An Ontology of Time for the Semantic Web

JERRY R. HOBBS and FENG PAN
University of Southern California, Information Sciences Institute

In connection with the DAML project for bringing about the Semantic Web, an ontology of time is being developed for describing the temporal content of Web pages and the temporal properties of Web services. This ontology covers topological properties of instants and intervals, measures of duration, and the meanings of clock and calendar terms.

Categories and Subject Descriptors: I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods - *Representations (procedural and rule-based)*; *Temporal logic;* I.2.7 [**Artificial Intelligence**]: Natural Language Processing - *Text analysis*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services - *Web-based services*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing - *Linguistic processing*
General Terms: Design, Documentation, Languages, Theory, Verification
Additional Key Words and Phrases: Ontology, time, semantic web, temporal information, temporal relation, duration, clock and calendar, time zone

1. INTRODUCTION

The DARPA Agent Markup Language (DAML) project is DARPA's effort to bring into reality the semantic Web, in which Web users and automatic agents will be able to access information on the Web via descriptions of the content and capabilities of Web resources rather than key words. An important part of this effort is the development of representative ontologies of the most commonly used domains. We have developed such an ontology of temporal concepts for describing the temporal content of Web pages and the temporal properties of Web services. This effort has been informed by temporal ontologies developed at a number of sites; it is intended to capture the essential features of all of them and make them easily available to a large group of Web developers and users, embedded in the ontology mark-up language OWL.[1]

The bulk of information on the Web is in natural language; this information will be easier to encode for the semantic Web insofar as community-wide annotation and automatic tagging schemes and the DAML time ontology are compatible with each other. Indeed, this compatibility was explored by Hobbs and Pustejovsky [2003].

In this article we outline the temporal ontology. Five categories of temporal concepts are considered, and the principal predicates and their associated properties are described for each category.

This research was supported by the Defense Advanced Research Projects Agency under Air Force Research Laboratory contract F30602-00-C-0168 and by the Advanced Research and Development Agency.
Authors' addresses: University of Southern California / Information Sciences Institute, 4676 Admiralty Way, Marina del Rey, CA 90292, USA. Emails: {hobbs, pan}@isi.edu.

[1] http://www.w3c.org/2001/sw/WebOnt/

A note on notation before we begin: conjunction ($\wedge$) takes precedence over implication ($\supset$) and equivalence ($\equiv$). Formulas are assumed to be universally quantified on the variables appearing in the antecedent of the highest-level implication. Thus,

$$p_1(x) \wedge p_2(y) \supset q_1(x,y) \wedge q_2(y)$$

is to be interpreted as

$$(\forall x,y)[[p_1(x) \wedge p_2(y)] \supset [q_1(x,y) \wedge q_2(y)]]$$

2. TOPOLOGICAL TEMPORAL RELATIONS

2.1 Instants and Intervals

There are two subclasses of *TemporalEntity*: *Instant* and *Interval*.

$$Instant(t) \supset TemporalEntity(t)$$

$$Interval(T) \supset TemporalEentity(T)$$

These are the only two subclasses of temporal entities.

$$(\forall T)[TemporalEntity(T) \supset Interval(T) \vee Instant(T)$$

As we will see, intervals are, intuitively, things with extent and instants are, intuitively, point-like in that they have no interior points. (In what follows, lower case *t* is used for instants, upper case *T* for intervals and for temporal-entities unspecified as to subtype. This is strictly for the reader's convenience, and has no formal significance.)

The predicates *begins* and *ends* are relations between instants and temporal entities.

$$begins(t,T) \supset Instant(t) \wedge TemporalEntity(T)$$

$$ends(t,T) \supset Instant(t) \wedge TemporalEntity(T)$$

For convenience, we can say that the beginning and end of an instant is itself. The converses of these rules are also true.

$$Instant(t) \equiv begins(t,t)$$

$$Instant(t) \equiv ends(t,t)$$

The beginnings and ends of temporal entities, if they exist, are unique.

$$TemporalEntity(T) \wedge begins(t_1,T) \wedge begins(t_2,T) \supset t_1 = t_2$$

$$TemporalEntity(T) \wedge ends(t_1,T) \wedge ends(t_2,T) \supset t_1 = t_2$$

In one approach to infinite intervals, a positively infinite interval has no end and a negatively infinite interval has no beginning. Hence, we use the relations *begins* and *ends* in the core ontology, rather than defining functions *beginningOf* and *endOf*, since the functions would not be total. They can be defined in an extension of the core ontology that posits instants at positive and negative infinity.

The predicate *inside* is a relation between an instant and an interval.

$$inside(t,T) \supset Instant(t) \wedge Interval(T)$$

This concept of *inside* is not intended to include beginnings and ends of intervals, as seen below.

It will be useful in characterizing clock and calendar terms to have a relation between instants and intervals that says that the instant is inside or the beginning of the interval.

$$(\forall t,T)[beginsOrIn(t,T) \equiv [begins(t,T) \vee inside(t,T)]]$$

The predicate *timeBetween* is a relation among a temporal entity and two instants.

$$timeBetween(T,t_1,t_2) \supset TemporalEntity(T) \wedge Instant(t_1) \wedge Instant(t_2)$$

The two instants are the beginning and end points of the temporal entity.

$$(\forall t_1,t_2)[t_1 \neq t_2 \supset (\forall T)[timeBetween(T,t_1,t_2) \equiv begins(t_1,T) \wedge ends(t_2,T)]]$$

The ontology is silent about whether the interval from *t* to *t*, if it exists, is identical to the instant *t*.

The ontology is silent about whether intervals *consist of* instants.

The ontology is silent about whether intervals are uniquely determined by their starts and ends.

The core ontology is silent about whether intervals are uniquely determined by their beginnings and ends.

We can define a proper interval as one whose start and end are not identical.

$$(\forall T)ProperInterval(T)$$
$$\equiv Interval(T) \wedge (\forall t_1,t_2)[begins(t_1,T) \wedge ends(t_2,T) \supset t_1 \neq t_2\,]]$$

The ontology is silent about whether there are any intervals that are not proper intervals.

2.2 Before

There is a *before* relation on temporal entities, which gives directionality to time. If temporal entity T_1 is before temporal entity T_2, then the end of T_1 is before the start of T_2. Thus, *before* can be considered to be basic to instants and derived for intervals.

$$(\forall\, T_1,T_2)[before(T_1,T_2)$$
$$\equiv (\exists\, t_1,t_2)[ends(t_1,T_1) \wedge begins(t_2,T_2) \wedge before(t_1,t_2)]]$$

The *before* relation is anti-reflexive, anti-symmetric, and transitive.

$$before(T_1,T_2) \supset T_1 \neq T_2$$
$$before(T_1,T_2) \supset \neg before(T_2,T_1)$$
$$before(T_1,T_2) \wedge before(T_2,T_3) \supset before(T_1,T_3)$$

The end of an interval is not before the beginning of the interval.

$$Interval(T) \wedge begins(t_1,T) \wedge ends(t_2,T) \supset \neg before(t_2,t_1)$$

The beginning of a proper interval is before the end of the interval.

$$ProperInterval(T) \wedge begins(t_1,T) \wedge ends(t_2,T) \supset before(t_1,t_2)$$

If one instant is before another, there is a time between them.

$$Instant(t_1) \wedge Instant(t_2) \wedge before(t_1,t_2) \supset (\exists\, T)\ timeBetween(T,t_1,t_2)$$

The ontology is silent about whether there is a time between *t* and *t*.

If an instant is inside a proper interval, then the beginning of the interval is before the instant, which is before the end of the interval. This is the principal property of *inside*.

$$inside(t,T) \wedge begins(t_1,T) \wedge ends(t_2,T) \wedge ProperInterval(T)$$
$$\supset before(t_1,t) \wedge before(t,t_2)$$

The relation *after* is defined in terms of *before*.

$$after(T_1,T_2) \equiv before(T_2,T_1)$$

The ontology is silent about whether time is linearly ordered. Thus it supports theories of time, such as the branching futures theory, which conflate time and possibility or knowledge.

The basic ontology is silent about whether time is dense, that is, whether between any two instants there is a third instant. Thus it supports theories in which time consists of discrete instants.

2.3 Interval Relations

The relations between intervals defined in Allen's temporal interval calculus [Allen and Kautz 1985; Allen and Ferguson 1997] can be defined in a straightforward fashion in terms of *before* and identity on the beginning and end points. It is a bit more complicated than the reader might at first suspect, since allowance has to be made for the possibility of infinite intervals. Since one of the intervals could be infinite and lack an end point, the relation between the end points has to be dependent on their existence.

The standard interval calculus assumes all intervals are proper; and we do this here too. The definitions of the interval relations in terms of *before* relations among their beginning and end points, when they exist, are given by the following axioms. In these axioms, t_1 and t_2 are the beginning and end of interval T_1; t_3 and t_4 are the beginning and end of T_2.

$$(\forall\, T_1,T_2)[intEquals(T_1,T_2)$$
$$\equiv [ProperInterval(T_1) \wedge ProperInterval(T_2)$$
$$\wedge\ (\forall\, t_1)[begins(t_1,T_1) \equiv begins(t_1,T_2)]$$
$$\wedge\ (\forall\, t_2)[ends(t_2,T_1) \equiv ends(t_2,T_2)]]]$$

$$(\forall\, T_1,T_2)[intBefore(T_1,T_2)$$
$$\equiv ProperInterval(T_1) \wedge ProperInterval(T_2) \wedge before(T_1,T_2)]$$

$$(\forall\, T_1,T_2)[intMeets(T_1,T_2)$$
$$\equiv [ProperInterval(T_1) \wedge ProperInterval(T_2)$$
$$\wedge\ (\exists\, t)[ends(t,T_1) \wedge begins(t,T_2)]]]$$

$$(\forall\, T_1,T_2)[intOverlaps(T_1,T_2)$$
$$\equiv [ProperInterval(T_1) \wedge ProperInterval(T_2)$$
$$\wedge\ (\exists\, t_2,t_3)[ends(t_2,T_1) \wedge begins(t_3,T_2) \wedge before(t_3,t_2)$$
$$\wedge\ (\forall\, t_1)[begins(t_1,T_1) \supset before(t_1,t_3)]$$
$$\wedge\ (\forall\, t_4)[ends(t_4,T_2) \supset before(t_2,t_4)]]]]$$

$$(\forall\, T_1,T_2)[intStarts(T_1,T_2)$$
$$\equiv [ProperInterval(T_1) \wedge ProperInterval(T_2)$$
$$\wedge\ (\exists\, t_2)[ends(t_2,T_1) \wedge (\forall\, t_1)[begins(t_1,T_1) \equiv begins(t_1,T_2)]$$
$$\wedge\ (\forall\, t_4)[ends(t_4,T_2) \supset before(t_2,t_4)]]]]$$

$$(\forall\, T_1,T_2)[intDuring(T_1,T_2)$$
$$\equiv [ProperInterval(T_1) \wedge ProperInterval(T_2)$$
$$\wedge\ (\exists\, t_1,t_2)[begins(t_1,T_1) \wedge ends(t_2,T_1)$$
$$\wedge\ (\forall\, t_3)[begins(t_3,T_2) \supset before(t_3,t_1)]$$
$$\wedge\ (\forall\, t_4)[ends(t_4,T_2) \supset before(t_2,t_4)]]]]$$

$$(\forall\ T_1,T_2)[intFinishes(T_1,T_2)$$
$$\equiv [ProperInterval(T_1)\ \wedge\ ProperInterval(T_2)$$
$$\wedge\ (\exists\ t_1)[begins(t_1,T_1)\ \wedge\ (\forall\ t_3)[begins(t_3,T_2)\supset before(t_3,t_1)]$$
$$\wedge\ (\forall\ t_4)[ends(t_4,T_2)\equiv ends(t_4,T_1)]]]]$$

The inverse interval relations can be defined in terms of these relations.

$$intAfter(T_1,T_2)\equiv intBefore(T_2,T_1)$$
$$intMetBy(T_1,T_2)\equiv intMeets(T_2,T_1)$$
$$intOverlappedBy(T_1,T_2)\equiv intOverlaps(T_2,T_1)$$
$$intStartedBy(T_1,T_2)\equiv intStarts(T_2,T_1)$$
$$intContains(T_1,T_2)\equiv intDuring(T_2,T_1)$$
$$intFinishedBy(T_1,T_2)\equiv intFinishes(T_2,T_1)$$

In addition, it will be useful below to have a single predicate for intervals intersecting in at most an instant.

$$nonoverlap(T_1,T_2)$$
$$\equiv [intBefore(T_1,T_2)\vee intAfter(T_1,T_2)\vee intMeets(T_1,T_2)$$
$$\vee intMetBy(T_1,T_2)]$$

We could have as easily defined this in terms of *before* relations on the beginnings and ends of the intervals.

So far the concepts and axioms in the ontology of time would be appropriate for scalar phenomena in general.

2.4 Linking Time and Events

The time ontology links to other things in the world through four predicates: *atTime*, *during*, *holds*, and *timeSpan*. We assume that another ontology provides for the description of events, either a general ontology of event structure abstractly conceived, or specific, domain-dependent ontologies for specific domains.

The term “eventuality" will be used to cover events, states, processes, propositions, states of affairs, and anything else that can be located with respect to time. The possible natures of eventualities would be spelled out in the event ontologies. The term “eventuality" in this article is only an expositional convenience and has no formal role in the time ontology.

The predicate *atTime* relates an eventuality to an instant, and is intended to say that the eventuality holds, obtains, or is taking place at that time.

$$atTime(e,t)\supset Instant(t)$$

The predicate *during* relates an eventuality to an interval, and is intended to say that the eventuality holds, obtains, or is taking place during that interval.

$$during(e,T)\supset Interval(T)$$

If an eventuality obtains during an interval, it obtains at every instant inside the interval and during every subinterval.

$$during(e,T)\ \wedge\ inside(t,T)\supset atTime(e,t)$$
$$during(e,T)\ \wedge\ intDuring(T_1,T)\supset during(e,T_1)$$

Note that this means that an intermittant activity, like writing a book, does not hold "during" the interval from the beginning to the end of the activity. Rather the "convex hull" of the activity holds "during" the interval.

Whether a particular process is viewed as instantaneous or as occuring over an interval is a granularity decision that may vary according to the context of use, and is assumed to be provided by the event ontology.

Often, the eventualities in the event ontology are best thought of as propositions; the relation between these and times is most naturally called *holds*. The predication *holds*(*e*,*T*) would say that *e* holds at instant *T* or during interval *T*. The predicate *holds* would be part of the event ontology, not part of the time ontology, although its second argument would be provided by the time ontology. The designers of the event ontology may or may not want to relate *holds* to *atTime* and *during* by axioms such as the following:

$$holds(e,t) \wedge Instant(t) \equiv atTime(e,t)$$
$$holds(e,T) \wedge Interval(T) \equiv during(e,T)$$

Similarly, the event ontology may provide other ways of linking events with times, for example, by including a time parameter in predications.

$$p(x,t)$$

This time ontology provides ways of reasoning about the *t*'s; their use as arguments of predicates from another domain would be a feature of the ontology of the other domain.

The predicate *timeSpan* relates eventualities to instants or intervals (or temporal sequences of instants and intervals). For contiguous states and processes, it tells the entire instant or interval for which the state or process obtains or takes place.

$$timeSpan(T,e) \supset TemporalEntity(T) \vee tseq(T)^2$$
$$timeSpan(T,e) \wedge Interval(T) \supset during(e,T)$$
$$timeSpan(t,e) \wedge Instant(t) \supset atTime(e,t)$$
$$timeSpan(T,e) \wedge Interval(T) \wedge \neg inside(t,T) \wedge \neg begins(t,T) \wedge \neg ends(t,T) \supset \neg atTime(e,t)$$
$$timeSpan(t,e) \wedge Instant(t) \wedge t_1 \neq t \supset \neg atTime(e,t_1)$$

Whether the eventuality obtains at the start and end points of its time span is a matter for the event ontology to specify. The silence here on this issue is the reason *timeSpan* is not defined in terms of necessary and sufficient conditions.

In an extension of the time ontology, we also allow temporal predicates to apply directly to events, should the user wish. Thus, *begins*(*t*,*e*) says that the instant *t* begins the interval that is the time span of eventuality *e*; see the documentation[3] for details.

Different communities have different ways of representing the times and durations of states and events (or processes). In one approach, states and events can both have durations, and at least events can be instantaneous. In another approach, events can only be instantaneous and only states can have durations. In the latter approach, events that one might consider as having duration (e.g., heating water) are modeled as a state of the system that is initiated and terminated by instantaneous events. That is, there is the instantaneous event of the start of the heating at the start of an interval, which transitions the system into a state in which the water is heating. The state continues until another instantaneous event occurs—the stopping of the heating at the end of the interval. These two perspectives on events are straightforwardly interdefinable in terms of the ontology

[2] *tseq*(*T*): T is a temporal sequence.
[3] http://www.isi.edu/~pan/damltime/time-entry-documentation.txt

we have provided. This is a matter for the event ontology to specify; this time ontology is neutral with respect to the choice.

3. MEASURING DURATIONS

3.1 Temporal Units

This development assumes that ordinary arithmetic is available.

There are at least two approaches that can be taken toward measuring intervals. The first is to consider units of time as functions from Intervals to Reals. Owing to infinite intervals, the range must also include Infinity.

minutes: Intervals → Reals ∪ {Infinity}
minutes([5:14,5:17]) = 3

The other approach is to consider temporal units to constitute a set of entities, call it TemporalUnits, and have a single function *duration* mapping Intervals × TemporalUnits into the Reals.

duration: Intervals × TemporalUnits → Reals ∪ {Infinity}
duration([5:14,5:17],**Minute**) = 3

The two approaches are interdefinable:

seconds(*T*) = *duration*(*T*,**Second**)
minutes(*T*) = *duration*(*T*,**Minute**)
hours(*T*) = *duration*(*T*,**Hour**)
days(*T*) = *duration*(*T*,**Day**)
weeks(*T*) = *duration*(*T*,**Week**)
months(*T*) = *duration*(*T*,**Month**)
years(*T*) = *duration*(*T*,**Year**)

Ordinarily, the first is more convenient for stating specific facts about particular units; the second is more convenient for stating general facts about all units.

The arithmetic relations among the various units are as follows:

seconds(*T*) = 60∗*minutes*(*T*)
minutes(*T*) = 60∗*hours*(*T*)
hours(*T*) = 24∗*days*(*T*)
days(*T*) = 7∗*weeks*(*T*)
months(*T*) = 12∗*years*(*T*)

The relation between days and months (and, to a lesser extent, years) are specified as part of the ontology of clock and calendar, below. On their own, however, month and year are legitimate temporal units.

3.2 Concatenation and *Hath*

The multiplicative relations above don't tell the whole story of the relations among temporal units. Temporal units are *composed of* smaller temporal units. A larger temporal unit is a concatenation of smaller temporal units. We first define a general relation of concatenation between an interval and a set of smaller intervals. We then introduce a predicate *Hath* that specifies the number of smaller unit intervals that concatenate to a larger interval.

Concatenation: A proper interval *x* is a concatenation of a set *S* of proper intervals if and only if *S* covers all of *x*, and all members of *S* are subintervals of *x* and are mutually disjoint. (The third conjunct on the right side of ≡ is because *beginsOrIn* only covers instants that begin or are inside *x*.)

$$
\begin{aligned}
&\mathit{concatenation}(x,S)\\
&\quad \equiv \mathit{ProperInterval}(x)\\
&\quad\ \wedge\ (\forall\, z)[\mathit{beginsOrIn}(z,x) \supset (\exists\, y)[\mathit{member}(y,S)\ \wedge\ \mathit{beginsOrIn}(z,y)]]\\
&\quad\ \wedge\ (\forall\, z)[\mathit{ends}(z,x) \supset (\exists\, y)[\mathit{member}(y,S)\ \wedge\ \mathit{ends}(z,y)]]\\
&\quad\ \wedge\ (\forall\, y)[\mathit{member}(y,S)\\
&\qquad\qquad \supset [\mathit{intStarts}(y,x) \vee \mathit{intDuring}(y,x) \vee \mathit{intFinishes}(y,x)\\
&\qquad\qquad\qquad \vee\ \mathit{intEquals}(y,x)]]\\
&\quad\ \wedge\ (\forall\, y_1,y_2)[\mathit{member}(y_1,S)\ \wedge\ \mathit{member}(y_2,S)\\
&\qquad\qquad \supset [y_1 = y_2\ \vee \mathit{nonoverlap}(y_1,y_2)]]
\end{aligned}
$$

The following properties of *concatenation* can be proved as theorems:
There are elements in S that start and finish x:

$$\mathit{concatenation}(x,S) \supset (\exists\, !\, y_1)[\mathit{member}(y_1,S)\ \wedge\ \mathit{intStarts}(y_1,x)]$$

$$\mathit{concatenation}(x,S) \supset (\exists\, !\, y_2)[\mathit{member}(y_2,S)\ \wedge\ \mathit{intFinishes}(y_2,x)]$$

If S is a singleton set, its single element is x.

$$\mathit{concatenation}(x,S)\ \wedge\ \mathit{card}(S) = 1 \supset S = \{x\}$$

The property of convexity holds in the ontology if and only if the end points of an interval uniquely determine it. This is an assumption the user can make for any application, and will normally want to.

$$\mathit{Convex}() \equiv (\forall\, T_1,T_2)[\mathit{intEquals}(T_1,T_2) \equiv T_1 = T_2]$$

If convexity holds, then except for the first and last elements of S, every element of S has elements that precede and follow it.

$$
\begin{aligned}
&\mathit{Convex}() \supset\\
&\quad [\mathit{concatenation}(x,S)\\
&\qquad \supset (\forall\, y_1)[\mathit{member}(y_1,S)\\
&\qquad\qquad \supset [\mathit{intFinishes}(y_1,x)\\
&\qquad\qquad\quad \vee (\exists\, !\, y_2)[\mathit{member}(y_2,S)\ \wedge\ \mathit{intMeets}(y_1,y_2)]]]]
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{Convex}() \supset\\
&\quad [\mathit{concatenation}(x,S)\\
&\qquad \supset (\forall\, y_2)[\mathit{member}(y_2,S)\\
&\qquad\qquad \supset [\mathit{intStarts}(y_2,x)\\
&\qquad\qquad\quad \vee (\exists\, !\, y_1)[\mathit{member}(y_1,S)\ \wedge\ \mathit{intMeets}(y_1,y_2)]]]]
\end{aligned}
$$

The uniqueness (∃ !) follows from non-overlap.

Hath: The basic predicate used here for expressing the composition of larger intervals out of smaller clock and calendar intervals is *Hath*, from statements like "30 days hath September" and "60 minutes hath an hour." Its structure is

$$\mathit{Hath}(N,u,x)$$

meaning "N proper intervals of duration one unit u hath the proper interval x." That is, if $\mathit{Hath}(N,u,x)$ holds, then x is the concatenation of N unit intervals where the unit is u. For example, if x is some month of September then $\mathit{Hath}(30,*\mathit{Day}*,x)$ would be true.

Hath is defined as follows:

$$
\begin{aligned}
&\mathit{Hath}(N,u,x) \equiv (\exists\, S)[\mathit{card}(S) = N\ \wedge\ (\forall\, z)[\mathit{member}(z,S) \supset \mathit{duration}(z,u) = 1]\\
&\qquad\qquad \wedge\ \mathit{concatenation}(x,S)]
\end{aligned}
$$

That is, x is the concatenation of a set S of N proper intervals of duration one unit u.

The type constraints on its arguments can be proved as a theorem: N is an integer (assuming that is the constraint on the value of *card*), u is a temporal unit, and x is a proper interval:

$$Hath(N,u,x) \supset integer(N) \wedge TemporalUnit(u) \wedge ProperInterval(x)$$

This treatment of *concatenation* will work for scalar phenomena in general. This treatment of *Hath* will work for measurable quantities in general.

3.3 The Structure of Temporal Units

We now define predicates true of intervals that are 1 temporal unit long. For example, *week* is a predicate true of intervals whose duration is one week.

$$second(T) \equiv seconds(T) = 1$$
$$minute(T) \equiv minutes(T) = 1$$
$$hour(T) \equiv hours(T) = 1$$
$$day(T) \equiv days(T) = 1$$
$$week(T) \equiv weeks(T) = 1$$
$$month(T) \equiv months(T) = 1$$
$$year(T) \equiv years(T) = 1$$

We are now in a position to state the relations between successive temporal units.

$$minute(T) \supset Hath(60,*Second*,T)$$
$$hour(T) \supset Hath(60,*Minute*,T)$$
$$day(T) \supset Hath(24,*Hour*,T)$$
$$week(T) \supset Hath(7,*Day*,T)$$
$$year(T) \supset Hath(12,*Month*,T)$$

The relations between months and days are discussed in Section 4.5.

4. CLOCK AND CALENDAR

4.1 Time Zones

What hour of the day an instant is in is relative to the time zone. This is also true of minutes, since there are regions in the world, e.g., central Australia, where the hours are not aligned with GMT hours, but are, e.g., offset half an hour. To our knowledge, seconds are not relative to the time zone.

Days, weeks, months, and years are also relative to the time zone, since, e.g., 2004 began in the Eastern Standard time zone, three hours before it began in the Pacific Standard time zone. Thus, predications about all clock and calendar intervals except seconds are relative to a time zone.

This can be carried to what seems like a ridiculous extreme, but turns out to yield a very concise treatment. The Common Era (CE or AD) is also relative to a time zone because 2004 years ago it began three hours earlier in what is now the Eastern Standard time zone than in what is now the Pacific Standard time zone. What we think of as the Common Era is in fact 24 (actually more) slightly displaced half-infinite intervals. (We leave BCE to specialized ontologies.)

The principal functions and predicates will specify a clock or calendar unit interval to be the nth such unit in a larger interval. The time zone need not be specified in this predication if it is already built into the nature of the larger interval. That means that the time zone only needs to be specified in the largest interval, that is, the Common Era; that time zone will be inherited by all smaller intervals. Thus, the Common Era can be considered as a function from time zones to intervals.

$$CE(z) = T$$

Fortunately, this counterintuitive conceptualization will usually be invisible and, for example, in Section 4.5, will not be evident in the most useful expressions for time. In fact, the CE predication functions as a good place to hide considerations of time zone when they are not relevant.

We have been referring to time zones, but in fact it is more convenient to work in what we might call the "time standard" in a time zone. That is, it is better to work with *PST* as a legal entity than with the *PST* zone as a geographical region. A time standard is a way of computing the time relative to a world-wide system of computing time. For each time standard, there is a zone, or geographical region, and a time of the year in which it is used to describe local times. Where and when a time standard is used has to be axiomatized; this involves interrelating a time ontology and at least a simple geographical ontology. These relations can be quite complex. We have done this for the entire world; see Section 4.2.

If we were to conflate time zones (i.e., geographical regions) and time standards, it would likely result in problems in several situations. For example, the Eastern Standard zone and the Eastern Daylight zone are not identical, since most of Indiana is on Eastern Standard time all year. The state of Arizona and the Navajo Indian Reservation, two overlapping geopolitical regions, have different time standards – one is Pacific and one is Mountain.

Time standards that seem equivalent, like Eastern Standard and Central Daylight, should be thought of as separate entities. Whereas they function identically in the time ontology, they do not do so in the ontology that articulates time and geography. For example, it would be false to say that parts of Indiana shift in April from Eastern Standard to Central Daylight time.

4.2 Time Zone Data in OWL

We have developed a time zone resource in OWL, not only for the US but also for the entire world,[4] which includes three parts: the time ontology file, the US time zone instance file, and the world time zone instance file.

The time zone ontology links a simple geographic ontology with our time ontology. It defines a vocabulary about regions, political regions (countries, states, counties, reservations, and cities), time zones, daylight savings policies, and the relationships among these concepts. Its instances also link to other existing data on the Web, such as Terry Payne's US states instances,[5] FIPS 55 county instances,[6] and ISO country instances.[7]

It can handle all the usual time zone and daylight savings cases. For example, Los Angles uses PST, the time offset from Greenwich Mean Time (GMT) is -8 hours, and it observed daylight savings from April 4 to October 31 in 2004. But it handles unusual cases as well. For example, in Idaho the northern part is in the Pacific zone, the southern part in the Mountain. The city of West Wendover, Nevada is in the Mountain time zone, while the rest of Nevada is in the Pacific.

For the details, see the documentation,[8] which includes an outline of the ontology and examples of anticipated use.

[4] http://www.isi.edu/ pan/timezonehomepage.html
[5] http://www.daml.ri.cmu.edu/ont/USRegionState.daml
[6] http://www.daml.org/2003/02/fips55/
[7] http://www.daml.org/2001/09/countries/iso
[8] http://www.isi.edu/~pan/damltime/time-zone-documentation.txt

4.3 Clock and Calendar Units

The aim of this section is to explicate the various standard clock and calendar intervals. A day as a calender interval begins at and includes midnight, and goes until, but does not include, the next midnight. By contrast, a day as a duration is any interval that is 24 hours in length. The day as a duration was dealt with in Section 3; this section deals with the day as a calendar interval.

Including the beginning but not the end of a calendar interval in the interval may strike some as arbitrary. But we get a cleaner treatment if, for example, all times of the form 12:xx am, including 12:00 am, are part of the same hour and day, and all times of the form 10:15:xx, including 10:15:00, are part of the same minute.

It is useful to have three ways of saying the same thing: the clock or calendar interval y is the nth clock or calendar interval of type u in a larger interval x. For minutes, this can be expressed as follows:

$$min(y,n,x)$$

Under the reasonable assumption that there is only one such y, this can also be expressed as follows:

$$minFn(n,x) = y$$

For stating general properties about clock intervals, it is also useful to have the following way of expressing the same thing:

$$clockInt(y,n,u,x)$$

This expression says that y is the nth clock interval of type u in x. For example, the proposition *clockInt*(10:03,3,**Minute**,[10:00,11:00]) holds.

Here u is a member of the set of clock units, that is, one of *Second*, *Minute*, or *Hour*.

The larger interval x may not line up exactly with clock intervals. In this case we take y to be the nth *complete* clock interval of type u in x.

In addition, there is a calendar unit function with similar structure:

$$calInt(y,n,u,x)$$

This says that y is the nth calendar interval of type u in x. For example, the proposition *calInt*(12*Mar*2002,12,**Day**,*Mar*2002) holds. Here u is one of the calendar units *Day*, *Week*, *Month*, and *Year*.

The unit *DayOfWeek* is introduced in Section 4.4.

The relations among these modes of expression are as follows:

$$sec(y,n,x) \equiv clockInt(y,n,*Second*,x)$$
$$secFn(n,x) = y \equiv clockInt(y,n,*Second*,x)$$
$$min(y,n,x) \equiv clockInt(y,n,*Minute*,x)$$
$$minFn(n,x) = y \equiv clockInt(y,n,*Minute*,x)$$
$$hr(y,n,x) \equiv clockInt(y,n,*Hour*,x)$$
$$hrFn(n,x) = y \equiv clockInt(y,n,*Hour*,x)$$
$$da(y,n,x) \equiv calInt(y,n,*Day*,x)$$
$$daFn(n,x) = y \equiv calInt(y,n,*Day*,x)$$
$$mon(y,n,x) \equiv calInt(y,n,*Month*,x)$$
$$monFn(n,x) = y \equiv calInt(y,n,*Month*,x)$$
$$yr(y,n,x) \equiv calInt(y,n,*Year*,x)$$
$$yrFn(n,x) = y \equiv calInt(y,n,*Year*,x)$$

Weeks and months are dealt with separately, below.

The am/pm designation of hours is represented by the function *hr*12.

$$hr12(y,n,*am*,x) \equiv hr(y,n,x)$$

$$hr12(y,n,*pm*,x) \equiv hr(y,n+12,x)$$

A distinction is made above between clocks and calendars because they differ in how they number their unit intervals. The first minute of an hour is labelled with 0; for example, the first minute of the hour [10:00,11:00] is 10:00. The first day of a month is labelled 1; the first day of March is March 1. We number minutes for the number just completed; we number days for the day we are working on. Thus, if the larger unit has N smaller units, the argument n in *clockInt* runs from 0 to N-1; whereas, in *calInt, n* runs from 1 to N. To state properties true of both clock and calendar intervals, we can use the predicate *calInt* and relate the two notions with the axiom

$$calInt(y,n,u,x) \equiv clockInt(y,n\text{-}1,u,x)$$

Note that the Common Era is a calendar interval in this sense, since it begins with 1 CE and not 0 CE.

The type constraints on the arguments of *calInt* are as follows:

$$calInt(y,n,u,x) \supset Interval(y) \wedge integer(n) \wedge TemporalUnit(u) \wedge Interval(x)$$

Each of the calendar intervals is that unit long; for example, a calendar year is a year long.

$$calInt(y,n,u,x) \supset duration(y,u) = 1$$

There are properties relating to the labelling of clock and calendar intervals. If N *u*'s *hath* x and y is the nth *u* in x, then n is between 1 and N.

$$calInt(y,n,u,x) \wedge Hath(N,u,x) \supset 0 < n \leq N$$

The larger interval x need not line up with calendar intervals. For example, it might go from 12:36 pm, February 1, to 12:36 pm, February 3; this interval *Hath* two days, but not two calendar days. However, if x is itself a calendar interval, and u is a unit other than *Week*, then there is a 1st small interval, and it starts the large interval.

$$Hath(N,u,x) \wedge calInt(x,n_1,u_1,x_1) \wedge u \neq *Week* \supset (\exists\,!\,y)\, calInt(y,1,u,x)$$

$$Hath(N,u,x) \wedge calInt(x,n_1,u_1,x_1) \wedge u \neq *Week* \wedge calInt(y,1,u,x) \supset intStarts(y,x)$$

Under the same conditions, there is an Nth small interval, and it finishes the large interval.

$$Hath(N,u,x) \wedge calInt(x,n_1,u_1,x_1) \wedge u \neq *Week* \supset (\exists\,!\,y)\, calInt(y,N,u,x)]$$

$$Hath(N,u,x) \wedge calInt(x,n_1,u_1,x_1) \wedge u \neq *Week* \wedge calInt(y,N,u,x) \supset intFinishes(y,x)$$

Under these conditions, all but the last small interval have a small interval that succeeds and is met by it.

$$calInt(y_1,n,u,x) \wedge calInt(x,n_1,u_1,x_1) \wedge u \neq *Week* \wedge Hath(N,u,x) \wedge n < N \supset (\exists\,!\,y_2)[calInt(y_2,n+1,u,x) \wedge intMeets(y_1,y_2)]$$

Moreover, all but the first small interval have a small interval that precedes and meets it.

$$calInt(y_2,n,u,x) \wedge calInt(x,n_1,u_1,x_1) \wedge u \neq *Week* \wedge Hath(N,u,x) \wedge 1 < n \supset (\exists\,!\,y_1)[calInt(y_1,n\text{-}1,u,x) \wedge intMeets(y_1,y_2)]$$

4.4 Weeks

A week is any seven consecutive days. A calendar week, by contrast, according to a commonly adopted convention, starts at midnight, Saturday night, and goes to the next

midnight, Saturday night. That is, weeks start with Sunday. (By contrast, the ISO 8061 standard week starts with Monday.) There are 52 weeks in a year, but there are not usually 52 calendar weeks in a year.

Weeks are independent of months and years. However, we can still talk about the *n*th week in some larger period of time, e.g., the third week of the month or the fifth week of the semester.[9] So the same three modes of representation are appropriate for weeks as well.

$$wk(y,n,x) \equiv calInt(y,n,*Week*,x)$$
$$wkFn(n,x) = y \equiv calInt(y,n,*Week*,x)$$

As it happens, the *n* and *x* arguments will often be irrelevant when we only want to say that some period is a calendar week.

The day of the week is a calendar interval of type *Day*. The *n*th day-of-the-week in a week is the *n*th day in that interval.

$$dayofweek(y,n,x) \equiv da(y,n,x) \wedge (\exists n_1, x_1)\, wk(x,n_1,x_1)$$

The days of the week have special names in English.

$$dayofweek(y,1,x) \equiv Sunday(y,x)$$
$$dayofweek(y,2,x) \equiv Monday(y,x)$$
$$dayofweek(y,3,x) \equiv Tuesday(y,x)$$
$$dayofweek(y,4,x) \equiv Wednesday(y,x)$$
$$dayofweek(y,5,x) \equiv Thursday(y,x)$$
$$dayofweek(y,6,x) \equiv Friday(y,x)$$
$$dayofweek(y,7,x) \equiv Saturday(y,x)$$

For example, $Sunday(y,x)$ says that y is the Sunday of week x.

The ISO 8061 standard week is related to the traditional week as follows:

$$0 < n < 7 \supset [isodayofweek(y,n,x) \equiv dayofweek(y,n+1,x)]$$
$$isodayofweek(y,7,x) \equiv Sunday(y,x)$$

Since a day of the week is also a calendar day, it is a theorem that it is a day long.

$$dayofweek(y,n,x) \supset day(y)$$

One correspondence anchors the cycle of weeks to the rest of the calendar, for example, saying that January 1, 2002 was the Tuesday of some week x.

$$(\forall z)(\exists x)Tuesday(dayFn(1,monFn(1,yrFn(2002,CE(z)))),x)$$

We can define weekdays and weekend days as follows:

$$weekday(y,x) \equiv [Monday(y,x) \vee Tuesday(y,x) \vee Wednesday(y,x) \vee Thursday(y,x) \vee Friday(y,x)]$$
$$weekendday(y,x) \equiv [Saturday(y,x) \vee Sunday(y,x)]$$

4.5 Months and Years

The months have special names in English. In these rules we specify that the larger interval is a calendar year.

$$[yr(x,n_1,x_1) \wedge mon(y,1,x)] \equiv January(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,2,x)] \equiv February(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,3,x)] \equiv March(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,4,x)] \equiv April(y,x)$$

[9]This may not accord perfectly with how we talk about such things, since we really mean the *n*th *complete* week in *x*.

$$[yr(x,n_1,x_1) \wedge mon(y,5,x)] \equiv May(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,6,x)] \equiv June(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,7,x)] \equiv July(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,8,x)] \equiv August(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,9,x)] \equiv September(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,10,x)] \equiv October(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,11,x)] \equiv November(y,x)$$
$$[yr(x,n_1,x_1) \wedge mon(y,12,x)] \equiv December(y,x)$$

The number of days in a month have to be spelled out for individual months.

$$January(m,y) \supset Hath(31,*Day*,m)$$
$$March(m,y) \supset Hath(31,*Day*,m)$$
$$April(m,y) \supset Hath(30,*Day*,m)$$
$$May(m,y) \supset Hath(31,*Day*,m)$$
$$June(m,y) \supset Hath(30,*Day*,m)$$
$$July(m,y) \supset Hath(31,*Day*,m)$$
$$August(m,y) \supset Hath(31,*Day*,m)$$
$$September(m,y) \supset Hath(30,*Day*,m)$$
$$October(m,y) \supset Hath(31,*Day*,m)$$
$$November(m,y) \supset Hath(30,*Day*,m)$$
$$December(m,y) \supset Hath(31,*Day*,m)$$

The definition of a leap year is as follows:

$$(\forall z)[leapYear(y) \equiv (\exists n,x)[yr(y,n,CE(z)) \wedge [divides(400,n) \vee [divides(4,n) \wedge \neg divides(100,n)]]]$$

We leave leap seconds to specialized ontologies.

Now the number of days in February can be specified.

$$February(m,y) \wedge leapYear(y) \supset Hath(29,*Day*,m)$$
$$February(m,y) \wedge \neg leapYear(y) \supset Hath(28,*Day*,m)$$

A reasonable approach to defining month as a unit of temporal measure would be to specify that the start and end points have to be on the same days of successive months. The following rather ugly axiom captures this.

$$\begin{aligned} &month(T) \\ &\equiv (\exists t_1,t_2,d_1,d_2,n_1,n_2,n_3,n_4,m_1,m_2,y_1,y_2,e,h_1,h_2,j_1,j_2,s_1,s_2) \\ &\quad [begins(t_1,T) \wedge ends(t_2,T) \wedge beginsOrIn(t_1,d_1) \\ &\quad \wedge beginsOrIn(t_2,d_2) \wedge da(d_1,n_1,m_1) \wedge mon(m_1,n_3,y_1) \\ &\quad \wedge yr(y_1,n_4,e) \wedge da(d_2,n_2,m_2) \\ &\quad \wedge [mon(m_2,n_3+1,y_1) \\ &\qquad \vee (\exists y_2)[n_1 = 12 \wedge mon(m_2,1,y_2) \wedge yr(y_2,n_4+1,e)]] \\ &\quad \wedge Hath(n,*Day*,m_2) \wedge [[n \geq n_1 \wedge n_2 = n_1] \\ &\qquad \vee [n < n_1 \wedge n_2 = n]] \end{aligned}$$

$$\wedge\ beginsOrIn(t_1,h_1)\ \wedge\ beginsOrIn(t_2,h_2)$$
$$\wedge\ (\exists\ i)[hr(h_1,i,d_1)\ \wedge\ hr(h_2,i,d_2)]$$
$$\wedge\ beginsOrIn(t_1,j_1)\ \wedge\ beginsOrIn(t_2,j_2)$$
$$\wedge\ (\exists\ i)[min(j_1,i,h_1)\ \wedge\ min(j_2,i,h_2)]$$
$$\wedge\ beginsOrIn(t_1,s_1)\ \wedge\ beginsOrIn(t_2,s_2)$$
$$\wedge\ (\exists\ i)[sec(s_1,i,j_1)\ \wedge\ sec(s_2,i,j_2)]]$$

The first eight conjuncts in the consequent identify and label the days the interval begins and ends in. The ninth conjunct checks that the months are right, taking care of the rollover from December to January. The tenth and eleventh conjuncts make sure the days are right, taking care of the case where the second month has too few days. In this definition, one month from 2:36 pm, January 31, 2004, is 2:36 pm, February 29, 2004. The last nine conjuncts make sure the hours, minutes, and seconds are right. This definition does not handle precisions less than a second. It would be a technical exercise to do so.

Thus, the month as a measure of duration is related to days as a measure of duration only indirectly, mediated by the calendar. It is possible to prove that months are between 28 and 31 days.

The mismatch between days and months in the calendar entails significant difficulties in working out a satisfactory temporal arithmetic. We will deal with this problem in a subsequent paper.

To say that July 4 is a holiday in the United States, we could write

$$(\forall\ d,m,y)[da(d,4,m)\ \wedge\ July(m,y) \supset holiday(d,USA)]$$

Holidays like Easter can be defined in terms of this ontology coupled with an ontology of the phases of the moon.

Other calendar systems could be axiomatized similarly, and the BCE era could also be axiomatized in this framework; these are left as exercises for interested developers.

5. DESCRIBING TIMES AND DURATIONS

5.1 Timestamps

Standard notation for times list the year, month, day, hour, minute, second, and time zone. It is useful to define a predication for this.

$$timeOf(t,y,m,d,h,n,s,z)$$
$$\equiv beginsOrIn(t,secFn(s,minFn(n,hrFn(h,$$
$$daFn(d,monFn(m,yrFn(y,CE(z)))))))$$

Alternatively,

$$timeOf(t,y,m,d,h,n,s,z)$$
$$\equiv (\exists\ s_1,n_1,h_1,d_1,m_1,y_1,e)$$
$$[beginsOrIn(t,s_1)\ \wedge\ sec(s_1,s,n_1)\ \wedge\ min(n_1,n,h_1)$$
$$\wedge\ hr(h_1,h,d_1)\ \wedge\ da(d_1,d,m_1)\ \wedge\ mon(m_1,m,y_1)$$
$$\wedge\ yr(y_1,y,e)\ \wedge\ CE(z) = e]$$

For example, an instant t has the time

5:14:35pm PST, Wednesday, February 6, 2002

if the following properties hold for *t*:

$$timeOf(t,2002,2,6,17,14,35,*PST*)$$
$$\wedge\ (\exists\ w,x)[beginsOrIn(t,w)\ \wedge\ Wednesday(w,x)]$$

The second line says that *t* is in the Wednesday *w* of some week *x*.

The relations among time zones can be expressed in terms of the *timeOf* predicate. Two examples follow:

$$timeOf(t,y,m,d,h,n,s,*EST*) \equiv timeOf(t,y,m,d,h,n,s,*CDT*)$$
$$timeOf(t,y,m,d,h,n,s,*GMT*)\ \wedge\ hours(T) = 8\ \wedge\ ends(t,T)$$
$$\wedge\ begins(t_1,T)\ \wedge\ timeOf(t_1,y_1,m_1,d_1,h_1,n,s,*GMT*)$$
$$\supset timeOf(t,y_1,m_1,d_1,h_1,n,s,*PST*)$$

In the second rule, subtracting an interval of 8 hours from the time *t* and looking at the time of its beginning point hides the ugly details of computing the years, months, and days in case of rollover. For those who prefer to see the ugly details, here they are:

$$(\forall\ t,y,m,d,h,n,s)[h \geq 8$$
$$\supset [timeOf(t,y,m,d,h,s,*GMT*) \equiv timeOf(t,y,m,d,h\text{-}8,s,*PST*)$$
$$(\forall\ t,y,m,d,h,n,s)[h < 8\ \wedge\ d > 1$$
$$\supset [timeOf(t,y,m,d,h,s,*GMT*) \equiv timeOf(t,y,m,d\text{-}1,h\text{+}16,s,*PST*)$$
$$(\forall\ t,y,m,d,h,n,s,M,d_1,Y)[h < 8\ \wedge\ d = 1\ \wedge\ m > 1\ \wedge\ mon(M,m,Y)$$
$$\wedge\ yr(Y,y,CE(*GMT*)\ \wedge\ Hath(d_1,*Day*,M)$$
$$\supset [timeOf(t,y,m,d,h,s,*GMT*) \equiv timeOf(t,y,m\text{-}1,d_1,h\text{+}16,s,*PST*)$$
$$(\forall\ t,y,m,d,h,n,s)[h < 8$$
$$\supset [timeOf(t,y,1,1,h,s,*GMT*) \equiv timeOf(t,y\text{-}1,12,31,h\text{+}16,s,*PST*)$$

5.2 Calendar-Clock Descriptions

It is inconvenient to express *calInt*(*y*,*n*,*u*,*x*) and *clockInt*(*y*,*n*,*u*,*x*) directly in a description logic-based markup language, such as OWL, since *x* is itself a clock or calendar interval that requires description. So we have defined a calendar-clock or time description in OWL for specifying both calendar and clock information for a calendar-clock interval.

A calendar-clock description has the following properties or fields: unitType, yearOf, monthOf, weekOf, dayOf, hourOf, minuteOf, secondOf, and timeZoneOf. The property unitType specifies the temporal unit type of the calendar-clock description, and its domain is the set of temporal units.

For example, the unit type of 10:30 is *minute*, and the unit type of March 20, 2003 is *day*. The unit type is required. For a given temporal unit type, all the fields or properties for smaller units will be ignored; for instance, if the temporal unit type is *day*, the values of the fields or properties hourOf, minuteOf, and secondOf, if present, will be ignored.

Since calendar-clock descriptions are for describing calendar-clock intervals, we have defined a property or relation, called *calendarClockDescriptionOf* with *CalendarClockDescription* as the range.

To express *calInt*(12*Mar*2002,12,**Day**,*Mar*2002), for example, using a calendar-clock description, we need an instance of *CalendarClockDescription* that has values only for unitType (*day*), yearOf (2002), monthOf (3), and dayOf (12). *clockInt*(10:03,3,**Minute**,[10:00, 11:00]) can be expressed similarly.

More details about calendar-clock descriptions, as well as duration descriptions, together with examples used in OWL-S[10] can be found in Pan and Hobbs [2004].

[10] http://www.daml.org/services/owl-s/

5.3 Duration Descriptions

There are two systems of time that are based on different astronomical facts. The year-month system is based on the revolution of the earth around the sun. The week-day-hour-minute-second system is based on the rotation of the earth around its axis. As long as we don't mix these two systems, temporal arithmetic is simple. But they don't align well, and when we try to relate days and months, complications arise, as we have already seen.

We cannot simply rule out months as units, as some have suggested. Monthly rates play a very important role in commerce. If you pay $1000 a month in rent, you are paying more per day for your apartment in February than in March, and often when rents are prorated, the number of days in that specific month is used in the calculation, although in some industries months have been normalized to 28 or 30 days.

Hence it is important to build a consistent system of duration measurement that involves both months and days.

Here we introduce duration descriptions, in which the duration of an arbitrary finite interval can be described as a concatenation of years, months, weeks, days, hours, minutes, seconds, and fractions of seconds. The primary convention we follow is that used by car rental and other companies that have different rates for different periods of time. From the beginning of the interval, we fit in as many of the largest unit type as possible. Then, into the remainder, we fit in as many as possible of the next largest unit type, and so on. For example, when we rent a car, we pay the weekly rate for as many full weeks as we keep the car, then we pay the daily rate for any leftover full days, then the hourly rate for any leftover hours.

The predication $durationOf(T,y,m,w,d,h,n,s)$ says that duration of the interval T is y years, m months, w weeks, d days, h hours, n minutes, and s seconds. The values of the numeric arguments can be any real number, although indeterminacies will arise if we try to determine the identity of a duration described as a fractional number of months and a duration described in terms of days. We allow real numbers, rather than restricting the values to integers, because we frequently talk about such durations as 1 1/2 months. However, for the rest of this development we assume that all of the numeric arguments are integers.

The predicate *durationOf* can be defined in the following rather cumbersome manner:

$$
\begin{aligned}
&durationOf(T,y,m,w,d,h,n,s)\\
&\quad \equiv (\exists\, S,T_1)[concatenation(T,S \cup \{T_1\}) \wedge card(S) = y\\
&\qquad \wedge\ (\forall\, v)[v \in S \supset year(v)]\\
&\qquad \wedge\ intFinishes(T_1,T) \wedge durationOf(T_1,0,m,w,d,h,n,s)]\\
&durationOf(T,0,m,w,d,h,n,s)\\
&\quad \equiv (\exists\, S,T_1)[concatenation(T,S \cup \{T_1\}) \wedge card(S) = m\\
&\qquad \wedge\ (\forall\, v)[v \in S \supset month(v)]\\
&\qquad \wedge\ intFinishes(T_1,T) \wedge durationOf(T_1,0,0,w,d,h,n,s)]\\
&durationOf(T,0,0,w,d,h,n,s)\\
&\quad \equiv (\exists\, S,T_1)[concatenation(T,S \cup \{T_1\}) \wedge card(S) = w\\
&\qquad \wedge\ (\forall\, v)[v \in S \supset week(v)]\\
&\qquad \wedge\ intFinishes(T_1,T) \wedge durationOf(T_1,0,0,0,d,h,n,s)]
\end{aligned}
$$

$$\begin{aligned}
&\mathit{durationOf}(T,0,0,0,d,h,n,s)\\
&\quad\equiv (\exists\, S,T_1)[\mathit{concatenation}(T,S\cup\{T_1\}) \wedge \mathit{card}(S)=d\\
&\qquad\wedge\ (\forall\, v)[v\in S\supset \mathit{day}(v)]\\
&\qquad\wedge\ \mathit{intFinishes}(T_1,T) \wedge \mathit{durationOf}(T_1,0,0,0,0,h,n,s)]\\
&\mathit{durationOf}(T,0,0,0,0,h,n,s)\\
&\quad\equiv (\exists\, S,T_1)[\mathit{concatenation}(T,S\cup\{T_1\}) \wedge \mathit{card}(S)=h\\
&\qquad\wedge\ (\forall\, v)[v\in S\supset \mathit{hour}(v)]\\
&\qquad\wedge\ \mathit{intFinishes}(T_1,T) \wedge \mathit{durationOf}(T_1,0,0,0,0,0,n,s)]\\
&\mathit{durationOf}(T,0,0,0,0,0,n,s)\\
&\quad\equiv (\exists\, S,T_1)[\mathit{concatenation}(T,S\cup\{T_1\}) \wedge \mathit{card}(S)=n\\
&\qquad\wedge\ (\forall\, v)[v\in S\supset \mathit{minute}(v)]\\
&\qquad\wedge\ \mathit{intFinishes}(T_1,T) \wedge \mathit{durationOf}(T_1,0,0,0,0,0,0,s)]\\
&\mathit{durationOf}(T,0,0,0,0,0,0,s)\\
&\quad\equiv (\exists\, S,T_1)[\mathit{concatenation}(T,S) \wedge \mathit{card}(S)=s\\
&\qquad\wedge\ (\forall\, v)[v\in S\supset \mathit{second}(v)]]
\end{aligned}$$

The axiom saying that an instant has 0 duration is

$$\mathit{Instant}(t)\supset \mathit{durationOf}(t,0,0,0,0,0,0,0)$$

The predicates *timeOf* and *durationOf* can be related. Corresponding to every time is the duration of the interval from the beginning of the Common Era to that time.

$$\begin{aligned}
&\mathit{timeOf}(t_0,1,1,1,0,0,0) \wedge \mathit{timeBetween}(T,t_0,t)\\
&\quad\supset (\forall\, y,m,d,h,n,s)[\mathit{timeOf}(t,y,m,d,h,n,s)\\
&\qquad\equiv \mathit{durationOf}(T,y\text{-}1,m\text{-}1,d\text{-}1,h,n,s)]
\end{aligned}$$

The duration of an interval can have many different descriptions. An interval can be 1 day 2 hours, or 26 hours, or 1560 minutes, and so on. It is useful to be able to talk about these descriptions in a convenient way as independent objects and to talk about their equivalences. Thus, we define a specific kind of individual called a "duration description," together with a number of functions relating the duration description to the values of each of the eight arguments of *durationOf*. So we convert the 8-ary predicate *durationOf* into eight binary relations that are more convenient for description logic-based markup languages like OWL. Here is the definition of the duration description:

$$\begin{aligned}
&(\forall\, T,y,m,w,d,h,n,s)[\mathit{durationOf}(T,y,m,w,d,h,n,s)\\
&\quad\equiv (\exists\, D)[\mathit{durationDescriptionOf}(D,T) \wedge \mathit{DurationDescription}(D)\\
&\qquad\wedge\ \mathit{yearsOf}(D)=\mathrm{y} \wedge \mathit{monthsOf}(D)=m\\
&\qquad\wedge\ \mathit{weeksOf}(D)=w \wedge \mathit{daysOf}(D)=d\\
&\qquad\wedge\ \mathit{hoursOf}(D)=h \wedge \mathit{minutesOf}(D)=n \wedge \mathit{secondsOf}(D)=s]]
\end{aligned}$$

We say that a duration description is canonical if the number of weeks is zero and the number of all other units is less than the number of those units in the next higher unit. That is, there is an arbitrarily large number of years, less than 12 months, less than 24 hours, less than 60 minutes, and less than 60 seconds. The number of days is less than the number that could be consumed by one more month, given where the interval is anchored in time.

The definition of *canonicalDurDescr*(*D*) is as follows:

$$
\begin{aligned}
&\textit{canonicalDurDescr}(D)\\
&\quad \equiv [0 \le \textit{monthsOf}(D) < 12 \ \wedge \ \textit{weeksOf}(D) = 0\\
&\qquad \wedge \ 0 \le \textit{hoursOf}(D) < 24 \ \wedge \ 0 \le \textit{minutesOf}(D) < 60\\
&\qquad \wedge \ 0 \le \textit{secondsOf}(D) < 60\\
&\qquad \wedge \ (\exists\, T, T_1, T_2, t, t_1, t_2)\\
&\qquad\quad [\textit{durationOf}(T_1, \textit{yearsOf}(D), \textit{monthsOf}(D), 0,0,0,0,0)\\
&\qquad\quad \wedge \ \textit{durationDescriptionOf}(D,T) \ \wedge \ \textit{begins}(t_1, T)\\
&\qquad\quad \wedge \ \textit{begins}(t_1, T_1) \ \wedge \ \textit{month}(T_2) \ \wedge \ \textit{intMeets}(T_1, T_2)\\
&\qquad\quad \wedge \ \textit{ends}(t_2, T_2) \ \wedge \ \textit{ends}(t,T) \ \wedge \ \textit{before}(t, t_2)]]
\end{aligned}
$$

The existentially quantified expression at the end requires explanation. T is the interval that D describes. T_1 is the interval starting at the same point and including only D's year and month segments. T_2 is a month-long interval that is appended to the end of T_1. The *daysOf* slot of D is canonical if and only if T ends before T_2 does. The complexities of day-month arithmetic are hidden in the predicate *month*.

6. FUTURE DIRECTIONS

6.1 Temporal Arithmetic

As long as we stay within the year-month system or the week-day-hour-minute-second system, temporal arithmetic is just arithmetic and requires only a few simple axioms to encode. When we mix months and days, problems arise.

We are currently working on a set of relatively simple rules that will allow us to do temporal arithmetic with months and days with a moderate degree of consistency. (This will be the subject of a future paper.) However, just to give the reader a flavor of the problems, consider that January 31, 2003, plus 2 months equals March 31, 2003. But if we add the months one at a time, we get a different result. January 31, 2003, plus one month is February 28, 2003. February 28, 2003, plus one month would seem to be March 28, 2003. If we want to avoid results like this, we need, in some sense, to keep track of the history of the computation.

6.2 Deictic Time

Deictic temporal concepts such as "now," "today," "tomorrow night," and "last year" are more common in natural language texts than they will be in descriptions of Web resources, hence we have postponed development of this domain until the first three are in place. But since most of the content on the Web is in natural language, it will ultimately be necessary for this ontology to be developed. It should, as well, mesh well with the annotation standards used in automatic tagging of text (cf., Hobbs and Pustejovsky [2003]).

We expect that the key concept in this area will be a relation *now* between an instant and an utterance or document.

now(*t*,*d*)

It may refer to the time of writing, the time of reading, a period of validity, or some other functionally determined instant or interval.

The concept of “today" would also be relative to a document, and be defined as follows:

$$today(T,d) \equiv (\exists\, t,n,x)[now(t,d) \wedge inside(t,T) \wedge da(T,n,x)]$$

That is, T is today with respect to document d if and only if there is an instant t in T that is now with respect to the document and T is a calendar day (and thus the nth calendar day in some interval x).

Present, past, and future can be defined in the obvious way in terms of *now* and *before*.

Another feature of a treatment of deictic time is an axiomatization of the concepts of “last," “this," and “next" on anchored sequences of temporal entities.

6.3 Aggregates of Temporal Entities

A number of common expressions and commonly used properties are properties of sequences of temporal entities. These properties may be properties of all the elements in the sequence, as in “every Wednesday” or they may be properties of parts of the sequence, as in “three times a week” or “an average of once a year”. We have also postponed development of this domain until the first three domains are well in hand.

6.4 Vague Temporal Concepts

In natural language, a very important class of temporal expressions are inherently vague. Included in this category are such terms as “soon," “recently," “late," and “a little while." These require an underlying theory of vagueness, and in any case are probably not immediately critical for the Semantic Web. (This area will be postponed for a little while).

ACKNOWLEDGMENTS

We have profited from discussions with James Allen, George Ferguson, Pat Hayes, Inderjeet Mani, Drew McDermott, Adam Pease, James Pustejovsky, Stephen Reed, and Austin Tate, among others, none of whom, however, would necessarily agree entirely with the way we have characterized the effort.

REFERENCES

ALLEN, J.F. AND FERGUSON, G. 1997. Actions and events in interval temporal logic. In *Spatial and Temporal Reasoning*. O. Stock, ed., Kluwer, Dordrecht, Netherlands, 205-245.

ALLEN, J.F. AND KAUTZ, H.A. 1985. A model of naive temporal reasoning. In *Formal Theories of the Commonsense World*. J.R. Hobbs and R.C. Moore, eds., Ablex., 251-268.

HOBBS, J.R. AND PUSTEJOVSKY, J. 2003. Annotating and reasoning about time and events. In *Proceedings of AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning* (Stanford, CA, March 2003).

PAN, F. AND HOBBS, J.R. 2004. Time in OWL-S. In *Proceedings of AAAI Spring Symposium on Semantic Web Services* (Stanford University, CA, March 2004).

Received March 2004; revised June 2004; accepted June 2004