

Date and Time on the Internet

Status of this memo

This document is an Internet Draft. Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a ``working draft'' or ``work in progress``.

To learn the current status of any Internet-Draft, please check the `1id-abstracts.txt` listing contained in the Internet-Drafts Shadow Directories on `ds.internic.net`, `nic.nordu.net`, `ftp.isi.edu`, or `munniari.oz.au`.

A revised version of this draft document will be submitted to the IESG as a Proposed Standard for the Internet Community. Discussion and suggestions for improvement are requested. This document will expire six months after publication. Distribution of this draft is unlimited.

1. Introduction

Date and time formats cause a lot of confusion and interoperability problems on the Internet. This document will address many of the problems encountered and make recommendations to improve consistency and interoperability when representing and using date and time in Internet protocols.

This document includes an Internet profile of the ISO 8601 [[ISO8601](#)] standard for representation of dates and times using the Gregorian calendar.

Changes from draft -00:

- * added some more definitions and references (fixed NTP reference)

- * language clarifications
- * include rules for day of month and leap seconds
- * disallow hour of 24
- * add registry of named timezones and local time format
- * use . instead of , for fractions of second
- * removed references to AM/PM
- * simplified [appendix B](#) program
- * added [appendix C](#) program to calculate leap year

Controversial in last draft, but unchanged:

- * use of "T" as date/time separator. Proposal is to use " " instead, but my reading of ISO 8601 does not permit that.
- * suggestion to make minutes offset from UTC optional.
- * interpretation of 2 digit years

Open issues:

- * See controversial issues above. More comment is welcome.
- * Are more definitions needed?
- * A number of the timezones in the initial registry list are duplicates for future times. I already removed the Indiana county ones since they all duplicate Indianapolis for future times.
- * Need reference to good article demonstrating year 2000 problems.
- * Need commentary on registry and to set up address for registry.
- * Will add [appendix D](#), E with sample POSIX generation/parsing code for [section 5.6](#). Markus Kuhn is working on code.

2. Definitions

UTC	Coordinated Universal Time as maintained by the Bureau International de l'Heure (International Time Bureau).
second	A basic unit of measurement of time in the International System of Units.
minute	A period of time of 60 seconds.
hour	A period of time of 60 minutes.
day	A period of time of 24 hours.
leap year	In the Gregorian calendar, a year which has 366 days. A leap year is a year whose number is divisible by four an integral number of times, except that if it is a centennial year it shall be divisible by four hundred an integral number of times.
ABNF	Augmented Backus-Naur Form, a format used to represent

permissible strings in a protocol or language, as defined in [\[IMAIL\]](#).

Email Date/Time Format

The date/time format used by Internet Mail as defined by [RFC 822](#) [\[IMAIL\]](#) and amended by [RFC 1123](#) [\[HOST-REQ\]](#).

Internet Date/Time Format

The date format defined in [section 5](#) of this document.

For more information about time scales, see [Appendix E](#) of [\[NTP\]](#), Section 3 of [\[ISO8601\]](#), and the appropriate ITU documents [\[ITU-R-TF\]](#).

3. Two Digit Years

Two digit years are expected to cause great expense to many as the year 2000 approaches. Many existing computer programs simply add or subtract 1900 from a two digit year. Such programs will clearly stop functioning on the year 2000 and will have to be upgraded, possibly at great expense [XXX - ref to article on IRS year 2000 problems would be cool]. The following requirements are made of Internet protocols to address this problem:

- o Internet Protocols MUST generate four digit years in dates.
- o If a two digit year is received, the values 00-49 MUST be interpreted as referring to the 21st century (add 2000) and the values 50-99 MUST be interpreted as referring to the 20th century (add 1900). While different interpretations may result in a few more years of 2-digit usability for some applications, it is believed that a single interpretation for two digit years in all Internet protocols will result in better interoperability. In addition, it is reasonable to expect all Internet Protocols using 2 digit dates to be upgraded by the year 2050.
- o It is possible that a program using two digit years will represent years after 1999 as three digits. This occurs if the program simply subtracts 1900 from the year and doesn't check the number of digits. Programs wishing to robustly deal with dates generated by such broken software may add 1900 to three digit years.
- o It is possible that a program using two digit years will represent years after 1999 as ":0", ":1", ... ":9", ";0", ... This occurs if the program simply subtracts 1900 from the year

and adds the decade to the US-ASCII character zero. Programs wishing to robustly deal with dates generated by such broken software should detect non-numeric decades and interpret appropriately.

The problems with two digit years amply demonstrate why all dates and times used in Internet protocols MUST be fully qualified.

4. Local Time

4.1. Coordinated Universal Time (UTC)

Because the daylight rules for local timezones are so convoluted and can change based on local law at unpredictable times, true interoperability is best achieved by using Coordinated Universal Time (UTC).

4.2. Local Offsets

The offset between local time and UTC is often useful information. For example, in electronic mail [[IMAIL](#)] the local offset provides a useful heuristic to determine the probability of a prompt response. Attempts to label local offsets with alphabetic strings have resulted in poor interoperability in the past [[IMAIL](#)], [[HOST-REQ](#)]. Therefore numeric offsets are now REQUIRED in Internet Mail Date/Time Format.

Numeric offsets are calculated as "local time minus UTC". So the equivalent time in UTC can be determined by subtracting the offset from the local time. For example, 18:50:00-04:00 is the same time as 22:58:00Z.

4.3. Unknown Local Offset Convention

If the time in UTC is known, but the offset to local time is unknown, this can be represented with an offset of "-00:00". This differs semantically from an offset of "Z" which implies that UTC is the preferred reference point for the specified time. This convention MAY also be used in the Email Date/Time Format.

4.4. Unqualified Local Time

A number of devices currently connected to the Internet run their internal clocks in local time and are unaware of UTC. While the

Internet does have a tradition of accepting reality when creating specifications, this should not be done at the expense of interoperability. Since interpretation of an unqualified local timezone will fail in approximately 23/24 of the globe, the interoperability problems of unqualified local time are deemed unacceptable for the Internet. Devices which are unaware of the time in UTC MUST use one of the following techniques when communicating on the Internet:

- o Use Network Time Protocol [[NTP](#)] to obtain the time in UTC.
- o Use another host in the same local timezone as a gateway to the Internet. This host MUST correct unqualified local times before they are transmitted to other hosts.
- o Prompt the user for the local timezone and daylight savings settings.

[5.](#) Date and Time formats

This section discusses desirable qualities of date and time formats and defines a profile of ISO 8601 for use in new Internet protocols. Email Date/Time Format lacks many of these characteristics and its use in new protocols is discouraged.

[5.1.](#) Ordering

If date and time components are ordered from least precise to most precise, then a useful property is achieved. Assuming that the timezones of the dates and times are the same (e.g. all in UTC), then the date and time strings may be sorted as strings (e.g. using the `strcmp()` function in C) and a time-ordered sequence will result. The presence of optional punctuation would violate this characteristic.

[5.2.](#) Human Readability

Human readability has proved to be a valuable feature of Internet protocols. Human readable protocols greatly reduce the costs of debugging since telnet often suffices as a test client and network analysers need not be modified with knowledge of the protocol. On the other hand, human readability sometimes results in interoperability problems. For example, the date format "10/11/1996" is completely unsuitable for global interchange because it is interpreted differently in different countries. In addition, the date format in [[IMAIL](#)] has resulted in

interoperability problems when people assumed any text string was permitted and translated the three letter abbreviations to other languages or substituted date formats which were easier to generate (e.g. the format used by the C function ctime). For this reason, a balance must be struck between human readability and interoperability.

Because no date and time format is readable according to the conventions of all countries, Internet clients SHOULD be prepared to transform dates into a display format suitable for the locality. This may include translating UTC to local time.

5.3. Rarely Used Options

A format which includes rarely used options is likely to cause interoperability problems. This is because rarely used options are less likely to be used in alpha or beta testing, so bugs in parsing are less likely to be discovered. Rarely used options should be made mandatory or omitted for the sake of interoperability whenever possible.

The format defined below includes only one rarely used option: fractions of a second. It is expected that this will be used only by applications which require strict ordering of date/time stamps or which have an unusual precision requirement.

5.4. Redundant Information

If a date/time format includes redundant information, that introduces the possibility that the redundant information will not correlate. For example, including the day of the week in a date/time format introduces the possibility that the day of week is incorrect but the date is correct, or vice versa. Since it is not difficult to compute the day of week from a date (see [Appendix B](#)), the day of week should not be included in a date/time format.

5.5. Simplicity

The complete set of date and time formats specified in ISO 8601 [[ISO8601](#)] is quite complex in an attempt to provide multiple representations and partial representations. [Appendix A](#) contains an attempt to translate the complete syntax of ISO 8601 into ABNF as defined in [[IMAIL](#)]. Internet protocols have somewhat different requirements and simplicity has proved to be an important characteristic. In addition, Internet protocols usually need

complete specification of data in order to achieve true interoperability. Therefore, the complete grammar for ISO 8601 is deemed too complex for most Internet protocols.

The following section defines a profile of ISO 8601 for use on the Internet. It is a conformant subset of the ISO 8601 extended format. Simplicity is achieved by making most fields and punctuation mandatory.

[5.6.](#) Internet Date/Time Format

The following profile of ISO 8601 [[ISO8601](#)] dates SHOULD be used in new protocols on the Internet. This is specified using ABNF as defined in [[IMAIL](#)].

```
date-fullyear    = 4DIGIT
date-month       = 2DIGIT ; 01-12
date-mday        = 2DIGIT ; 01-28, 01-29, 01-30, 01-31 based on month/year
time-hour        = 2DIGIT ; 00-23
time-minute      = 2DIGIT ; 00-59
time-second      = 2DIGIT ; 00-59, 00-60 based on leap second rules
time-secfrac     = "." 1*DIGIT
time-numoffset   = ("+" / "-") time-hour ":" time-minute
time-offset      = "Z" / time-numoffset

partial-time     = time-hour ":" time-minute ":" time-second
                  [time-secfrac]
full-date        = date-fullyear "-" date-month "-" date-mday
full-time        = partial-time time-offset
date-time        = full-date "T" full-time
```

[5.7.](#) Restrictions

The grammar element date-mday represents the day number within the current month. The maximum value varies based on the month and year as follows:

Month Number	Month/Year	Maximum value of date-mday
-----	-----	-----
01	January	31
02	February, normal	28
02	February, leap year	29
03	March	31
04	April	30
05	May	31
06	June	30
07	July	31
08	August	31
09	September	30
10	October	31
11	November	30
12	December	31

[Appendix C](#) contains sample C code to determine if a year is a leap year.

The grammar element time-second may have the value "60" at the end of June (XXXX-06-30T23:59:60) or December (XXXX-12-31T23:59:60). At all other times the maximum value of time-second is "59".

Although ISO 8601 permits the hour to be "24", this profile of ISO 8601 only allows values between "00" and "23" for the hour in order to reduce confusion.

[5.8](#). Examples

Here are three examples of Internet date/time format.

1985-04-12T23:20:50.52Z

This represents 20 minutes and 50.52 seconds after the 23rd hour of April 12th, 1985 in UTC.

1996-12-19T16:39:57-08:00

This represents 39 minutes and 57 seconds after the 16th hour of December 19th, 1996 with an offset of -08:00 from UTC (Pacific Standard Time). Note that this is equivalent to 1996-12-20T00:39:57Z in UTC.

1990-12-31T23:59:60Z

This represents the leap second inserted at the end of 1990.

6. Future Events in Local Time

Some applications (e.g. calendaring) require the representation of repeating or future dates in local time. Because the conversion rules between UTC and local time may change by season and political whim, it is necessary to label the local time zone with a standard label so that if new conversion rules are issued the interpretation of the time relative to UTC can be corrected. For this reason, an IANA registry of timezone names which may be used to represent future dates is necessary.

6.1. Problems Too Hard to Solve

Since local timezone rules are set by local governments, the only authoritative reference for such rules is those governments, most of which do not currently provide their rules on line in a computer parsible format. In addition, local timezones were historically set by cities and towns, so attempting to exhaustively enumerate all historical timezones for use in representing past dates is not practical. Attempting to predict where new timezones will be created as a subset of the area covered by an old timezone is also a hopeless prospect.

Therefore the only formal part of the registry will be names for a minimal set of modern timezones. As a convenience, the registry will also include the base UTC offset and daylight savings rules (if determinable) at the time of registration. Because the UTC offset and rules may be changed by other bodies, they will not be considered an authoritative part of the registry.

6.2. Prior Art

An informal collection of timezone information is currently being maintained by volunteer Internet participants. The current location of this information is:

[<ftp://elsie.nci.nih.gov/pub/>](ftp://elsie.nci.nih.gov/pub/)

This is valuable work, and is used in some operating systems. The initial set of timezone names for the IANA registry is a subset of the names collected by this effort.

6.3. Legal Characters in Timezone Names

Only the US-ASCII characters A-Z, a-z, 0-9, "-", "_" and "/" are

legal in timezone names. The basic format is the name of a continent or ocean followed by a "/" followed by the name of a city or political entity. A political entity may be followed by a "/" and a subentity if necessary. Timezone names SHOULD use the standard case in the registry, but MUST be interpreted in a case insensitive manner. New timezone names SHOULD use "-" rather than "_", as the latter is difficult to see in some output contexts.

6.4. Template for IANA Registration of Timezone Names

To: timezone@XXX
Subject: Timezone Name Registration

Timezone Name:

ISO 3166 2-character country code:

Description:

6.5. Procedure for IANA Registration of Timezone Names

The IESG is responsible for appointing a reviewer of Timezone Names. The job of this reviewer is to verify that the new timezone name has unique UTC rules, is likely to be used, fits the rules in [section 6.3](#) and does not conflict unnecessarily with prior art (especially that mentioned in [section 6.2](#)). Within two weeks of posting, the reviewer must take one of the following actions:

- (1) Pass the registration proposal to IANA.
- (2) Reject the registration proposal.
- (3) Recommend alterations to the registration proposal likely to make it acceptable.

In order to assist the reviewer, the address timezone@XXX will be a public mailing list where registration proposals may be discussed. Subscription and unsubscription requests may be sent to timezone-request@XXX.

6.6. Initial List of IANA Timezone Names

The following list will serve as the initial list of IANA Timezone Names. This list was generated from the archive mentioned in [section 6.2](#). Some of these timezone names (especially within the

same country) are redundant for future dates, but compatibility with the timezone names in the databases discussed [section 6.2](#). is useful.

Timezone Name	Country
-----	-----
Europe/Andorra	AD
Asia/Dubai	AE
Asia/Kabul	AF
America/Antigua	AG
America/Anguilla	AI
Europe/Tirane	AL
Asia/Yerevan	AM
America/Curacao	AN
Africa/Luanda	AO
Antarctica/Casey	AQ
Antarctica/DumontDURville	AQ
Antarctica/Mawson	AQ
Antarctica/McMurdo	AQ
Antarctica/Palmer	AQ
Antarctica/South_Pole	AQ
America/Buenos_Aires	AR
America/Catamarca	AR
America/Cordoba	AR
America/Jujuy	AR
America/Mendoza	AR
America/Rosario	AR
Pacific/Pago_Pago	AS
Europe/Vienna	AT
Australia/Adelaide	AU
Australia/Brisbane	AU
Australia/Broken_Hill	AU
Australia/Darwin	AU
Australia/Hobart	AU
Australia/Lindeman	AU
Australia/Lord_Howe	AU
Australia/Melbourne	AU
Australia/Perth	AU
Australia/Sydney	AU
America/Aruba	AW
Asia/Baku	AZ
Europe/Sarajevo	BA
America/Barbados	BB
Asia/Dacca	BD
Europe/Brussels	BE
Africa/Ouagadougou	BF
Europe/Sofia	BG
Asia/Bahrain	BH

Africa/Bujumbura	BI
Africa/Porto-Novo	BJ
Atlantic/Bermuda	BM
Asia/Brunei	BN
America/La_Paz	BO
America/Cuiaba	BR
America/Fortaleza	BR
America/Maceio	BR
America/Manaus	BR
America/Noronha	BR
America/Porto_Acre	BR
America/Sao_Paulo	BR
America/Nassau	BS
Asia/Thimbu	BT
Africa/Gaborone	BW
Europe/Minsk	BY
America/Belize	BZ
America/Dawson	CA
America/Dawson_Creek	CA
America/Edmonton	CA
America/Glace_Bay	CA
America/Goose_Bay	CA
America/Halifax	CA
America/Inuvik	CA
America/Iqaluit	CA
America/Montreal	CA
America/Nipigon	CA
America/Pangnirtung	CA
America/Rainy_River	CA
America/Rankin_Inlet	CA
America/Regina	CA
America/St_Johns	CA
America/Swift_Current	CA
America/Thunder_Bay	CA
America/Vancouver	CA
America/Whitehorse	CA
America/Winnipeg	CA
America/Yellowknife	CA
Indian/Cocos	CC
Africa/Bangui	CF
Africa/Brazzaville	CG
Europe/Zurich	CH
Africa/Abidjan	CI
Pacific/Rarotonga	CK
America/Santiago	CL
Pacific/Easter	CL
Africa/Douala	CM
Asia/Chungking	CN

Asia/Harbin	CN
Asia/Kashgar	CN
Asia/Shanghai	CN
Asia/Urumqi	CN
America/Bogota	CO
America/Costa_Rica	CR
America/Havana	CU
Atlantic/Cape_Verde	CV
Indian/Christmas	CX
Asia/Nicosia	CY
Europe/Prague	CZ
Europe/Berlin	DE
Africa/Djibouti	DJ
Europe/Copenhagen	DK
America/Dominica	DM
America/Santo_Domingo	DO
Africa/Algiers	DZ
America/Guayaquil	EC
Pacific/Galapagos	EC
Europe/Tallinn	EE
Africa/Cairo	EG
Africa/El_Aaiun	EH
Africa/Asmera	ER
Africa/Ceuta	ES
Atlantic/Canary	ES
Europe/Madrid	ES
Africa/Addis_Ababa	ET
Europe/Helsinki	FI
Pacific/Fiji	FJ
Atlantic/Stanley	FK
Pacific/Kosrae	FM
Pacific/Ponape	FM
Pacific/Truk	FM
Pacific/Yap	FM
Atlantic/Faeroe	FO
Europe/Paris	FR
Africa/Libreville	GA
Europe/Belfast	GB
Europe/London	GB
America/Grenada	GD
Asia/Tbilisi	GE
America/Cayenne	GF
Africa/Accra	GH
Europe/Gibraltar	GI
America/Godthab	GL
America/Scoresbysund	GL
America/Thule	GL
Africa/Banjul	GM

Africa/Conakry	GN
America/Guadeloupe	GP
Africa/Malabo	GQ
Europe/Athens	GR
Atlantic/South_Georgia	GS
America/Guatemala	GT
Pacific/Guam	GU
Africa/Bissau	GW
America/Guyana	GY
Asia/Hong_Kong	HK
America/Tegucigalpa	HN
Europe/Zagreb	HR
America/Port-au-Prince	HT
Europe/Budapest	HU
Asia/Jakarta	ID
Asia/Jayapura	ID
Asia/Ujung_Pandang	ID
Europe/Dublin	IE
Asia/Gaza	IL
Asia/Jerusalem	IL
Asia/Calcutta	IN
Indian/Chagos	IO
Asia/Baghdad	IQ
Asia/Tehran	IR
Atlantic/Reykjavik	IS
Europe/Rome	IT
America/Jamaica	JM
Asia/Amman	JO
Asia/Ishigaki	JP
Asia/Tokyo	JP
Africa/Nairobi	KE
Asia/Bishkek	KG
Asia/Phnom_Penh	KH
Pacific/Enderbury	KI
Pacific/Kiritimati	KI
Pacific/Tarawa	KI
Indian/Comoro	KM
America/St_Kitts	KN
Asia/Pyongyang	KP
Asia/Seoul	KR
Asia/Kuwait	KW
America/Cayman	KY
Asia/Alma-Ata	KZ
Asia/Aqtau	KZ
Asia/Aqtobe	KZ
Asia/Vientiane	LA
Asia/Beirut	LB
America/St_Lucia	LC

Europe/Vaduz	LI
Asia/Colombo	LK
Africa/Monrovia	LR
Africa/Maseru	LS
Europe/Vilnius	LT
Europe/Luxembourg	LU
Europe/Riga	LV
Africa/Tripoli	LY
Africa/Casablanca	MA
Europe/Monaco	MC
Europe/Chisinau	MD
Indian/Antananarivo	MG
Pacific/Kwajalein	MH
Pacific/Majuro	MH
Europe/Skopje	MK
Africa/Bamako	ML
Africa/Timbuktu	ML
Asia/Rangoon	MM
Asia/Ulan_Bator	MN
Asia/Macao	MO
Pacific/Saipan	MP
America/Martinique	MQ
Africa/Nouakchott	MR
America/Montserrat	MS
Europe/Malta	MT
Indian/Mauritius	MU
Indian/Maldives	MV
Africa/Blantyre	MW
America/Ensenada	MX
America/Mazatlan	MX
America/Mexico_City	MX
America/Tijuana	MX
Asia/Kuala_Lumpur	MY
Asia/Kuching	MY
Africa/Maputo	MZ
Africa/Windhoek	NA
Pacific/Noumea	NC
Africa/Niamey	NE
Pacific/Norfolk	NF
Africa/Lagos	NG
America/Managua	NI
Europe/Amsterdam	NL
Europe/Oslo	NO
Asia/Katmandu	NP
Pacific/Nauru	NR
Pacific/Niue	NU
Pacific/Auckland	NZ
Pacific/Chatham	NZ

Asia/Muscat	OM
America/Panama	PA
America/Lima	PE
Pacific/Gambier	PF
Pacific/Marquesas	PF
Pacific/Tahiti	PF
Pacific/Port_Moresby	PG
Asia/Manila	PH
Asia/Karachi	PK
Europe/Warsaw	PL
America/Miquelon	PM
Pacific/Pitcairn	PN
America/Puerto_Rico	PR
Atlantic/Azores	PT
Atlantic/Madeira	PT
Europe/Lisbon	PT
Pacific/Palau	PW
America/Asuncion	PY
Asia/Qatar	QA
Indian/Reunion	RE
Europe/Bucharest	RO
Asia/Anadyr	RU
Asia/Irkutsk	RU
Asia/Kamchatka	RU
Asia/Krasnoyarsk	RU
Asia/Magadan	RU
Asia/Novosibirsk	RU
Asia/Omsk	RU
Asia/Vladivostok	RU
Asia/Yakutsk	RU
Asia/Yekaterinburg	RU
Europe/Kaliningrad	RU
Europe/Moscow	RU
Europe/Samara	RU
Africa/Kigali	RW
Asia/Riyadh	SA
Pacific/Guadalcanal	SB
Indian/Mahe	SC
Africa/Khartoum	SD
Europe/Stockholm	SE
Asia/Singapore	SG
Atlantic/St_Helena	SH
Europe/Ljubljana	SI
Arctic/Longyearbyen	SJ
Atlantic/Jan_Mayen	SJ
Europe/Bratislava	SK
Africa/Freetown	SL
Europe/San_Marino	SM

Africa/Dakar	SN
Africa/Mogadishu	SO
America/Paramaribo	SR
Africa/Sao_Tome	ST
America/El_Salvador	SV
Asia/Damascus	SY
Africa/Mbabane	SZ
America/Grand_Turk	TC
Africa/Ndjamena	TD
Indian/Kerguelen	TF
Africa/Lome	TG
Asia/Bangkok	TH
Asia/Dushanbe	TJ
Pacific/Fakaofo	TK
Asia/Ashkhabad	TM
Africa/Tunis	TN
Pacific/Tongatapu	TO
Europe/Istanbul	TR
America/Port_of_Spain	TT
Pacific/Funafuti	TV
Asia/Taipei	TW
Africa/Dar_es_Salaam	TZ
Europe/Kiev	UA
Europe/Simferopol	UA
Africa/Kampala	UG
Pacific/Johnston	UM
Pacific/Midway	UM
Pacific/Wake	UM
America/Adak	US
America/Anchorage	US
America/Boise	US
America/Chicago	US
America/Denver	US
America/Detroit	US
America/Indianapolis	US
America/Juneau	US
America/Los_Angeles	US
America/Louisville	US
America/Menominee	US
America/New_York	US
America/Nome	US
America/Phoenix	US
America/Shiprock	US
America/Yakutat	US
Pacific/Honolulu	US
America/Montevideo	UY
Asia/Tashkent	UZ
Europe/Vatican	VA

America/St_Vincent	VC
America/Caracas	VE
America/Tortola	VG
America/St_Thomas	VI
Asia/Saigon	VN
Pacific/Efate	VU
Pacific/Wallis	WF
Pacific/Apia	WS
Asia/Aden	YE
Indian/Mayotte	YT
Europe/Belgrade	YU
Africa/Johannesburg	ZA
Africa/Lusaka	ZM
Africa/Kinshasa	ZR
Africa/Lubumbashi	ZR
Africa/Harare	ZW

[6.7.](#) Local Date/Time Format

The following format MAY be used to refer to future dates in a local timezone. This is defined based on the format in [section 5.6](#).

```
zone-char = ALPHA / DIGIT / "-" / "_" / "/"
zone-name = 1*zone_char
           ; case insensitive interpretation
offset-hint = time-numoffset
```

```
local-datetime = full-date "T" partial-time " " zone-name
                [" " offset-hint]
```

A local-datetime represents an event relative to a specific local timezone. The offset-hint represents the generator's prediction of what the UTC offset will be at that local time, and may become incorrect if the rules for the specified zone are changed. The offset-hint MAY be omitted if the generating program only knows local time, but the zone-name is REQUIRED. This format SHOULD NOT be used for timestamps or past events.

[6.8.](#) Examples

Here are some examples of Local Date/Time Format:

```
1999-12-31T23:59:59 America/New_York -05:00
```

This represents a time one (or two if there's a leap second) second

before the year 2000 in the timezone used in New York City in North America (currently U.S. Eastern Time). The offset-hint is the number to add to the local time to get an estimate UTC for that date, so this will probably be equivalent to 2000-01-01T04:59:59Z.

2000-12-31T23:59:59 Australia/Adelaide +09:30

This represents a time one (or two if there's a leap second) second before the 21st century in Adelaide, Australia. The hint suggests that this will be equivalent to 2000-12-31T14:29:59Z.

2000-03-31T02:00:00 America/Los_Angeles -08:00

The represents a time of the 2nd hour on the 31st of March in Los Angeles, USA. The hint suggests that would be equivalent to 2000-03-31T10:00:00Z. However, if the U.S. government were to adopt the daylight savings rules currently used by the European Union, which change daylight savings on the last Sunday of March, then the time would be equivalent to 2000-03-31T09:00:00Z.

7. Acknowledgements

May thanks to the following people who have provided helpful advice for this document: Ned Freed, Neal McBurnett, David Keegel, Markus Kuhn, Paul Eggert and Robert Elz. Thanks are also due to participants of the IETF Calendaring/Scheduling working group mailing list, and participants of the timezone mailing list.

8. References

[Zeller] Chr. Zeller, "Kalender-Formeln", Acta Mathematica, Vol. 9, Nov 1886.

[IMAIL] Crocker, D., "Standard for the Format of Arpa Internet Text Messages", [RFC 822](#), University of Delaware, August 1982.

<<ftp://ds.internic.net/rfc/rfc822.txt>>

[ISO8601] "Data elements and interchange formats -- Information interchange -- Representation of dates and times", ISO 8601:1988(E), International Organization for Standardization, June, 1988.

[HOST-REQ] Braden, R., "Requirements for Internet Hosts -- Application and Support", [RFC 1123](#), Internet Engineering Task Force, October 1989.

<<ftp://ds.internic.net/rfc/rfc1123.txt>>

[NTP] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis", [RFC 1305](#), University of Delaware, March 1992.

<<ftp://ds.internic.net/rfc/rfc1305.tar.Z>>
<<ftp://ds.internic.net/rfc/rfc1305.txt>>

[ITU-R-TF] International Telecommunication Union Recommendations for Time Signals and Frequency Standards Emissions.

<<http://www.itu.ch/publications/itu-r/iturtf.htm>>

9. Security Considerations

Since the local time zone of a site may be useful for determining a time when systems are less likely to be monitored and might be more susceptible to a security probe, some sites may wish to emit times in UTC only. Others might consider this to be loss of useful functionality at the hands of paranoia.

10. Author's Address

Chris Newman
Innosoft International, Inc.
[1050](#) East Garvey Ave. South
West Covina, CA 91790 USA

Email: chris.newman@innosoft.com

APPENDIX

A. ISO 8601 Collected ABNF

ISO 8601 does not specify a formal grammar for the date and time formats it defines. The following is an attempt to create a formal grammar from ISO 8601. This is informational only and may contain errors. ISO 8601 remains the authoratative reference.

Note that due to ambiguities in ISO 8601, some interpretations had to be made. First, ISO 8601 is not clear if mixtures of basic and extended format are permissible. This grammar permits mixtures. ISO 8601 is not clear on whether an hour of 24 is permissible only if minutes and seconds are 0. This assumes that an hour of 24 is permissible in any context. Restrictions on date-mday in [section 5.7](#) apply. ISO 8601 states that the "T" may be omitted under some circumstances. This grammar requires the "T" to avoid ambiguity.

ISO 8601 also requires (in [section 5.3.1.3](#)) that a decimal fraction be preceded by a "0" if less than unity. Annex B.2 of ISO 8601 gives examples where the decimal fractions are not preceded by a "0". This grammar assumes [section 5.3.1.3](#) is correct and that Annex B.2 is in error.

```

date-century      = 2DIGIT ; 00-99
date-decade       = DIGIT  ; 0-9
date-subdecade    = DIGIT  ; 0-9
date-year         = date-decade date-subdecade
date-fullyear     = date-century date-year
date-month        = 2DIGIT ; 01-12
date-wday         = DIGIT  ; 1-7 ; 1 is Monday, 7 is Sunday
date-mday         = 2DIGIT ; 01-28, 01-29, 01-30, 01-31 based on month/year
date-yday         = 3DIGIT ; 001-365, 001-366 based on year
date-week         = 2DIGIT ; 01-52, 01-53 based on year

```

```

datepart-fullyear = [date-century] date-year ["-"]
datepart-ptyear   = "-" [date-subdecade ["-"]]
datepart-wkyear   = datepart-ptyear / datepart-fullyear

```

```

dateopt-century   = "-" / date-century
dateopt-fullyear  = "-" / datepart-fullyear
dateopt-year      = "-" / (date-year ["-"])
dateopt-month     = "-" / (date-month ["-"])
dateopt-week      = "-" / (date-week ["-"])

```

```

datespec-full     = datepart-fullyear date-month ["-"] date-mday
datespec-year     = date-century / dateopt-century date-year
datespec-month    = "-" dateopt-year date-month ["-"] date-mday
datespec-mday     = "--" dateopt-month date-mday
datespec-week     = datepart-wkyear "W"
                  (date-week / dateopt-week date-wday)
datespec-wday     = "---" date-wday
datespec-yday     = dateopt-fullyear date-yday

```

```

date              = datespec-full / datespec-year / datespec-month /
                  datespec-mday / datespec-week / datespec-wday / datespec-yday

```

Time:

```

time-hour         = 2DIGIT ; 00-24
time-minute       = 2DIGIT ; 00-59
time-second       = 2DIGIT ; 00-59, 00-60 based on leap-second rules
time-fraction     = ("." / ".") 1*DIGIT
time-numoffset    = ("+" / "-") time-hour [":" time-minute]
time-zone         = "Z" / time-numoffset

```

```
timeopt-hour      = "-" / (time-hour [":"])
timeopt-minute    = "-" / (time-minute [":"])

timespec-hour     = time-hour [":"] time-minute [":"] time-second]]
timespec-minute   = timeopt-hour time-minute [":"] time-second]
timespec-second   = "-" timeopt-minute time-second
timespec-base     = timespec-hour / timespec-minute / timespec-second

time              = timespec-base [time-fraction] [time-zone]

iso-date-time     = date "T" time
```

Durations (periods):

```
dur-second        = 1*DIGIT "S"
dur-minute        = 1*DIGIT "M" [dur-second]
dur-hour          = 1*DIGIT "H" [dur-minute]
dur-time          = "T" (dur-hour / dur-minute / dur-second)
dur-day           = 1*DIGIT "D"
dur-week          = 1*DIGIT "W"
dur-month         = 1*DIGIT "M" [dur-day]
dur-year          = 1*DIGIT "Y" [dur-month]
dur-date          = (dur-day / dur-month / dur-year) [dur-time]

duration          = "P" (dur-date / dur-time / dur-week)
```

Periods:

```
period-explicit   = date-time "/" date-time
period-start      = date-time "/" duration
period-end        = duration "/" date-time

period            = period-explicit / period-start / period-end
```

B. Day of the Week

The following is a sample C subroutine loosely based on Zeller's Congruence [[Zeller](#)] which may be used to obtain the day of the week:

```
char *day_of_week(int day, int month, int year)
{
    char *dayofweek[] = {
        "Sunday", "Monday", "Tuesday", "Wednesday",
        "Thursday", "Friday", "Saturday"
    };

    /* adjust months so February is the last one */
    month -= 2;
    if (month < 1) {
        month += 12;
        --year;
    }
    /* split by century */
    cent = year / 100;
    year %= 100;
    return (dayofweek[((26 * month - 2) / 10 + day + year
                        + year / 4 + cent / 4 - 2 * cent) % 7]);
}
```

C. Leap Years

Here's a sample C subroutine to calculate if a year is a leap year:

```
/* This returns non-zero if year is a leap year. Must use 4 digit year.
 */
int leap_year(int year)
{
    return (year % 4 == 0 && (year % 100 != 0 || year % 400 == 0));
}
```