`, which maps to `2017-01-14T23:00:00Z`. This determines a sequence of daily start points around the ***reference date***, e.g., `2017-01-13T23:00:00Z`, `2017-01-14T23:00:00Z`, `2017-01-15T23:00:00Z`, `2017-01-16T23:00:00Z` and `2017-01-17T23:00:00Z`.

The end point of the range is `17:00:00+10:00`, which maps to `2017-01-15T07:00:00Z`. This determines a sequence of daily end points around the ***reference date***, e.g., `2017-01-13T07:00:00Z`, `2017-01-14T07:00:00Z`, `2017-01-15T07:00:00Z`, `2017-01-16T07:00:00Z` and `2017-01-17T07:00:00Z`.

Suppose that the current time of day generated by the ***context handler*** is `11:00:00+10:00`, which maps to the dateTime value `2017-01-15T01:00:00Z`. In this case the time-in-recurring-range function returns “True” because `2017-01-15T01:00:00Z` is greater than the start point `2017-01-14T23:00:00Z` and less than the next greater end point of `2017-01-15T07:00:00Z`.

Suppose that the current time of day generated by the ***context handler*** is instead `18:00:00-07:00`, which maps to the dateTime value `2017-01-16T01:00:00Z`. In this case the time-in-recurring-range function also returns “True” because `2017-01-16T01:00:00Z` is greater than the start point `2017-01-15T23:00:00Z` and less than the next greater end point of `2017-01-16T07:00:00Z`. See Figure 3.

Figure 3 - Start point less than end point



3.2.2 Example 2

{Non-normative}

Consider the following XACML expression for testing whether the current time is in the range `17:00:00+10:00` to `09:00:00+10:00`, i.e., outside “business hours”.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId="urn:oasis:names:tc:xacml:3.0:function:time-in-recurring-range">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"
      DataType="http://www.w3.org/2001/XMLSchema#time"
      MustBePresent="false"/>
  </Apply>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >17:00:00+10:00</AttributeValue>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >09:00:00+10:00</AttributeValue>
</Apply>
```

The range could be read as "from 5:00pm today until 9:00am tomorrow", or "from 5:00pm yesterday until 9:00am today". Since the range recurs, both statements are valid characterizations.

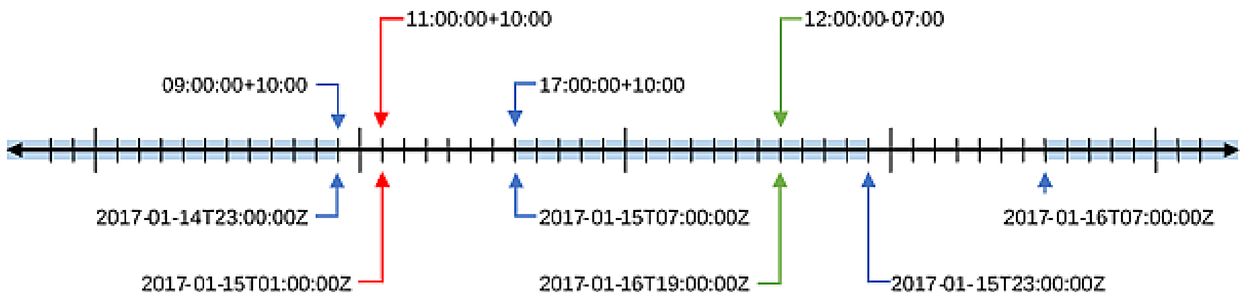
The start point of the range is `17:00:00+10:00`, which maps to `2017-01-15T07:00:00Z`. This determines a sequence of daily start points around the ***reference date***, e.g., `2017-01-13T07:00:00Z`, `2017-01-14T07:00:00Z`, `2017-01-15T07:00:00Z`, `2017-01-16T07:00:00Z` and `2017-01-17T07:00:00Z`.

The end point of the range is `09:00:00+10:00`, which maps to `2017-01-14T23:00:00Z`. This determines a sequence of daily end points around the ***reference date***, e.g., `2017-01-13T23:00:00Z`, `2017-01-14T23:00:00Z`, `2017-01-15T23:00:00Z`, `2017-01-16T23:00:00Z` and `2017-01-17T23:00:00Z`.

Suppose that the current time of day generated by the ***context handler*** is `11:00:00+10:00`, which maps to the dateTime value `2017-01-15T01:00:00Z`. In this case the time in recurring range function returns "False" because `2017-01-15T01:00:00Z` is greater than both the start point `2017-01-14T07:00:00Z` and the next greater end point of `2017-01-14T23:00:00Z`. The start point `2017-01-14T07:00:00Z` is the greatest start point that is still less than or equal to `2017-01-15T01:00:00Z`.

At another time, suppose that the current time of day generated by the ***context handler*** is `12:00:00-07:00`, which maps to the dateTime value `2017-01-15T19:00:00Z`. In this case the time-in-recurring-range function returns "True" because `2017-01-15T19:00:00Z` is greater than the start point `2017-01-15T07:00:00Z` and less than the next greater end point of `2017-01-15T23:00:00Z`. See Figure 4.

Figure 4 - End point less than the start point



3.2.3 Implementation

{Non-normative}

Implementations of the time-in-recurring-range function are free to use any method that produces the same results. Here is a simple way to evaluate the function:

1. If any argument evaluates to “Indeterminate”, then return “Indeterminate”.
2. Convert each of the three arguments to a dateTime value according to the procedure in Section 3.1.
3. In the comparisons that follow, either:
    a. reset the date part of the dateTime values to the ***reference date*** and compare the values in accordance with the algorithm defined in [**XSD2**], section 3.2.7.4., or
    b. directly compare only the time fields of the dateTime values, ignoring the date and time zone fields.
4. If the end point is greater than or equal to the start point and the first argument is greater than or equal to the start point and less than or equal to the end point, then return “True”.
5. If the end point is less than the start point and the first argument is less than or equal to the end point *or* greater than or equal to the start point, then return “True”.
6. Otherwise, return “False”.

3.3 The recurring-time-equal Function

The recurring-time-equal function tests whether one time value is equal to another time value that repeats daily. It is identified by the URI
`urn:oasis:names:tc:xacml:3.0:function:recurring-time-equal`.

This function SHALL take two arguments of data-type `http://www.w3.org/2001/XMLSchema#time` and SHALL return an `http://www.w3.org/2001/XMLSchema#boolean`. If no time zone is provided for the first argument, it SHALL use the default time zone at the ***context handler***. If no time zone is provided for the second argument, then it SHALL use the same time zone as the first argument. Both of the arguments are then converted to a dateTime value according to the procedure in Section 3.1.

The second argument converted to a dateTime value defines a series of dateTime values with the same time of day and every possible date (in practice it is only necessary to consider two days either side of the ***reference date***).

If either argument evaluates to “Indeterminate”, then the function evaluates to “Indeterminate”; otherwise, the function returns “True” if the first argument converted to a dateTime value is equal to one of the series of dateTime values defined by the second argument; otherwise, the function returns “False”. The dateTime values are compared according to the algorithm defined in [**XSD2**], section 3.2.7.4.

3.3.1 Implementation

{Non-normative}

Implementations of the recurring-time-equal function are free to use any method that produces the same results. Here is a simple way to evaluate the function:

1. If either argument evaluates to “Indeterminate”, then return “Indeterminate”.
2. Convert both arguments to a dateTime value according to the procedure in Section 3.1.
3. In the comparison that follows, either:
    a. reset the date part of the dateTime values to the ***reference date*** and compare the values in accordance with the dateTime equal function [**XACML3**], or
    b. directly compare only the time fields of the dateTime values, ignoring the date and time zone fields.
4. If the first argument is equal to the second argument, then return “True”; otherwise, return “False”.

3.4 The time-add-dayTimeDuration Function

The `urn:oasis:names:tc:xacml:3.0:function:time-add-dayTimeDuration` function adds a duration to a time value.

This function SHALL take two arguments, the first SHALL be of data-type `http://www.w3.org/2001/XMLSchema#time` and the second SHALL be of data-type `http://www.w3.org/2001/XMLSchema#dayTimeDuration`. It SHALL return a result of data-type `http://www.w3.org/2001/XMLSchema#time`. The second argument MAY be a negative duration.

If either argument evaluates to "Indeterminate", then the function evaluates to "Indeterminate"; otherwise, the function returns the time value calculated as follows:

1. The first argument is converted to a dateTime value by setting the date fields to the ***reference date***. The dateTime value MUST NOT be normalized to UTC.
2. The second argument is added to the dateTime value according to the specification for adding durations to dateTime values [**XSD2**] Appendix E.
3. The result of the previous step is converted to a time value by discarding the date fields. The result MUST use the same time zone as the first argument (or have no time zone if the first argument has no time zone). Note that the algorithm for adding durations preserves the original time zone information.

The second argument MAY have a non-zero value for the days field, however this field will have no effect on the result of the function.

3.4.1 Implementation

{Non-normative}

Implementations of the time-add-dayTimeDuration function can be optimized by skipping the calculation of the day, month and year fields since they are ultimately discarded.

3.5 The time-subtract-dayTimeDuration Function

The `urn:oasis:names:tc:xacml:3.0:function:time-subtract-dayTimeDuration` function subtracts a duration from a time value.

This function SHALL take two arguments, the first SHALL be of data-type `http://www.w3.org/2001/XMLSchema#time` and the second SHALL be of data-type `http://www.w3.org/2001/XMLSchema#dayTimeDuration`. It SHALL return a result of data-type `http://www.w3.org/2001/XMLSchema#time`. The second argument MAY be a negative duration.

If either argument evaluates to "Indeterminate", then the function evaluates to "Indeterminate"; otherwise, the function returns the time value calculated as follows:

1. The first argument is converted to a dateTime value by setting the date fields to the ***reference date***. The dateTime value MUST NOT be normalized to UTC.
2. If the second argument is a positive duration, then the corresponding negative duration is added to the dateTime value according to the specification for adding durations to dateTime values [**XSD2**] Appendix E. Otherwise (the second argument is a negative duration), the corresponding positive duration is added to the dateTime value.
3. The result of the previous step is converted to a time value by discarding the date fields. The result MUST use the same time zone as the first argument (or have no time zone if the first argument has no time zone). Note that the algorithm for adding durations preserves the original time zone information.

The second argument MAY have a non-zero value for the days field, however this field will have no effect on the result of the function.

4 The time-zone Attribute

A policy writer may wish to restrict access to a particular time range in the local time of the ***subject*** or ***resource***, whatever that might be, rather than tying access to local time in a specific time zone. The time-zone attribute is defined to support such restrictions when the time zone in which a policy is *evaluated* cannot be controlled with any certainty.

The `urn:oasis:names:tc:xacml:3.0:entity:time-zone` attribute indicates the time zone at the location of the entity containing the attribute, e.g., the ***subject*** or ***resource***. The time zone SHOULD be represented as a single value of the `http://www.w3.org/2001/XMLSchema#dayTimeDuration` data-type, where the duration is the difference between UTC and the current time zone of the entity. The duration value SHOULD NOT have non-zero components for days, seconds or fractional seconds, and the absolute value SHOULD NOT exceed 14 hours. The sign of the value MUST be consistent with the usual representation of the time zone in a dateTime value. For example, if the entity is in the Australian Eastern Standard Time time zone then the duration would be `PT10H`, and if the entity is in the Pacific Daylight Time time zone then the duration would be `-PT7H`.

4.1 Example 1

{Non-normative}

Consider the following XACML expression for testing whether the current local time of the ***subject*** is in the range `09:00:00` to `17:00:00`.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId="urn:oasis:names:tc:xacml:3.0:function:time-in-recurring-range">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"
      DataType="http://www.w3.org/2001/XMLSchema#time"
      MustBePresent="false"/>
  </Apply>
  <Apply FunctionId=
      "urn:oasis:names:tc:xacml:3.0:function:time-add-dayTimeDuration">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
      >09:00:00Z</AttributeValue>
    <Apply FunctionId=
        "urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-one-and-only">
      <AttributeDesignator
        Category=
          "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
        AttributeId="urn:oasis:names:tc:xacml:3.0:entity:time-zone"
        DataType="http://www.w3.org/2001/XMLSchema#dayTimeDuration"
        MustBePresent="false"/>
    </Apply>
  </Apply>
  <Apply FunctionId=
      "urn:oasis:names:tc:xacml:3.0:function:time-add-dayTimeDuration">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
      >17:00:00Z</AttributeValue>
    <Apply FunctionId=
        "urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-one-and-only">
      <AttributeDesignator
        Category=
          "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
        AttributeId="urn:oasis:names:tc:xacml:3.0:entity:time-zone"
        DataType="http://www.w3.org/2001/XMLSchema#dayTimeDuration"
        MustBePresent="false"/>
    </Apply>
  </Apply>
</Apply>
```

Table 1 summarizes the result of evaluating this expression with different values of the ***subject*** time-zone attribute and different values and representations of the current time. The Effective Start Point column indicates the value of the second argument to the time-in-recurring-range function (the start of the range), i.e., `09:00:00Z` plus the ***subject's*** time zone offset. The Effective End Point column indicates the value of the third argument to the time-in-recurring-range function (the end of the range), i.e., `17:00:00Z` plus the ***subject's*** time zone offset.

Current Time	Subject Time Zone	Effective Start Point	Effective End Point	Result
`12:00:00+10:00` `02:00:00Z`	`PT10H`	`09:00:00+10:00` `23:00:00Z`	`17:00:00+10:00` `07:00:00Z`	True
`19:00:00-07:00` `02:00:00Z`	`PT10H`	`09:00:00+10:00` `23:00:00Z`	`17:00:00+10:00` `07:00:00Z`	True
`12:00:00+10:00` `02:00:00Z`	`-PT7H`	`09:00:00-07:00` `16:00:00Z`	`17:00:00-07:00` `00:00:00Z`	False
`19:00:00-07:00` `02:00:00Z`	`-PT7H`	`09:00:00-07:00` `16:00:00Z`	`17:00:00-07:00` `00:00:00Z`	False
`05:00:00+10:00` `19:00:00Z`	`PT10H`	`09:00:00+10:00` `23:00:00Z`	`17:00:00+10:00` `07:00:00Z`	False
`12:00:00-07:00` `19:00:00Z`	`PT10H`	`09:00:00+10:00` `23:00:00Z`	`17:00:00+10:00` `07:00:00Z`	False
`05:00:00+10:00` `19:00:00Z`	`-PT7H`	`09:00:00-07:00` `16:00:00Z`	`17:00:00-07:00` `00:00:00Z`	True
`12:00:00-07:00` `19:00:00Z`	`-PT7H`	`09:00:00-07:00` `16:00:00Z`	`17:00:00-07:00` `00:00:00Z`	True

Table 1 - Results using the time-zone attribute for a subject

An equivalent and more-efficient but less intuitive way to express the same condition is:

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId="urn:oasis:names:tc:xacml:3.0:function:time-in-recurring-range">
  <Apply FunctionId=
      "urn:oasis:names:tc:xacml:3.0:function:time-subtract-dayTimeDuration">
    <Apply FunctionId=
        "urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
      <AttributeDesignator
        Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
        AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"
        DataType="http://www.w3.org/2001/XMLSchema#time"
        MustBePresent="false"/>
    </Apply>
    <Apply FunctionId=
        "urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-one-and-only">
      <AttributeDesignator
        Category=
          "urn:oasis:names:tc:xacml:1.0:subject-category:access-subject"
        AttributeId="urn:oasis:names:tc:xacml:3.0:entity:time-zone"
        DataType="http://www.w3.org/2001/XMLSchema#dayTimeDuration"
        MustBePresent="false"/>
    </Apply>
  </Apply>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >09:00:00Z</AttributeValue>
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
    >17:00:00Z</AttributeValue>
</Apply>
```

4.2 Example 2

{Non-normative}

The following XACML expression tests whether the current local time of the ***resource*** is in the range `08:00:00` to `18:00:00`. Such an expression might be used in a policy to control physical access to a physical resource in a fixed location, for example, for controlling electronic locks on a building, a room or a vault.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId="urn:oasis:names:tc:xacml:3.0:function:time-in-recurring-range">
  <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:time-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-time"
      DataType="http://www.w3.org/2001/XMLSchema#time"
      MustBePresent="false"/>
  </Apply>
  <Apply FunctionId=
      "urn:oasis:names:tc:xacml:3.0:function:time-add-dayTimeDuration">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
      >08:00:00Z</AttributeValue>
    <Apply FunctionId=
        "urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-one-and-only">
      <AttributeDesignator
        Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
        AttributeId="urn:oasis:names:tc:xacml:3.0:entity:time-zone"
        DataType="http://www.w3.org/2001/XMLSchema#dayTimeDuration"
        MustBePresent="false"/>
    </Apply>
  </Apply>
  <Apply FunctionId=
      "urn:oasis:names:tc:xacml:3.0:function:time-add-dayTimeDuration">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
      >18:00:00Z</AttributeValue>
    <Apply FunctionId=
        "urn:oasis:names:tc:xacml:1.0:function:dayTimeDuration-one-and-only">
      <AttributeDesignator
        Category="urn:oasis:names:tc:xacml:3.0:attribute-category:resource"
        AttributeId="urn:oasis:names:tc:xacml:3.0:entity:time-zone"
        DataType="http://www.w3.org/2001/XMLSchema#dayTimeDuration"
        MustBePresent="false"/>
    </Apply>
  </Apply>
</Apply>
```

5 Date Functions

This section defines functions for performing arithmetic on date values, in particular to allow adding a number of days to, or subtracting a number of days from, a date. The functions are defined by reference to XML Schema [**XSD2**]. Implementations are free to use any method that produces the same results.

5.1 The date-add-dayTimeDuration Function

The `urn:oasis:names:tc:xacml:3.0:function:date-add-dayTimeDuration` function adds a duration to a date value.

This function SHALL take two arguments, the first SHALL be of data-type `http://www.w3.org/2001/XMLSchema#date` and the second SHALL be of data-type `http://www.w3.org/2001/XMLSchema#dayTimeDuration`. It SHALL return a result of data-type `http://www.w3.org/2001/XMLSchema#date`. The second argument MAY be a negative duration.

This function SHALL return the value resulting from adding the second argument to the first argument according to the specification for adding durations to date values ([**XSD2**] Appendix E). Note that the time components are discarded from the result of the addition.

5.2 The date-subtract-dayTimeDuration Function

The `urn:oasis:names:tc:xacml:3.0:function:date-subtract-dayTimeDuration` function subtracts a duration from a date value.

This function SHALL take two arguments, the first SHALL be of data-type `http://www.w3.org/2001/XMLSchema#date` and the second SHALL be of data-type `http://www.w3.org/2001/XMLSchema#dayTimeDuration`. It SHALL return a result of data-type `http://www.w3.org/2001/XMLSchema#date`. The second argument MAY be a negative duration.

If the second argument is a positive duration, then this function SHALL return the value resulting from adding the corresponding negative duration to the date value according to the specification for adding durations to date values ([**XSD2**] Appendix E). Otherwise (the second argument is a negative duration), the function SHALL return the value resulting from adding the corresponding positive duration to the date value.

6 The dayOfWeek Data-type

The ***dayOfWeek*** data-type is used to represent one of the days of the week as a number from 1 to 7 with an optional time zone. It is identified by the URI `urn:oasis:names:tc:xacml:3.0:data-type:dayOfWeek`.

The lexical representation for a value of this data-type is defined by the `dayOfWeek` rule in the following ABNF [**RFC5234**]:

```
dayOfWeek = day [ timeZone ]
day       = ( "1" / "2" / "3" / "4" / "5" / "6" / "7" )
timeZone  = ( "+" | "-" ) hours ":" minutes
          / ( "+" | "-" ) "14:00"
          / "Z"

hours   = "0" digit
        / "1" ( "0" / "1" / "2" / "3" )
minutes = ( "0" / "1" / "2" / "3" / "4" / "5" ) digit

digit = "0" / "1" / "2" / "3" / "4" / "5" / "6" / "7" / "8" / "9"
```

The days of the week are numbered in order where Monday is represented by the number 1 and Sunday is represented by the number 7 (this assignment has been chosen for consistency with ISO 8601 [**ISO8601**]).

In the XML representation [**XML**] of a ***dayOfWeek*** value, the sequence of character information items in the [children] [**INFOSET**] of an `<AttributeValue>` element [**XACML3**], after the removal of any leading and/or trailing XML whitespace, MUST conform to the lexical representation.

A ***dayOfWeek*** value is represented in JSON [**XACMLJSON**] as the lexical representation in a JSON string. The JSON shorthand type code for the ***dayOfWeek*** data-type is "`dayOfWeek`". This data-type MUST always be explicitly given in the JSON representation; it cannot be inferred from an attribute value.

6.1 Examples of dayOfWeek Values

{Non-normative}

Tuesday in Australian Eastern Standard Time:

```
XML:
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
      >2+10:00</AttributeValue>
JSON:
    "2+10:00"
```

Friday in Pacific Daylight Time:

```
XML:
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time">
      5-07:00
    </AttributeValue>
JSON:
    "5-07:00"
```

Wednesday in local time:

```
XML:
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#time"
      >3</AttributeValue>
```

```
JSON:
    "3"
```

7 Day of the Week Functions

7.1 The dayOfWeek-from-string Function

The `urn:oasis:names:tc:xacml:3.0:function:dayOfWeek-from-string` function SHALL take one argument of data-type `http://www.w3.org/2001/XMLSchema#string`. If the argument is a valid lexical representation of a ***dayOfWeek*** value after the removal of any leading and/or trailing XML whitespace, then the result SHALL be the corresponding value of the ***dayOfWeek*** data-type; otherwise, the result SHALL be “Indeterminate” with status code `urn:oasis:names:tc:xacml:1.0:status:syntax-error`.

7.2 The string-from-dayOfWeek Function

The `urn:oasis:names:tc:xacml:3.0:function:string-from-dayOfWeek` function SHALL take one argument of the ***dayOfWeek*** data-type and return a value of data-type `http://www.w3.org/2001/XMLSchema#string`. The returned value SHALL be the lexical representation of the argument (leading or trailing whitespace is stripped).

7.3 The dayOfWeek-one-and-only Function

The `urn:oasis:names:tc:xacml:3.0:function:dayOfWeek-one-and-only` function SHALL take a bag of values of the ***dayOfWeek*** data-type as its only argument. If the bag contains exactly one value, then the function returns that value; otherwise, the function evaluates to “Indeterminate”.

7.4 The dayOfWeek-bag-size Function

The `urn:oasis:names:tc:xacml:3.0:function:dayOfWeek-bag-size` function SHALL take a bag of values of the ***dayOfWeek*** data-type as its only argument and SHALL return an `http://www.w3.org/2001/XMLSchema#integer` value indicating the number of values in the bag.

7.5 The dayOfWeek-bag Function

The `urn:oasis:names:tc:xacml:3.0:function:dayOfWeek-bag` function SHALL take any number of arguments of the ***dayOfWeek*** data-type and return a bag containing the values of those arguments. An application of this function to zero arguments SHALL produce an empty bag of the ***dayOfWeek*** data-type.

7.6 The dateTime-in-dayOfWeek-range Function

The `urn:oasis:names:tc:xacml:3.0:function:dateTime-in-dayOfWeek-range` function tests whether a dateTime value is within a range of days of the week given by two ***dayOfWeek*** values.

This function SHALL take three arguments and SHALL return an `http://www.w3.org/2001/XMLSchema#boolean`. The data-type of the first argument SHALL be `http://www.w3.org/2001/XMLSchema#dateTime`. The data-type of the second and third arguments SHALL be `urn:oasis:names:tc:xacml:3.0:data-type:dayOfWeek`.

If no time zone is provided for the first argument, it SHALL use the default time zone at the ***context handler***. If no time zone is provided for the second argument, then it SHALL use the same time zone as the first argument. If no time zone is provided for the third argument, then it SHALL use the same time zone as the first argument.

The second argument is converted to a dateTime value by following these steps:

1. Create a new dateTime value where:
    a. the date components, i.e., year, month and day, are set to the values of the same components in the ***reference date***,

   b. the time components, i.e., hour, minute and second, are set to zero,
   c. fractional seconds are absent, and
   d. the time zone is set to the time zone of the second argument.

   For example, given the dayOfWeek value `2+10:00`, the dateTime value becomes `2017-01-15T00:00:00+10:00`.

2. Create a new dayTimeDuration value where the day component has the same value as the day component of the second argument and all other components are zero or absent. For example, the dayOfWeek value `2+10:00` becomes `P2D`.
3. Add the dayTimeDuration value from step 2 to the dateTime value from step 1 according to the specification for adding durations to dateTime values, [**XSD2**] Appendix E, to obtain the final converted value (e.g., `2017-01-17T00:00:00+10:00`).

The second argument converted to a dateTime value defines a series of inclusive dateTime start points that recur every seven days backwards and forwards in time.

The third argument is converted to a dateTime value by following these steps:

1. Create a new dateTime value where:
   a. the date components, i.e., year, month and day, are set to the values of the same components in the ***reference date***,
   b. the time components, i.e., hour, minute and second, are set to zero,
   c. fractional seconds are absent, and
   d. the time zone is set to the time zone of the third argument.
2. Create a new dayTimeDuration value where the day component has the value *one more than* the day component of the third argument and all other components are zero or absent. For example, the ***dayOfWeek*** value `4+10:00` becomes `P5D`.
3. Add the dayTimeDuration value from step 2 to the dateTime value from step 1 according to the specification for adding durations to dateTime values, [**XSD2**] Appendix E, to obtain the final converted value.

The third argument converted to a dateTime value defines a series of exclusive dateTime end points that recur every seven days backwards and forwards in time.

If any argument evaluates to “Indeterminate”, then the function evaluates to “Indeterminate”; otherwise, the function returns “True” if the first argument is greater than or equal to one of the start points and less than the end point that is greater than that start point by no more than 7 days (i.e., the closest end point greater than the start point); otherwise, the function returns “False”. The dateTime values are compared according to the algorithm defined in [**XSD2**], section 3.2.7.4.

7.6.1 Example 1

{Non-normative}

Note that the algorithm for comparing dateTime values normalizes its arguments to UTC before comparing fields. This example and the following example normalize the converted arguments to the dateTime-in-dayOfWeek-range function earlier for clarity.

Consider the following XACML expression for testing whether the current time is on a Tuesday, Wednesday or Thursday in Australian Eastern Standard Time.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId=
    "urn:oasis:names:tc:xacml:3.0:function:dateTime-in-dayOfWeek-range">
  <Apply FunctionId=
           "urn:oasis:names:tc:xacml:1.0:function:dateTime-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-dateTime"
      DataType="http://www.w3.org/2001/XMLSchema#dateTime"
      MustBePresent="false"/>
  </Apply>
  <AttributeValue DataType="urn:oasis:names:tc:xacml:3.0:data-type:dayOfWeek"
    >2+10:00</AttributeValue>
  <AttributeValue DataType="urn:oasis:names:tc:xacml:3.0:data-type:dayOfWeek"
    >4+10:00</AttributeValue>
</Apply>
```

The start point of the range is `2+10:00`, which converts to `2017-01-17T00:00:00+10:00` (i.e., `P2D` added to `2017-01-15T00:00:00+10:00`), which is equivalent to `2017-01-16T14:00:00Z`. This determines a sequence of inclusive weekly start points before and after the ***reference date***, e.g., `2017-01-09T14:00:00Z`, `2017-01-16T14:00:00Z`, `2017-01-23T14:00:00Z`, …, `2017-06-05T14:00:00Z`, `2017-06-12T14:00:00Z` and `2017-06-19T14:00:00Z`.

The end point of the range is `4+10:00`, which converts to `2017-01-20T00:00:00+10:00` (i.e., `P5D` added to `2017-01-15T00:00:00+10:00`) or `2017-01-19T14:00:00Z`. This determines a sequence of exclusive weekly end points before and after the ***reference date***, e.g., `2017-01-12T14:00:00Z`, `2017-01-19T14:00:00Z`, `2017-01-26T14:00:00Z`, …, `2017-06-08T14:00:00Z`, `2017-06-15T14:00:00Z` and `2017-06-22T14:00:00Z`.

Suppose that the current dateTime generated by the ***context handler*** is `2017-06-13T09:00:00+10:00` (9:00am Tuesday), which normalizes to the dateTime value `2017-06-12T23:00:00Z`. In this case the dateTime-in-dayOfWeek-range function returns “True” because `2017-06-12T23:00:00Z` is greater than or equal to the start point `2017-06-12T14:00:00Z` and less than the next greater end point of `2017-06-15T14:00:00Z`. The dateTime value `2017-06-12T16:00:00-07:00` (4:00pm Monday) also normalizes to `2017-06-12T23:00:00Z`, so the function would return “True” if the ***context handler*** had generated this value instead. Although it is still Monday in Pacific Daylight Time it is already Tuesday in Australian Eastern Standard Time.

7.6.2 Example 2

Consider the following XACML expression for testing whether the current time is on a Friday, Saturday, Sunday or Monday in Pacific Daylight Time.

```
<Apply xmlns="urn:oasis:names:tc:xacml:3.0:core:schema:wd-17"
  FunctionId=
    "urn:oasis:names:tc:xacml:3.0:function:dateTime-in-dayOfWeek-range">
  <Apply FunctionId=
           "urn:oasis:names:tc:xacml:1.0:function:dateTime-one-and-only">
    <AttributeDesignator
      Category="urn:oasis:names:tc:xacml:3.0:attribute-category:environment"
      AttributeId="urn:oasis:names:tc:xacml:1.0:environment:current-dateTime"
      DataType="http://www.w3.org/2001/XMLSchema#dateTime"
      MustBePresent="false"/>
  </Apply>
  <AttributeValue DataType="urn:oasis:names:tc:xacml:3.0:data-type:dayOfWeek"
    >5-07:00</AttributeValue>
  <AttributeValue DataType="urn:oasis:names:tc:xacml:3.0:data-type:dayOfWeek"
    >1-07:00</AttributeValue>
</Apply>
```

The start point of the range is `5-07:00`, which converts to `2017-01-20T00:00:00-07:00` (i.e., `P5D` added to `2017-01-15T00:00:00-07:00`), which is equivalent to `2017-01-20T07:00:00Z`. This determines a sequence of inclusive weekly start points before and after the ***reference date***, e.g., `2017-01-13T07:00:00Z`, `2017-01-20T07:00:00Z`, `2017-01-27T07:00:00Z`, …, `2017-06-02T07:00:00Z`, `2017-06-09T07:00:00Z` and `2017-06-16T07:00:00Z`.

The end point of the range is `1-07:00`, which converts to `2017-01-17T00:00:00-07:00` (i.e., `P2D` added to `2017-01-15T00:00:00-07:00`) or `2017-01-17T07:00:00Z`. This determines a sequence of exclusive weekly end points before and after the ***reference date***, e.g., `2017-01-10T07:00:00Z`, `2017-01-17T07:00:00Z`, `2017-01-24T07:00:00Z`, …, `2017-06-06T07:00:00Z`, `2017-06-13T07:00:00Z` and `2017-06-20T07:00:00Z`.

Suppose that the current dateTime generated by the ***context handler*** is `2017-06-12T09:00:00-07:00` (9:00am Monday), which maps to the dateTime value `2017-06-12T16:00:00Z`. In this case the dateTime-in-dayOfWeek-range function returns “True” because `2017-06-12T16:00:00Z` is greater than or equal to the start point `2017-06-09T07:00:00Z` (the start of Friday in PDT) and less than the next greater end point of `2017-06-13T07:00:00Z` (the end of Monday in PDT).

7.6.3 Implementation

{Non-normative}

Implementations of the dateTime-in-dayOfWeek-range function are free to use any method that produces the same results. Here is one way to evaluate the function:

1. If any argument evaluates to “Indeterminate”, then return “Indeterminate”.
2. Normalize the first argument to UTC.
3. Determine the day of the week of the normalized first argument, e.g., using the Doomsday rule [**DOOMSDAY**], then choose the date of the preceding Sunday as the ***reference date***. Even if the normalized first argument happens to fall on a Sunday still choose the preceding Sunday. Definition: Let the reference point be the time `00:00:00Z` on the chosen ***reference date***.
4. Convert the second argument to a dateTime start point according to Section 7.6 using the chosen ***reference date*** and normalize to UTC. If the difference between the start point and the reference point is less than one day, then add seven days to the start point.
5. Convert the third argument to a dateTime end point according to Section 7.6 using the chosen ***reference date*** and normalize to UTC. If the difference between the end point and the reference point is greater than eight days, then subtract seven days from the end point.
6. If the end point is greater than the start point and the first argument is greater than or equal to the start point *and* less than the end point, then return “True”.
7. If the end point is less than or equal to the start point and the first argument is less than the end point *or* greater than or equal to the start point, then return “True”.
8. Otherwise, return “False”.

8 Security Considerations

{Non-normative}

The data-type and functions defined in this document are subject to the same security concerns as other XACML data-types and functions. The reader should refer to the security and privacy considerations in the XACML core specification [**XACML3**].

The time-zone attribute should be protected from unauthorized discovery since its value may assist in determining the current location of a user. The Security Considerations of the Related and Nested Entities Profile [**ENTITIES**] discusses various ways in which the value of an attribute can be discovered by unauthorized users and possible ways to prevent that discovery.

9 Conformance

9.1 Conformance Targets

This document defines conformance for a ***PDP*** implementation and its associated ***context handler*** implementation, and for ***PAP***, ***PEP*** and ***PIP*** implementations.

9.2 Conformance Clause 1: “Evaluation”

A ***PDP*** implementation and its associated ***context handler*** implementation satisfy “Evaluation” conformance profile if they are able to evaluate:

- attribute designators, attribute selectors and attribute values that use the ***dayOfWeek*** data-type defined in Section 6, and
- for every function defined in Section 3, Section 5 and Section 7, apply and function expressions using the function.

9.3 Conformance Clause 2: “Composition”

A ***PAP*** implementation satisfies “Composition” conformance profile if it supports the creation of:

- attribute designators, attribute selectors and attribute values that use the ***dayOfWeek*** data-type defined in Section 6,
- attribute designators that use the attribute defined in Section 4, and
- for every function defined in Section 3, Section 5 and Section 7, apply and function expressions using the function.

9.4 Conformance Clause 2: “Request”

A ***PEP*** implementation satisfies “Request” conformance profile if it is able to generate authorization requests containing:

- attribute values that use the ***dayOfWeek*** data-type defined in Section 6, and
- the attribute defined in Section 4.

9.5 Conformance Clause 2: “Fetch”

A ***PIP*** implementation satisfies “Fetch” conformance profile if it is able to satisfy requests for attributes with values of the ***dayOfWeek*** data-type defined in Section 6 and the `http://www.w3.org/2001/XMLSchema#dayTimeDuration` data-type [**XACML3**].

Appendix A. Acknowledgments

Voting members of the XACML Technical Committee:
Mohammad Jafari, Veterans Health Administration
Steven Legg, ViewDS Identity Solutions
Rich Levinson, Oracle
Hal Lochhart, Individual Member
Bill Parducci, Individual Member

Appendix B. Revision History

Revision	Date	Editor	Changes Made
WD 01	2 August 2019	Steven Legg	Initial draft as a TC work product. Changes have been made with respect to the previous informal proposal. The subject:time-zone and resource:time-zone attributes have been merged into a common entity:time-zone attribute. Added string-from-dayOfWeek and dayOfWeek-from-string functions. Added conformance clauses and security considerations.
WD 02	1 November 2019	Steven Legg	Clarified for the time-in-recurring-range and dateTime-in-dayOfWeek-range functions that ***only*** the arguments specified in local time have their time zone revised. Added a non-normative clarification about disposal of the time components to the description of the date-add-dayTimeDuration function.