TOVE Digital City Programming Manual  
Part IIa: Temporal Representation and Reasoning

Mark S. Fox, [msf@eil.utoronto.ca](mailto:msf@eil.utoronto.ca)

20 March 2021

# Introduction

This report defines the standard temporal representation and reasoning features of the TOVE Digital City (TDC).

In the remainder of this report, we use the following ontology prefix’s:

|  |  |
| --- | --- |
| **Prefix** | **IRI** |
| ctime | http://ontology.eil.utoronto.ca/tove/ctime# |
| time | http://www.w3.org/2006/time# |

# OWL-Time

TDC adopts the OWL-Time ontology (Hobbs & Pan, 2006; Cox & Little, 2020) as its temporal representation with extensions defined in Section 4. Two important classes that are used directly are: time:Instant, time:DateTimeDescription and time:DurationDescription defined as follows in Table 1. Note that we extend the definitions by further restricting the range of values a property can have, and that some are functional:

Table 1: Extended OWL-TIME Relevant Instant and Duration Classes

|  |  |  |
| --- | --- | --- |
| **Class** | **Property** | **Value Restriction** |
| Instant | rdfs:subClassOf | time:TemporalEntity |
| time:before | only time:TemporalEntity |
| time:after | only time:TemporalEntity |
| time:inDateTime (functional) | exactly 1 time:DateTimeDescription |
| time:inXSDDateTimeStamp | exactly 1 xsd:dateTimeStamp |
| DateTimeDescription | rdfs:subClassOf | time:GeneralDateTimeDescription |
| year (functional) | exactly 1 xsd:positiveinteger |
| month (functional) | exactly 1 xsd:positiveinteger[<= 12] |
| day (functional) | exactly 1 xsd:positiveinteger[<= 31] |
| hour (functional) | exactly 1 xsd:positiveinteger[<= 24] |
| minute (functional) | exactly 1 xsd:positiveinteger[<= 60] |
| second (functional) | exactly 1 xsd:positiveinteger[<= 60] |
| unitType (functional) | exactly 1 time:TemporalUnit |
| DurationDescription | rdfs:subClassOf | time:GeneralDurationDescription |
| time:years (functional) | exactly 1 xsd:positiveInteger |
| time:months (functional) | exactly 1 xsd:positiveInteger |
| time:days (functional) | exactly 1 xsd:positiveInteger |
| time:hours (functional) | exactly 1 xsd:positiveInteger |
| time:minutes (functional) | exactly 1 xsd:positiveInteger |
| time:seconds (functional) | exactly 1 xsd:positiveInteger |

The time of a time:Instant is specified by the time:inDateTime property that links to an instance of time:DateTimeDescription. Note that an Instant also contains the data property time:inXSDDateTimeStamp as an alternative method of specified the time of the Instant, but is not used by TDC.

TDC includes OWL-Time temporal relations. Figure 3 depicts the interpretation of the temporal relations between pairs of intervals.

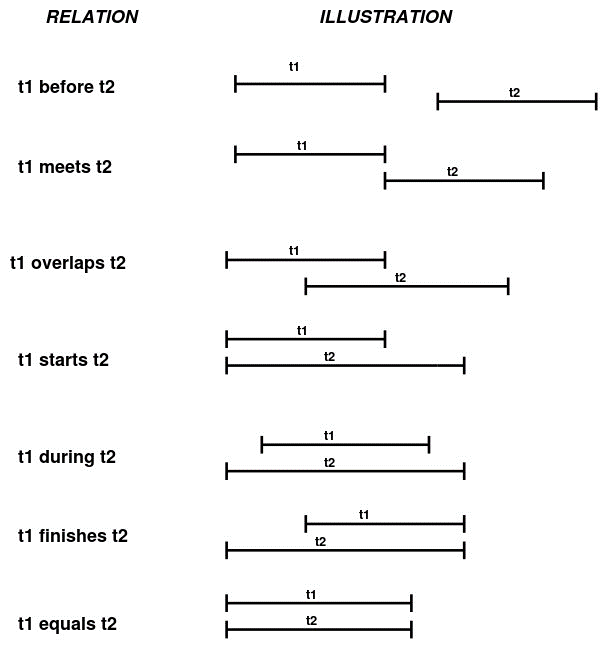


Figure 3: TEMPORAL RELATIONS

The following object properties that represent temporal relations are included.

Table 2: Extended OWL-Time Temporal Relations

|  |  |  |
| --- | --- | --- |
| **Property** | **Property** | **Value Restriction** |
| time:before | Domain | time:TemporalEntity |
| Range | time:TemporalEntity |
| Inverse | time:after |
| Transitive |  |
| time:after | Domain | time:TemporalEntity |
| Range | time:TemporalEntity |
| Inverse | time:before |
| Transitive |  |
| time:intervalBefore | subPropertyOf | time:before |
| Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalAfter |
| Transitive |  |
| time:intervalAfter | subPropertyOf | time:after |
| Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalBefore |
| Transitive |  |
| time:intervalOverlaps | Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalOverlappedBy |
| time:intervalOverlappedBy | Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalOverlaps |
| time:intervalDuring | subPropertyOf | time:intervalIn |
| Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalContains |
| Transitive |  |
| time:intervalContains | Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalDuring |
| Transitive |  |
| time:intervalMeets | Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalMetby |
| time:intervalMetBy | Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalMeets |
| time:intervalStarts | subPropertyOf | time:intervalIn |
| Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalStartedBy |
| Transitive |  |
| time:intervalStartedBy | Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalStarts |
| time:intervalFinishes | subPropertyOf | time:intervalIn |
| Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalFinishedBy |
| time:intervalFinishedBy | subPropertyOf | time:intervalIn |
| Domain | time:properInterval |
| Range | time:properInterval |
| Inverse | time:intervalFinishes |
| time:intervalEquals | subPropertyOf | time:intervalIn |
| Domain | time:properInterval |
| Range | time:properInterval |
| Transitive |  |

# OWL-Time Functions (http://ontology.eil.utoronto.ca/dt/code/otime.py)

This section lists the functions available for reasoning with the constrained time pattern. The functions are implemented in OWLReady2 (Lamy, 2017) module in Python.

|  |  |
| --- | --- |
| **createDTD(dts, ns=None)** | |
| *Creates a DateTimeDescription given a datetime in a string format.* | |
| dts | datetime in string format, e.g., "2021-2-10T21:6:0", “February 10, 2021 9:06PM” |
| **ns** | namespace the instance of time:DateTimeDescription is to be created. |
| **Returns** | new time:DateTimeDescription instance. |
|  | |
| **printDTD(dtd, prnt=True)** | |
| *Prints the DateTimeDescription in the following format: YYYY-MM-DDThh:mm:ss. If prnt is False, it does not print but returns the datetime as a string.* | |
| dtd | time:DateTimeDescription instance |
| prnt | Boolean, if true then the datetime string is printed |
| Returns | *YYYY-MM-DDThh:mm:ss* |
|  | |
| **copyDTD(dtd, ns=None)** | |
| *Copies dtd into a new instance of time:DateTimeDescription and returns the new instance.* | |
| **dtd** | time:DateTimeDescription instance to be copied |
| **ns** | namespace the new instance of time:DateTimeDescription is to be created. |
| **Returns** | new time:DateTimeDescription instance. |
|  | |
| **copyDTD2DTD(dtd1, dtd2)** | |
| *Copies the datetime information from dtd1 into dtd2* | |
| **dtd1** | time:DateTimeDescription instance to be copied |
| **dtd2** | Target time:DateTimeDescription instance to be copied into |
| **Returns** | None |
|  | |
| **addDTD(dtd, dur, ns=None)** | |
| *Adds the time:DateTimeDuration instance to the time:DurationDescription instance and returns a new time:DateTimeDescritpion instance containing the result.* | |
| **dtd** | time:DateTimeDescription instance |
| **dur** | time:DurationDescription instance |
| **Returns** | new time:DateTimeDescription instance containing the result of the addition |
|  | |
| **subtractDTD(dtd, dur, ns=None)** | |
| *Subtracts the time:DateTimeDuration instance from the time:DurationDescription instance and returns a new time:DateTimeDescription instance containing the result.* | |
| **dtd** | time:DateTimeDescription instance |
| **dur** | time:DurationDescription instance |
| **Returns** | new time:DateTimeDescription instance containing the result of the subtraction |
|  | |
| **lessThan(dtd1, dtd2)** | |
| *Determines if the first time:DateTimeDescription instance is less than the second* | |
| **dtd1** | time:DateTimeDescription instance |
| **dtd2** | time:DateTimeDescription instance |
| **Returns** | Boolean, True if dtd1 is less than dtd2 |
|  | |
| **greaterThan(dtd1, dtd2)** | |
| *Determines if the first time:DateTimeDescription instance is greater than the second* | |
| **dtd1** | time:DateTimeDescription instance |
| **dtd2** | time:DateTimeDescription instance |
| **Returns** | Boolean, True if dtd1 is greater than dtd2 |
|  | |
| **intervalBefore(int1, int2)** | |
| *Determines whether the first interval is temporally before the second interval.* | |
| **int1** | time:ProperInterval instance with time:inDateTime linking to DateTimeDescription |
| **int2** | time:ProperInterval instance |
| **Returns** | Boolean, true if int1 < int2 |
|  | |
| **intervalOverlaps(int1, int2)** | |
| *Determines whether the first interval is temporally overlaps the second interval.* | |
| **int1** | time:ProperInterval instance with time:inDateTime linking to DateTimeDescription |
| **int2** | time:ProperInterval instance |
| **Returns** | Boolean, true if int1.hasEnd < int2.hasEnd and int1.hasEnd > int2.hasBeginning |
|  | |
| **intervalDuring(int1, int2)** | |
| *Determines whether the first interval is temporally during the second interval.* | |
| **int1** | time:ProperInterval instance with time:inDateTime linking to DateTimeDescription |
| **int2** | time:ProperInterval instance |
| **Returns** | Boolean, true if int1.beginning > int2.hasBeginning and int1.hasEnd < int2.hasEnd |
|  | |
| **intervalContains(int1, int2)** | |
| *Determines whether the first interval is temporally before the second interval.* | |
| **int1** | time:ProperInterval instance with time:inDateTime linking to DateTimeDescription |
| **int2** | time:ProperInterval instance |
| **Returns** | Boolean, true if int1.beginning < int2.hasBeginning and int1.hasEnd > int2.hasEnd |
|  | |
| **intervalAfter(int1, int2)** | |
| *Determines whether the first interval is temporally after the second interval.* | |
| **int1** | time:ProperInterval instance with time:inDateTime linking to DateTimeDescription |
| **int2** | time:ProperInterval instance |
| **Returns** | Boolean, true if int1.hasBeginning < int2.hasEnd |

# References

Cox, S., and Little, C. (2020), “Time Ontology in OWL”, OGC Document Number OGC 16-071r3, <https://www.w3.org/TR/owl-time/>. Downloaded 3 February 2021.

Hobbs, J. R., & Pan, F. (2006). Time ontology in OWL. *W3C working draft*, *27*, 133.

Lamy JB. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. Artificial Intelligence In Medicine 2017;80:11-28