OSID V3 Specifications
calendaring package

Version Draft 3

This specifications represent a draft for OSID V3 interface definitions. These definitions may change at any time.

Last Modified: 12 October 2008

prepared by:
Tom Coppeto
OnTapSolutions

	OSID License
Copyright	Copyright © 2002-2008 Massachusetts Institute of Technology. All Rights Reserved.
License	This Work is being provided by the copyright holder(s) subject to the following license. By obtaining, using and/or copying this Work, you agree that you have read, understand, and will comply with the following terms and conditions. This Work and the information contained herein is provided on an "AS IS" basis. The Massachusetts Institute of Technology, the Open Knowledge Initiative, and THE AUTHORS DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE WORK OR THE USE OR OTHER DEALINGS IN THE WORK. Permission to use, copy and distribute unmodified versions of this Work, for any purpose, without fee or royalty is hereby granted, provided that you include the above copyright notice and the terms of this license on ALL copies of the Work or portions thereof. You may nodify or create Derivatives of this Work only for your internal purposes. You shall not distribute or transfer any such Derivative of this Work to any location or to any third party. For the purposes of this license, Derivative shall mean any derivative of the Work as defined in the United States Copyright Act of 1976, such as a translation or modification. The export of software employing encryption technology may require a specific license from the United States Government. It is the responsibility of any person or organization comtemplating export to obtain such a license before exporting this Work.

Package Description	osid.id package

Interfaces	osid.id.IdProfile osid.DateTime osid.DateTimeResolution osid.DateTimeList osid.DateTimeInterval osid.DateTimeIntervalList osid.id.IdList

Package	**osid.calendaring**
Title	**Calendaring Open Service Interface Definitions**
Version	3.0.0
Description	

Interface	**osid.calendaring.DateTime**
Implements	

Description	The DateTime interface defines a date and/or time. This interface provides a very broad range of dates, describes more or less precision, and/or conveys an uncertainty. A number of convenience methods for retrieving time elements are available but only those methods covered by the specified granularity are valid.
	
	A typical example is describing a day where the time isn't known, and the event did not occur at midnight.
	
	`getMillenium() == 2`
	`getCentury() == 18`
	`getYear() == 1776`
	`getMonth() == 7`
	`getDay() == 4`
	`getHour() == 0`
	`getGranularity() == DateTimeResolution.DAY`
	`definesUncertainty() == false`
	Another example showing that the time is probably 1pm but could have been as late as 3pm or early as noon..
	
	`getMillenium() == 3`
	`getCentury() == 21`
	`getYear() == 2008`
	`getMonth() == 3`
	`getDay() == 17`
	`getHour() == 13`
	`getMinute() == 0`
	`getGranularity() == TimeResolution.MINUTE`
	`definesUncertainty() == true`
	`getUncertaintyGranularity() == DateTimeResolution.HOUR`
	`getUncertaintyMinus() == 1`
	`getUncertaintyPlus == 2`
	
	An example marking the birth of the universe. 13.73 billion years +/- 120 million years. The granularity suggests that no more resolution than one million years can be inferred from the "clock", making errors in the exact number of millennia insignificant.
	
	`getEpoch() == -13,730`
	`getMillenium() == 0`
	`getCentury() == 0`
	`getYear() == 0`
	`getGranularity() == TimeResolution.EPOCH`
	`definesUncertainty() == true`
	`getUncertaintyGranularity() == DateTimeResolution.EPOCH`
	`getUncertaintyMinus() == 120`
	`getUncertaintyPlus == 120`

Method	getAeon	
Description	Gets the aeon starting from 1. 1B years.	
Return	integer	the aeon
Compliance	mandatory	This method must be implemented.
Method	**getEpoch**	
Description	Gets the epoch starting from 1.An epoch is 1M years.	
Return	integer	the millenium
Compliance	mandatory	This method must be implemented.
Method	**getMillenium**	
Description	Gets the millenium starting from 1. A millenium is 1,000 years.	
Return	integer	the millenium
Compliance	mandatory	This method must be implemented.
Method	**getCentury**	
Description	Gets the century starting from 1.	
Return	integer	the century
Compliance	mandatory	This method must be implemented.
Method	**getYear**	
Description	Gets the year starting from 1.	
Return	integer	the year
Compliance	mandatory	This method must be implemented.
Method	**getMonth**	
Description	Gets the month number starting from 1.	
Return	cardinal	the month
Compliance	mandatory	This method must be implemented.
Method	**getDay**	
Description	Gets the day of the month starting from 1.	
Return	cardinal	the day of the month
Compliance	mandatory	This method must be implemented.
Method	**getHour**	
Description	Gets the hour of the day 0-23.	
Return	cardinal	the hour of the day
Compliance	mandatory	This method must be implemented.
Method	**getMinute**	
Description	Gets the minute of the hour 0-59.	
Return	cardinal	the minute of the hour
Compliance	mandatory	This method must be implemented.
Method	**getSecond**	
Description	Gets the second of the minute 0-59.	
Return	cardinal	the second of the minute
Compliance	mandatory	This method must be implemented.
Method	**getMilliseconds**	
Description	Gets the number of milliseconds in this second 0-999. A millisecond is one thousandth of a second.	
Return	cardinal	the milliseconds of the second
Compliance	mandatory	This method must be implemented.
Method	**getMicroseconds**	
Description	Gets the number of microseconds of the second 0-999. A microsecond is one millionth of a second.	
Return	cardinal	the micrseconds of the millisecond
Compliance	mandatory	This method must be implemented.

Method	getNanoseconds	
Description	Gets the number of nanoseconds of the microsecond 0-999. A nanosecond is one billionth of a second.	
Return	cardinal	the nanoseconds of the microsecond
Compliance	mandatory	This method must be implemented.
Method	getPicoseconds	
Description	Gets the number of picoseconds of the nanosecond 0-999. A picosecond is one trillionth of a second.	
Return	cardinal	the picoseconds of the nanosecond
Compliance	mandatory	This method must be implemented.
Method	getFemtoseconds	
Description	Gets the number of femtoseconds of the picosecond 0-999. A femtosecond is one quadrillionth of a second.	
Return	cardinal	the femtoseconds of the picosecond
Compliance	mandatory	This method must be implemented.
Method	getAttoseconds	
Description	Gets the number of attoseconds of the femtoseconds 0-999. An attosecond is one quintillionth of a second.	
Return	cardinal	the attoseconds of the femtosecond
Compliance	mandatory	This method must be implemented.
Method	getZeptoseconds	
Description	Gets the number of zeptoseconds of the attosecond 0-999. A zeptosecond is one sextillionth of a second.	
Return	cardinal	the zeptoseconds of the attosecond
Compliance	mandatory	This method must be implemented.
Method	getYoctoseconds	
Description	Gets the number of yoctoseconds of the picosecond 0-999. A yoctosecond is one septillionth of a second. This is getting quite small.	
Return	cardinal	the yoctoseconds of the yoctosecond
Compliance	mandatory	This method must be implemented.
Method	getXoxxoseconds	
Description	Gets the number of xoxxoseconds of the yoctosecond 0-999. A xoxxosecond is one octillionth of a second. We're going with Rudy Rucker here.	
Return	cardinal	the xoxxoseconds of the yoctosecond
Compliance	mandatory	This method must be implemented.
Method	getWeebleseconds	
Description	Gets the number of weebleseconds of the xoxxosecond 0-999. A weeblesecond is one nonillionth of a second.	
Return	cardinal	the weebleseconds of the xoxxoseconds
Compliance	mandatory	This method must be implemented.
Method	getVatoseconds	
Description	Gets the number of vatoseconds of the xoxxosecond 0-999. A vatosecond is one decillionth of a second.	
Return	cardinal	the vatoseconds of the weeblesecond
Compliance	mandatory	This method must be implemented.
Method	getUndaseconds	
Description	Gets the number of undaseconds of the vatosecond 0-999. An undasecond is one unadecillionth of a second.	
Return	cardinal	the undaseconds of the vatosecond
Compliance	mandatory	This method must be implemented.
Method	getPlanckSeconds	
Description	Gets the number of Plancks of the vatoseconds. A Planck is 10 quattuordecillionths of a second.	
Return	float	the plancks of the undasecond
Compliance	mandatory	This method must be implemented.

Method	getGranularity	
Description	Gets the granularity of this time. The granularity indicates the resolution of the clock. More precision that what is specified in this method cannot be inferred from the available data.	
Return	osid.calendaring.DateTimeResolution	granularity
Compliance	mandatory	This method must be implemented.

Method	definesUncertainty	
Description	Tests if uncertainty is defined for this time.	
Return	boolean	true if uncertainty is defined, false otherwise
Compliance	mandatory	This method must be implemented.

Method	getUncertaintyUnits	
Description	Gets the units of the uncertainty.	
Return	osid.calendaring.DateTimeResolution	units of the uncertainty
Errors	ILLEGAL_STATE	definesUncertainty() is false
Compliance	mandatory	This method must be implemented.

Method	getUncertaintyMinus	
Description	Gets the uncertainty of this time in the negative direction.	
Return	cardinal	the uncertainty under this value
Errors	ILLEGAL_STATE	definesUncertainty() is false
Compliance	mandatory	This method must be implemented.

Method	getUncertaintyPlus	
Description	Gets the uncertainty of this time in the positive direction.	
Return	cardinal	the uncertainty over this value
Errors	ILLEGAL_STATE	definesUncertainty() is false
Compliance	mandatory	This method must be implemented.

Method	isGreater	
Description	Tests if the given time is greater than this one. A time is greater if its time inclusive of granularity minus uncertainty is greater than the other inclusive of its granularity plus its uncertainty.	
Parameters	osid.calendaring.DateTime time	the time to compare
Return	boolean	true if the given time is greater than this one, false otherwise
Errors	NULL_ARGUMENT	time is null
Compliance	mandatory	This method must be implemented.

Method	isLess	
Description	Tests if the given time is less than this one. A time is greater if its time inclusive of granularity plus uncertainty is less than the other inclusive of its granularity minus its uncertainty.	
Parameters	osid.calendaring.DateTime time	the time to compare
Return	boolean	true if the given time is less than this one, false otherwise
Errors	NULL_ARGUMENT	time is null
Compliance	mandatory	This method must be implemented.

Method	isInclusive	
Description	Tests if the given time is included in this one. A time is inclusive of this time and granularity plus and minus its uncertainty is completely contained within this one inclusive of its granulariity and uncertainty.	
Parameters	osid.calendaring.DateTime time	the time to compare
Return	boolean	true if the given time is included in this one, false otherwise
Errors	NULL_ARGUMENT	time is null
Compliance	mandatory	This method must be implemented.

Method	isExclusive	
Description	Tests if the given time is exclusive in this one. A time is exclsuive of this time if there is no overlap taking into account granularity and uncertainty.	
Parameters	osid.calendaring.DateTime time	the time to compare
Return	boolean	true if the given time is exclsuive of this one, false otherwise
Errors	NULL_ARGUMENT	time is null
Compliance	mandatory	This method must be implemented.

Method	isEqual	
Description	Tests if the given time is equal to this one. A time is equal if the data, granularity and uncertainty are equal.	
Parameters	osid.calendaring.DateTime time	the time to compare
Return	boolean	true if the given time is equal to this one, false otherwise
Errors	NULL_ARGUMENT	time is null
Compliance	mandatory	This method must be implemented.

Enumeration	**osid.calendaring.DateTimeResolution**	
Description	This enumeration contains the possible date/time resolutions.	
Vaues	PLANCKSECOND	ten quattuordecillionth second resolution
	UNDASECOND	one unadecillionth second resolution
	VATOSECOND	one decillionth second resolution
	WEEBLESECOND	one nonillionth second resolution
	XOXXOSECOND	one octillionth second resolution
	YOCTOSECOND	one septllionth second resolution
	ZEPTOSECOND	one sextillionth second second resolution
	ATTOSECOND	one quintillionth second resolution
	FEMTOSECOND	one quadrillionth second resolution
	PICOSECOND	one trillionth second resolution
	NANOSECOND	one billionth second resolution
	MICROSECOND	one millionth second resolution
	MILLISECOND	one thousandth second resolution
	SECOND	second resolution
	MINUTE	minute resolution
	QUARTER_HOUR	15 minute resolution
	HALF_HOUR	30 minute resolution
	HOUR	hour resolution
	DAY	day resolution
	WEEK	week resolution
	MONTH	month resolution
	QUARTER	quarter resolution (jan-mar, apr-jun, jul-sep, oct-dec)
	SEASON	season resolution (spring, winter, summer, fall)
	YEAR	yearly resolution
	BLUEMOON	once in a blue moon
	DECADE	decade resolution
	CENTURY	century resolution
	MILLENNIA	millenium resolution
	GLACIAL	100K years resolution
	EPOCH	1M years resolution
	AEON	1B years resolution
	INFINITY	clock is invalid

Interface	**osid.calendaring.DateTimeList**		
Implements	osid.OsidList		

Description	Like all OsidLists, DateTimeList provides a means for accessing DateTime elements sequentially either one at a time or many at a time. Examples:
	
	`while (dtl.hasNext()) {`
	`  DateTime dt = dtl.getNextDateTime();`
	`}`
	
	or
	
	`while dtl.hasNext()) {`
	`  DateTime[] dts =dtl.getNextDateTimes(dtl.available());`
	`}`		

Method	**getNextTime**		
Description	Gets the next DateTime in this list.		
Return	osid.calendaring.DateTime		the next DateTime in this list. The hasNext() method should be used to test that a next Time is available before calling this method.
Errors	ILLEGAL_STATE		no more elements available in this list
	OPERATION_FAILED		unable to complete request
Compliance	mandatory		This method must be implemented.
Method	**getNextTimes**		
Description	Gets the next set of DateTime elements in this list which must be less than or equal to the number returned from available().		
Parameters	cardinal	n	the number of DateTime elements requested which should be less than or equal to available()
Return	osid.calendaring.DateTime[]		an array of DateTime elements. The length of the array is less than or equal to the number specified.
Errors	ILLEGAL_STATE		no more elements available in this list
	OPERATION_FAILED		unable to complete request
Compliance	mandatory		This method must be implemented.

Interface	**osid.calendaring.DateTimeInterval**		
Implements			
Description	The DateTimeInterval interface defines an interval between two date times.		
Method	**getStart**		
Description	Gets the starting time for this interval.		
Return	osid.calendaring.DateTime		the starting time
Compliance	mandatory		This method must be implemented.
Method	**getEnd**		
Description	Gets the ending time for this interval. The ending time is greater than or equal to the starting time.		
Return	osid.calendaring.DateTime		the ending time
Compliance	mandatory		This method must be implemented.
Method	**isInclusive**		
Description	Tests if the given time interval is included in this one. A time interval is inclusive of this time interval if the start end end times of the given interval are completely contained in this one.		
Parameters	osid.calendaring.DateTimeInterval	interval	the interval to compare
Return	boolean		true if the given time interval is included in this one, false otherwise
Errors	NULL_ARGUMENT		interval is null
Compliance	mandatory		This method must be implemented.
Method	**isExclusive**		
Description	Tests if the given time interval is exclusive in this one. A time is exclsuive of this time interval if there is no overlap between the start end end times.		
Parameters	osid.calendaring.DateTimeInterval	interval	the interval to compare
Return	boolean		true if the given time is exclsuive of this one, false otherwise
Errors	NULL_ARGUMENT		interval is null
Compliance	mandatory		This method must be implemented.
Method	**isEqual**		
Description	Tests if the given time interval is equal to this one. A time interfaval is equal if the start and end times are also equal.		
Parameters	osid.calendaring.DateTimeInterval	interval	the interval to compare
Return	boolean		true if the given time interval is equal to this one, false otherwise
Errors	NULL_ARGUMENT		interval is null
Compliance	mandatory		This method must be implemented.
Method	**isTimeInclusive**		
Description	Tests if the given time is included in this time interval. A time is inclusive of this time interval if the start time and its granularity and uncertainty are completely contained in this interval.		
Parameters	osid.calendaring.DateTime	time	the date time to compare
Return	boolean		true if the given time is included in this interval, false otherwise
Errors	NULL_ARGUMENT		time is null
Compliance	mandatory		This method must be implemented.
Method	**isTimeExclusive**		
Description	Tests if the given time is exclusive in this time interval. A time is exclusive of this time interval if the start time and its granularity and uncertainty are completely outside this interval.		
Parameters	osid.calendaring.DateTime	time	the date time to compare
Return	boolean		true if the given time is exclsuive of this one, false otherwise
Errors	NULL_ARGUMENT		time is null
Compliance	mandatory		This method must be implemented.

Interface	**osid.calendaring.DateTimeIntervalList**		
Implements	osid.OsidList		

Description	Like all OsidLists, DateTimeIntervalList provides a means for accessing DateTimeInterval elements sequentially either one at a time or many at a time. Examples:
	
	`while (til.hasNext()) {`
	`  TimeInterval ti = til.getNextTimeInterval();`
	`}`
	
	or
	
	`while (til.hasNext()) {`
	`  TimeInterval[] tis = til.getNextTimeIntervals(til.available());`
	`}`		

Method	**getNextDateTimeInterval**		
Description	Gets the next DateTimeInterval in this list.		
Return	osid.calendaring.DateTimeInterval		the next DateTimeInterval in this list. The hasNext() method should be used to test that a next DateTimeInterval is available before calling this method.
Errors	ILLEGAL_STATE		no more elements available in this list
	OPERATION_FAILED		unable to complete request
Compliance	mandatory		This method must be implemented.
Method	**getNextDateTimeIntervals**		
Description	Gets the next set of DateTimeInterval elements in this list which must be less than or equal to the number returned from available().		
Parameters	cardinal	n	the number of DateTimeInterval elements requested which should be less than or equal to available()
Return	osid.calendaring.DateTimeInterval[]		an array of DateimeInterval elements. The length of the array is less than or equal to the number specified.
Errors	ILLEGAL_STATE		no more elements available in this list
	OPERATION_FAILED		unable to complete request
Compliance	mandatory		This method must be implemented.