
Designing Annotation Schemes: From Theory to Model

James Pustejovsky, Harry Bunt and Annie Zaenen

Abstract

In this chapter, we describe the method and process of transforming the theoretical formulations of a linguistic phenomenon, based on empirical observations, into a model that can be used for the development of a language annotation specification. We outline this procedure generally, and then examine the steps in detail by specific example. We look at how this methodology has been implemented in the creation of TimeML (and ISO-TimeML), a broad-based standard for annotating temporal information in natural language texts. Because of the scope of this effort and the richness of the theoretical work in the area, the development of TimeML illustrates very clearly the methodology of the early stages of the MATTER annotation cycle, where initial models and schemas cycle through progressively mature versions of the resulting specification. Furthermore, the subsequent effort to convert TimeML into an ISO compliant standard, ISO-TimeML, demonstrates the utility of the CASCADES model in distinguishing between the concrete syntax of the schema and abstract syntax of the model behind it.

J. Pustejovsky (✉)

Department of Computer Science, Brandeis University, Waltham, MA 02453, USA
e-mail: jamesp@cs.brandeis.edu

H. Bunt

TiCC, Tilburg Center for Cognition and Communication Tilburg University,
Tilburg, The Netherlands
e-mail: harry.bunt@uvt.nl

A. Zaenen

CSLI, Stanford University, Palo Alto, CA, USA
e-mail: azaenen@earthlink.net

Keywords

Annotation methodology · MATTER cycle · CASCADES · TimeML · ISO-TimeML · Specification design · Schemas · Models · Standards

1 Introduction

In a language annotation task, we typically take it for granted that the specification being followed for markup of the text is appropriate to the domain generally, and to the task specifically. If designed carefully, the specification corresponds to an abstract data model of the linguistic phenomena being studied. Where such a model comes from, however, and how it is developed, is often not documented or revealed, and this process can remain obscure to those who adopt the specification for their use. In this chapter, we examine the methodology involved in creating such an annotation model. As the chapter title suggests, this involves the transformation of a theory (or of multiple theories) of the phenomena into a coherent model, from which a specification can be designed and subsequently implemented for linguistic annotation. We will illustrate this process with a specific example, namely that of designing a model for general temporal awareness as expressed in language, including temporal and event expressions, and how they are related to each other. By studying the history of the conceptual development of a specific annotation model and specification language, ISO-TimeML, we hope to illustrate the method required for adequately capturing “theory” in a model, as well as the interdependencies between expressiveness and transparency of a data model.

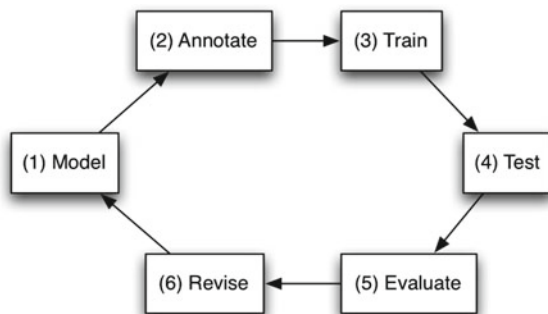
2 Annotation Methodology

In this section, we review the methodology adopted for arriving at a linguistically annotated corpus for use in training computational linguistic algorithms. This is best viewed as an annotation development cycle, and as such, we examine two models that are relevant to our concerns here: the MATTER cycle [60] and the CASCADES model [10]. The MATTER cycle outlines the methodology that is followed broadly when a researcher is interested in developing an algorithm to process natural language input with respect to a particular linguistic phenomenon or set of phenomena. The CASCADES model focuses on how the abstract syntax supports, on the one hand, the creation of the concrete specification that the researcher uses for annotation and, on the other, the formal semantics of the annotations that licenses their use in reasoning.

2.1 The MATTER Cycle

The goal of an annotation project is to ensure that the features that one adopts for encoding a specific phenomenon are rich enough to subsequently train machine

Fig. 1 The MATTER methodology

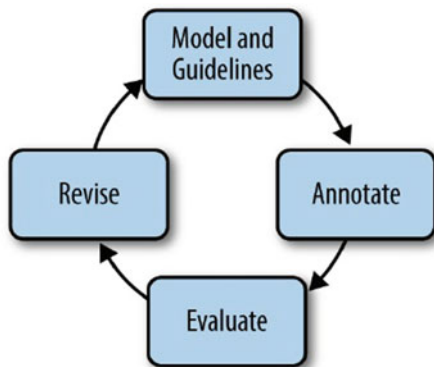


learning algorithms for automatic classification in the service of a given task. Linguistic descriptions are distilled from extensive theoretical modeling of a phenomenon, but in real life tasks we typically do not want to model a specific linguistic phenomenon on its own. The tasks often require us to consider the interaction between several phenomena that might be considered completely independent, from a theoretical perspective. Thus, a “theoretically informed” annotation model is rarely a linguistically pure one. The descriptions in turn form the basis for the annotation values of the specification language, which are themselves the features used in a development cycle for training and testing an identification or labeling algorithm over text (Fig. 1). Finally, based on an analysis and evaluation of the performance of a system, the model of the phenomenon may be revised, for retraining and testing. This particular cycle of development has been called the MATTER methodology, and consists of the following steps [60]:

- (1) a. **Model:** Structural descriptions provide theoretically-informed attributes derived from empirical observations over the data;
- b. **Annotate:** Annotation scheme assumes a feature set that encodes specific structural descriptions and properties of the input data;
- c. **Train:** Algorithm is trained over a corpus annotated with the target feature set;
- d. **Test:** Algorithm is tested against held-out data;
- e. **Evaluate:** Standardized evaluation of results;
- f. **Revise:** Revisit the model, annotation specification, or algorithm, in order to make the annotation more robust and reliable.

The main focus of this chapter is on the first component of this cycle, i.e., modeling the phenomenon from language data and established theoretical observations about them. A model can be seen as an abstract characterization of a phenomenon in terms that allow us to study the structural and expressive properties of the domain. We will define this initial model, M , as consisting of a vocabulary of terms, T , the relations between these terms, R , and their interpretation, I : $M = \langle T, R, I \rangle$.

Fig. 2 The MAMA sub-cycle



Creating an appropriate model can be a daunting task, particularly when the scope of the phenomenon is as broadly encompassing as accounting for how time and events should be annotated. For this reason, model building typically involves multiple iterations of attempts at generalizing the phenomena to a concise language for annotation, before an adequately expressive model fragment is arrived at. That is, a model is first proposed, it is then used for annotation over a small sample set of data, evaluated, and then revised. Within the MATTER cycle, the model revision process is referred to as the MAMA (Model-Annotate-Model-Annotate) cycle, or the “babbling” phase, as illustrated in Fig. 2.

Let us consider briefly how this process unfolds. Given a source document for markup, assume we have agreed upon an initial inventory of elements. For every target term, T , and relation, R , in the initial model, we then need to identify the strategies for marking the appropriate textual components in this document. Consider, for example, the sentence in (2).

(2) We visited the Eiffel Tower July 4, 2015.

The expression *July 4, 2015* will be tagged as a time, the verb *visited* will be tagged as an event, and a relation of temporal inclusion will link the event to the time. Both the temporal expression and the event are explicit textual extents, called “markables” or *consuming tags*. The temporal relation between them, however, is not explicitly associated with any word or phrase in the sentence, and is therefore sometimes referred to as a *nonconsuming tag*. Such informal strategies for how words or phrases are associated with the terms and relations of the model constitute the basis of an *annotation guideline*. In subsequent sections, we illustrate in some detail how model revision using MAMA can tighten and refine the vocabulary being adopted for the specification and the guideline accompanying the specification, as designed for a particular task.

2.2 The CASCADES Model

The CASCADES model describes a process related to the MATTER cycle, where the focus is on the internal structure of the Model and Revise steps in the MATTER cycle, in particular on the relations between the abstract data model of an annotation scheme, the abstract syntax with its semantic interpretation and the concrete syntax specifying annotation representations [8, 10, 11].

An *annotation language* serves to represent the information that annotations add to primary data. The CASCADES approach to designing annotation languages consists of the following stages:

- (3) 1: **Conceptual Analysis:** Formulate a conceptual view of the information to be captured in annotations. This results in an abstract data model or ‘metamodel’.
- 2: **Abstract Syntax:** Articulate the conceptual view in the form of an inventory of basic concepts and a formal specification of the possible ways of combining these elements in set-theoretical structures like pairs and triples, called *annotation structures*.
- 3: **Semantics:** Provide a formal semantics for the structures defined by the abstract syntax.
- 4: **Concrete Syntax:** Specify a representation format for the annotation structures defined by the abstract syntax;

The first of these steps, **Conceptual analysis**, serves to determine the conceptual content of the targeted annotations, identifying the basic concepts that form the building blocks of the annotations, and indicating the ways in which these concepts are interrelated. This early stage of designing an annotation language results in the establishment of what in ISO projects is called a *metamodel*, a diagrammatic representation of the kinds of elements that may occur in annotations and how they are related. An example can be found in Fig. 5 which shows the metamodel for annotating time and events according to ISO-TimeML.

The second stage, **Abstract Syntax Specification**, provides a formal specification of the concepts in the conceptual analysis and of the well-formed combinations of such concepts into set-theoretical structures like pairs and triples, called *annotation structures*.

The specification of what an annotation structure means is the specification of a **semantics** for these structures. This is the crucial stage 3 of the method. Any representation of an annotation structure inherits its semantics from the annotation structure that it represents. Defining the semantics of annotation representations in this indirect way, via the represented annotation structures, has the great advantage that any format for representing annotation structures inherits *the same* semantics, which is highly beneficial for improving the interoperability of semantic annotations (Fig. 3).¹

¹See [6,7] for the formal semantics of abstract annotation structures.

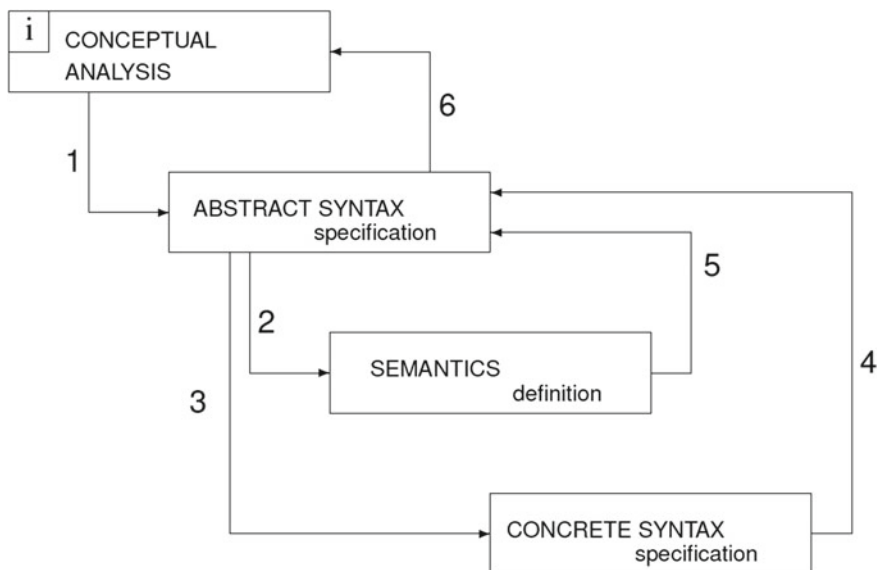


Fig. 3 The CASCADES model

The final stage of the CASCADES method is the definition of a reference format for representing the annotation structures defined by the abstract syntax, for example serialized in XML. A **concrete syntax** defines a representation format for a given abstract syntax. Such a representation format is required to have the properties of being *complete*, i.e. every annotation structure defined by the abstract syntax can be represented, and to be *unambiguous*, i.e., every expression defined by the concrete syntax represents only one annotation structure. A representation format that has these two properties is called *ideal*. It is easily shown that the representations of two ideal representation formats for a given abstract syntax can be converted from one format to the other in a meaning-preserving way [10].

Like the MATTER cycle, the CASCADES model has internal feedback loops, consisting of the steps indicated in Fig. 3 by the upward arrows numbered 4–6. In particular, the feedback loop $\langle 4, (\langle 2, 6 \rangle)^*, 3 \rangle$ is useful for the reverse-engineering of an annotation language starting from a concrete representation format, as in the case of defining ISO-TimeML starting from TimeML.

In the next section, we begin describing the way in which the model and specification for TimeML was built. Using the two methodologies mentioned above, the initial TimeML working group started with determining the appropriate scope of the specification, as determined by what applications and tasks were being targeted for development. This then lead to a more mature understanding of how the scope can be balanced by the actual effectiveness and reproducibility of annotation, given a specification design.

3 Scoping the Phenomena

As natural language understanding systems move beyond keywords and simple named entity extraction, the interpretation of temporal and event expressions in language is recognized as forming a critical component of such systems. This includes the identification of events along with their participants; the temporal anchoring and grounding of these events; and the ordering and structuring of these events into timelines and narratives for temporal reasoning and understanding. Since event recognition drives basic inferences from text, these are all interrelated and interdependent phenomena.

As it happens, however, most of the temporal information in an article, narrative, or discourse is actually left implicit. The exact temporal designation of events is rarely explicit and many temporal expressions are vague at best. A crucial first step in the automatic extraction of information from such texts, for use in applications such as question answering, machine translation, or summarization, is the ability to identify what events are being described and to make explicit when these events occurred. While remarkable progress has been made in the last decade in the use of statistical techniques for analyzing text, these techniques for the most part depend on having large amounts of annotated data, and annotations require an annotation scheme and a model. The scheme and model, in turn, come from a translation and distillation of theoretical observations about the phenomena being annotated.

Until fairly recently, there was, however, no broad-based or systematic specification language for annotating time and events within the computational linguistics community. There are several reasons for this and they relate to the main theme of this chapter; namely, how difficult it can be to build an annotation model from a theory. The first temporal feature that was identified for annotation was that associated with *temporal referring expressions*, such as *times*, *durations*, and *frequencies*. This includes expressions such as *June 11, 1989*, *Monday*, *two years*, *daily*, and so on. More complex constructions, such as *two days before yesterday*, as well as recursive structures, such as *several days during the winter* or *the first Saturday in every month*, were not typically modeled, mainly because they were low frequency expressions or they were out of the scope of the finite-state patterns used for identification [70]. The recognition of temporal expressions was initially done to support the time-stamping of domain-specific events of interest [51], typically targeted events in news articles, rather than any more general notion of eventuality. As such, the more general task of identifying those elements in language that contribute to the global temporal awareness of a text were not recognized as forming a unified computational problem.

The initial work on anchoring events to times was a step in the right direction, but to fully appreciate the complexity of a text with respect to time, the ability to identify and then temporally order events and time expressions is required. Soon it became clear that annotation efforts focused on time-stamping were not broad or expressive enough to handle the related linguistic phenomena of event semantics, tense, aspect, and temporal relations. In other words, the annotation language, and along with it, the data model supporting the specification, was too restrictive.

The model needed enriching in order to support a richer specification. This in turn required deeper theoretical foundations from event semantics, logic, and linguistics. In the discussion below, we outline the steps that are involved in identifying the theoretical underpinnings that make up a model for annotation, in this case, the language of TimeML.

From the outset, it was the goal of TimeML to represent a broad coverage of temporal information. This ruled out separate treatments for the connected phenomena, as well as shallow solutions for picking out entities associated with these different problems. Hence, simple named entity grammars or patterns for time expressions and verbal clusters denoting events would not be sufficient. Rather, the strategy was to carefully examine the linguistic phenomena that are implicated when reasoning about events and their temporal properties. The TimeML working group started with the following goals [61]:

- (4) a. to examine how to formally recognize events and their temporal anchoring in text (news articles); and
- b. to develop and evaluate algorithms for identifying and extracting events and temporal expressions from texts.

As the work progressed, it became clear that the specification would have to take into account the goals that are linked to the possible applications. That is, the results would be used not just for finding events, but also for the following tasks:

- (5) a. Order events with respect to each other (relating more than one event in terms of precedence, overlap, and inclusion);
- b. Reason about the ramifications of an event (what is changed by virtue of an event);
- c. Reason about the persistence of an event (how long an event or the outcome of an event persists);
- d. Determine whether an event actually happened, according to the text, or whether it was merely an intention, or even something that had been avoided or prevented.

This entailed examining a broader range of linguistic issues than expected, as well as reconciling the insights and generalizations of the various theories accounting for the data. In the event, the specification covers the areas of event semantics, tense and aspect, temporal logic, as well as the semantics of modal contexts and modal subordination [63].

4 Theoretical Models

Events are located in time, relate to each other and have an internal temporal structure such as having a duration or being punctual, having a build-in endpoint or not. Moreover, language users can consider them as ongoing or completely, or even as real or potential. All these aspects of events, and our attitude towards them, can find their reflection in language. We discuss them in the following subsection.

4.1 Tense and Aspect

Location in time is typically expressed through **tense**, defined by Comrie as “the grammaticalized expression of location in time” [15]. This grammaticalized expression involves the marking of particular syntactic elements, e.g., the verb and auxiliaries. For example, in *John ran a marathon*, the past tense morpheme is used to indicate that the event occurred at a time earlier than the speech time. In *John will run a marathon*, the modal auxiliary *will* is used to locate the event as occurring at a future time, i.e., later than the speech time. Tense is mainly marked on the verb and auxiliaries associated with the verb group but in some languages, it can be marked on the noun phrase [15], whereas languages such as Mandarin Chinese lack morphemes and use aspectual markers to express location in time, though sometimes even these may be absent [41]. There are also non-grammaticalized expressions of location in time, e.g. through temporal adverbials, such as *tomorrow*, *yesterday*, *two hours later*, etc. While a few languages lack tense altogether and are not able to distinguish past from present or future, they all have a **realis/irrealis** distinction [15].

In English, it is usually assumed that there are two morphologically expressed tenses, **present** and **past**, while there are grammatically three tenses, with the inclusion of **future**. The present tense usually locates events as occurring at the speech time, and a typical use is the “reporting present”, as seen in live sports broadcasts. There are also informal uses of the present to convey a past event, such as *so then he says* Present can also be used for imminent or projected future events, as in *I arrive at noon* and *we leave tomorrow*.

The past tense usually refers to a time prior to speech time. Thus, *Mary ate a cookie*, indicates that there is a time prior to the speech time when the eating event occurred. The past tense can also involve **definiteness**, i.e., the speaker has a particular time in mind, as in the example of [55], *I didn’t turn off the stove*. This is a reference to a specific situation where the stove was not turned off.

The **future tense** usually refers to a time after the speech time though, like the other tenses, it can also be used as an epistemic present e.g., *He’ll be at work by now*. Furthermore, our concept of the future is not really symmetric with that of the past, since the future involves branching possibilities. This lack of a simple correspondence

between morphological tenses and the categorization of locations in time is one of the reasons that temporal annotation is difficult and important.²

4.2 Event Semantics

Because the overall goal of our model is to create a specification that supports rich temporal awareness for reasoning, it is crucial to have a model of events as expressed in language, since events will be the package within which propositional information is contained and subsequently reasoned about. The notion of **event structure** is a representation of events as complex objects with specific components. Taken together, aspect and event structure provide what, in computational terms, is often called an **event ontology**: a theory of the different subclasses (related by what are sometimes called, in the AI literature, **isa** links), components (related by what are sometimes called **part-of** links), and temporal properties of events. Event ontologies are an important part of the specification because they enable one to make semantic distinctions found in natural language text.

There are three major approaches in linguistics to the modeling of events in language.

- (6) a. **Aktionsarten**: Predicates in language can be classified according to their event type or aspectual class, in order to specific capture grammatical and semantic behaviors [73];
- b. **Events as Arguments**: Predicates in language have an *event variable* that can be treated as a first-order individual in the semantics, to enable logical inference [16];
- c. **Typed Event Structure**: This combines the insights of both of the above approaches, resulting in a typed event structure representation [59].

The best known version of the approach in (6a) is that initiated by Vendler [73]. It groups verbs into various subclasses based on their temporal properties. Vendler notes that verbs which describe **activities** like running, working, etc., express actions that “consist of successive phases following each other in time.” As a result, it is natural to express events by means of a ‘continuous tense’, i.e., a verb in the progressive form (*John is running*). Vendler characterizes verbs that describe activities as **processes**. By contrast, **states** do not involve successive phases, as a result, they sound odd in the progressive form, e.g., **John is knowing*. Vendler also observes that while running or pushing a cart has no set terminal point, running a mile and drawing a circle do have a “climax.” Thus processes are distinguished from a further class of events that culminate, called **accomplishments**. Vendler then goes on to distinguish the class of **achievements**, namely events like reaching a hilltop or winning a race that can be

²See [44] for a review of tense and aspect in the context of temporal reasoning and event semantics.

predicated for single moments of time. Since achievements don't extend over time, they can't in general co-occur with "for" adverbials.

Statives are expressed by verbs like *have*, *love*, *believe*, *know*, etc., but also by adjectives with a copula, as in *is clever*, *is happy*. For any state *p* (like John's being hungry) that holds over a period *t*, *p* must also hold for every sub-interval of *t*. This subinterval property is a characteristic of states. Statives either denote a situation or entry into the situation (ingressive or inceptive readings, respectively). Thus, *John knows* describes a state, whereas *John realizes* describes entry into a state; hence *John is (gradually) realizing what's happening* is acceptable, but **John is knowing what's happening* is odd.

Activities are expressed by verbs such as *walk*, *run*, etc. and differ from other eventualities in that if an activity *p* (like John's walking) occurs in period *t*, a part of the activity (also an activity) must occur for most sub-intervals of *t*. Activities usually allow temporal adverbials with *for* (e.g., *John ran for an hour*), do not take temporal adverbial phrases with *in* [18].

Accomplishments (associated with verbs like *build*, *destroy*, etc.) are eventualities which can logically culminate or finish. Unlike activities, 'x Vs for an hour', where *V* is an accomplishment, does not entail 'x Vs' for all times in that hour; likewise 'x is Ving' does not entail that 'x has Ved'. Thus, *John is cooking dinner* does not entail *John has cooked dinner*. Accomplishments do take temporal adverbial phrases with *in*.

Achievements (associated with verbs like *win*, *blink*, *find*, *reach*, etc.) are instantaneous (or short-duration) events that finish and occur in a very short time period. They cannot be modified by temporal *for*-adverbials: **John dies for an hour*.

Accomplishment and achievements, which are events that can culminate, are sometimes called **telic** eventualities. Finally, there are also instantaneous activities, called **semelfactives**, like *knock* or *cough*, which are instantaneous, atelic, and dynamic.

We now turn to the approach in (6b). Unlike Vendler's semantic classification, Davidson's "event as argument" approach was motivated by how different predicates behaved inferentially. His introduction of a first-order event variable in the representation also solves some long standing problems with adverbial modification in the interpretation of sentences [54,72]. Under this proposal, two-place predicates such as *eat* and three-place predicates such as *give* contain an additional argument, the event variable, *e*, as depicted below.

- (7) a. $\lambda y \lambda x \lambda e [\text{eat}(e, x, y)]$
 b. $\lambda z \lambda y \lambda x \lambda e [\text{give}(e, x, y, z)]$

In this manner, Davidson is able to capture the appropriate entailments between propositions involving action and event expressions through the conventional mechanisms of logical entailment.

- (8) a. Mary ate an apple.
 b. Mary ate an apple in the kitchen.
 d. Mary ate an apple at 3:00pm.
 e. Mary ate in the kitchen at 3:00pm.

In this example, we can capture the inferential relation between modification by adverbs of manner, place, and time, and the underlying event.

- (9) a. $\exists e \exists x [\text{eat}(e, m, x) \wedge \text{apple}(x)]$
 b. $\exists e \exists x [\text{eat}(e, m, x) \wedge \text{apple}(x) \wedge \text{in}(e, \text{the_kitchen})]$
 c. $\exists e \exists x [\text{eat}(e, m, x) \wedge \text{apple}(x) \wedge \text{at}(e, 3:00\text{pm})]$
 d. $\exists e \exists x [\text{eat}(e, m, x) \wedge \text{apple}(x) \wedge \text{in}(e, \text{the_kitchen}) \wedge \text{at}(e, 3:00\text{pm})]$

There are of course many variants of the introduction of events into predicative forms, including the identification of arguments with specific named roles (or partial functions, cf. [13, 19]) such as thematic relations over the event, as in [54].

The final approach we review here, (6c), is that of “typed event structure”. Event structure representations adopt and extend the typological distinctions introduced by Vendler and combine them with the “event as argument” approach introduced by Davidson and Parsons [50, 58, 59]. Overall, we can characterize a general class of **eventualities**. The Vendler distinctions are structurally defined with an internal subevent structure. In some respects, this can be seen as extending the decompositional approach presented in [18] by explicitly reifying the events and subevents in the predicative expressions. Unlike Dowty’s treatment of lexical semantics, where the decompositional calculus builds on propositional or predicative units, a “syntax of event structure” makes explicit reference to quantified events as part of the word meaning. Pustejovsky further introduces a tree structure to represent the temporal ordering and dominance constraints on an event and its subevents.

- (10) a. $\text{EVENT} \rightarrow \text{STATE} \mid \text{PROCESS} \mid \text{TRANSITION}$
 b. $\text{STATE} \rightarrow e$
 c. $\text{PROCESS} \rightarrow e_1 \dots e_n$
 d. $\text{TRANSITION}_{ach} \rightarrow \text{STATE STATE}$
 e. $\text{TRANSITION}_{acc} \rightarrow \text{PROCESS STATE}$

For example, the accomplishment denoted by “building a house” consists of the building process, followed by the state representing the result of the object being built. [21] adopts this theory in her work on argument structure, where complex events such as *break* are given a similar representation. In such structures, the process consists of what an agent, x , does to cause the breaking of y , and the following state represents the resulting broken item. The process corresponds to the outer causing event as discussed above, and the state corresponds in part to the inner change of state event. Both Pustejovsky and Grimshaw differ from the authors above in assuming a specific level of representation for event structure, distinct from the representation

of other lexical properties. Furthermore, they follow [16,23], and particularly [54], in adopting an explicit reference to (at least one) event variable in the parameter structure of the verbal semantics.

Recently, [40,67] have adopted important aspects of the event structure model for their analysis of the resultative construction in English; event decomposition has also been employed for properties of adjectival selection, the interpretation of compounds, and stage and individual-level predication ([12,17,29]).

Research done by [20,38,39,71] and others enriches this typology by developing a theory of how an event is shaped by the incremental participation of the theme in that event. The central aspect of this view is that an accomplishment is defined as involving a homomorphism from parts of the event to parts of the incremental theme. Incrementality can be illustrated with the following examples.

- (11) a. John ate a hamburger.
b. Mary wrote a novel.

The process of eating something is an incremental activity and results in an accomplishment, described by reference to the quantized unit appearing in the direct object (theme) position, only when the entire hamburger has been eaten or the entire novel has been written.

Recent work on scalar change [3,22,35] and dynamic event semantics [52] suggests a new understanding of the interplay between verb meaning, event semantics and argument structure with these predicates, by focusing on the measurement of the change in value over the properties of the participants in each intermediate state during the event. Some of these fine-grained characteristics of events are reflected in the inferences one can draw from sentences and hence need to be annotated.

4.3 Time Expressions

While languages differ considerably in how time is expressed morphosyntactically, there are some generalizations in how temporal information is conveyed beyond the system of tense and aspect already discussed. Time can be referred to as an entity in itself or as a way to modify an entity, attribute, or event. When employed in the first way, it acts like a conventional NP argument to a predicate.

- (12) a. **Monday** works better than **Tuesday** for the meeting.
b. Mary likes **the morning**, since she is more awake.
c. **The 1960s** was a turbulent decade.

In its more typical use, time functions as a modifying phrase, e.g., an Adjectival, Adverbial, or a Prepositional Phrase (or bare temporal NP). Examples of these are illustrated below in (13).

- (13) a. Our **previous** meal was much cheaper.
 b. The plane arrived **late**.
 c. Our dinner is **at 8:00 pm**.
 d. Max teaches **Tuesdays**.

Models of temporal modification in language focus on the semantic contribution of the time expression to the overall meaning of the sentence. In the examples above, this is accomplished in different ways, but in each case, the temporal expression can be seen as anchoring or modifying an event in time. In (13a), the event is a meal that occurred just before an already mentioned meal, in (13b), the arrival of the plane is anchored at a time after that which had been expected, and so on. In parsing and the compositional construction of meaning, the identification and interpretation of such temporal expressions is crucial for constructing an operational representation for subsequent logical inference.

For the purpose at hand and the specific goals stated above, the most relevant distinction to make within temporal expressions is the manner in which they refer temporally. To this end, we can distinguish between the following types of temporal referring expressions [42,43]:

- (14) a. **Times:** *June 11, 1989, July 4;*
 b. **Durations:** *three months, several days;*
 c. **Frequencies:** *weekly, every year.*

Being able to identify and differentiate such expressions is important for situating the events in a text, referenced to either a calendar time or relative to each other.

Interpreting temporal expressions in language, however, is not as straightforward as simply distinguishing the types above. There are many complexities introduced by language that an annotation specification will have to deal with, if temporal information is to be properly modeled and interpreted.

- (15) a. **Temporal Relational Expressions:** expressions which specify a time in relation to another time or event; *two days before New Year's, a week after the party;*
 b. **Temporal Indexicals:** expressions which are contextually dependent, such as *now, the year before, two months ago;*
 c. **Vagueness:** expressions that do not specify the exact time or boundaries associated with the named interval; *the summer, after dinner.*

The phenomena above pose distinct challenges to the development of an annotation specification that is geared to help anchor events to times on a timeline.

As with interpretation into a formal model, for our present purposes, we will need to normalize all time expressions to a representation that can be mapped to an evaluation on a timeline. Such a normalization will allow us to simplify the interpretation by conflating the different ways a language has for referring to the

same time (e.g., 11/15/04, November 15, 2004, the 15th of November in 2004, etc.). It also resolves any indexical component there might be to a time expression. As mentioned above, many of them refer to a point in time via some indexical anchor, as seen in the following.

- (16) a. She arrives *today*.
- b. Your appointment is *next Friday*.
- c. Mary was sick *last week*.
- d. We were in Boston *in October*.
- e. Rob will be in London *in October*.

All of these expressions refer to a time, but they do not by themselves fully specify that time. They refer via reference to the moment of utterance—in the case of texts, what we will call the *document creation time*. One has to know the time of utterance in order to retrieve the time referred to and normalize them to some machine-readable form.³ However, English has numerous ways to express what might be called ‘determinate’ times, which cannot be determinately linked to a timeline. Some examples include: *in the Fall of this year*; *recently*; *yesterday morning*.

Such times cannot be interpreted directly as parts of a timeline, because their begin and end points are more or less vague. Nevertheless, they can be ordered with respect to most points on a timeline, and so a system for reasoning about events must have some way of normalizing them. We discuss a set of indicators in Sect. 5 that are useful for normalizing many such expressions.

The time expressions mentioned so far refer, with greater or lesser granularity and with greater or lesser precision, to coherent ‘chunks’ of the timeline. They provide a means for directly associating particular events with particular parts of the timeline, and are thus of primary importance to the annotation model. English contains two more kinds of time expressions which involve slightly more complex means of anchoring events to times. In fact they do not anchor time at all, but rather measure time, i.e., durations. Durations refer to quantities of time rather than parts of the timeline directly.

- (17) a. After *three weeks* John regained consciousness.
- b. The team took a *two-hour* flight to the game.
- c. He’s worked on this program for *twenty hours*.

The time expressions in these examples simply indicate the duration of events. Indeed, the events being measured need not be contiguous, as in the example in (17c), where reference is being made to the combined time spent on the program, and not one contiguous convex hull interval.

³There is an ISO standard, ISO 8601, that we will adopt as part of ISO-TimeML, which provides a useful standard for the purpose of normalizing times. See Sect. 5.4 below.

Another complication for time normalization arises when durations are used to measure a period between an event and another event or time. Consider the example below.

(18) The course begins *two weeks* from *today*.

We will call these *anchored durations*, because they express the time of an event by making explicit the duration of time between the event and a time. In fact, they can be said to be part of a compositional time expression. For example, in (18), the duration *two weeks* is anchored to the time expression *today*. Thus, in combination with the temporal preposition *from* and the time expressed by *today*, it refers to a time two weeks after the document creation time. Note that durations can also be used to anchor events to other events.

(19) John *finished* his book *three years* before its *publication*.

The time expression here does not refer to parts of the timeline, but indicates distance along it—the amount of time that separates the italicized events. As such, it does not directly anchor events to times, but may allow the time for an event to be inferred. Like durations anchored to times, the amount of time they indicate should be represented in any model for temporal awareness.

The final type of time expression mentioned in (14) is one of the most difficult to model semantically, namely *frequencies*. Examples of the manner in which frequencies can modify events are illustrated below.

- (20) a. You must take your medication *daily* in the morning.
 b. We see a movie with the children *every Saturday*.
 c. Max gets nervous before *every performance*.

These time expressions indicate what are referred to as “sets of times”. They refer neither to coherent chunks of the timeline, nor to distances along it, but to, roughly speaking, groups of distinct pieces of the timeline. They are used to place recurring events on the timeline, particularly when the recurrence is regular. While corpus study reveals that such time expressions are not common in English texts, they are one way English allows events to be associated with the timeline, and a language for representing the temporal aspects of English texts should have some way of normalizing them. We will return to this problem in Sect. 6.5.

4.4 Temporal Relations

Having examined the nature of events, tense, and temporal expressions, we turn now to the problem of ordering events relative to each other and relative to fixed temporal anchors, namely, *temporal relation identification*. Reasoning about time

is an essential competence that all humans possess and a signature of intelligent behavior in any cognitive system. Our ability to represent temporal knowledge of actions and events in the world is essential for modeling causation, constructing complex plans, hypothesizing possible outcomes of actions, and almost any higher order cognitive task.

As we saw above, natural language expresses temporal information through tense, aspect, temporal adverbials, and other devices. Motivated by linguistic considerations, our model must incorporate mechanisms for analyzing tense and aspect, and event ontologies for representing event classes and structure. These mechanisms were applied in some cases to identification of the temporal location of events mentioned in text. We now turn from linguistic considerations to considerations motivated by reasoning in general.

The problems of temporal reasoning involve in part, as in natural language, locating events in time. Thus, consider the following narrative.

- (21) a. Yesterday, John *fell* while *running*.
b. He *broke* his leg.

A temporal reasoning system needs to anchor the *falling*, *running*, and *breaking* events to the particular time (yesterday), as well as order the events relative to each other, e.g., the running precedes the falling, which precedes the breaking. Temporal reasoning is concerned with representing and reasoning about such anchoring and ordering relationships, and as such, relies on manipulating the most appropriate representations for events, states, and their temporal properties. The particular form of temporal representation depends on the type of reasoning problem under consideration, and we want our model to be as generally applicable as possible to the areas of computational linguistics, AI, and planning.

For temporal inference, knowing that the falling occurred before breaking, and that the falling occurred yesterday (facts obtained here from linguistic data), along with commonsense knowledge of the behavior of how things break and fall, may allow a system to infer that the falling precedes and causes the breaking, and that these events occurred yesterday. In planning, on the other hand, one is moving towards a desired outcome, so reasoning needs to proceed backwards from the goal to what needs to take place to make it happen.

In general then, the temporal relation problem entails determining the relation between all relevant events and times in a narrative or text. This includes:

- (22) a. **event-event relations:**
John *left* before Mary *arrived*.
b. **time-time relations:**
Mary left on *Tuesday last week*.
c. **event-time relations:**
The plane *landed at noon*.

A temporal logic allows one to use the representation and inference mechanisms of logic to reason about time. For this to happen, temporal information needs to be added to the logic. From a logical standpoint, there are two ways to provide for a temporal interpretation of a proposition:

- (23) a. Add a modal operator over the proposition, where temporal order is interpreted from the syntactic combination of an operator over an expression;
- b. Denote events and times as intervals with explicit ordering relations over them.

In modal solutions to anchoring the meanings of sentences in time, specifically modal temporal logic, operators play the combined role of verbal tense, temporal adverbials, as well as temporal prepositions and connectives. One such system introduced by Prior [57] is *Minimal Tense Logic*, known as K_t . For K_t , four axioms form the core knowledge about temporal relations:

- (24) a. $\phi \rightarrow \mathbf{H} \mathbf{F} \phi$: What is, has always been going to be;
- b. $\phi \rightarrow \mathbf{G} \mathbf{P} \phi$: What is, will always have been;
- c. $\mathbf{H}(\phi \rightarrow \psi) \rightarrow (\mathbf{H} \phi \rightarrow \mathbf{H} \psi)$: Whatever always follows from what always has been, always has been;
- d. $\mathbf{G}(\phi \rightarrow \psi) \rightarrow (\mathbf{G} \phi \rightarrow \mathbf{G} \psi)$: Whatever always follows from what always will be, always will be.

While such systems have become standard within computer science in the area of temporal database reasoning systems (as discussed in [47]), the use of modal operators for determining the relative temporal ordering of events to times and to each other is not widely adopted in natural language processing applications. One reason for this is that the number of relations grows quadratically to the number events and times in a text, making modal representations cumbersome and difficult to reason over. Just as significantly, the effort required in the human annotation of event-event relations using modal operators is considerably less intuitive than that associated with an approach where orderings between events and times are explicitly encoded as relations between first-order individuals.

A different approach to ordering times and events has emerged in the context of work in AI and temporal planning research, that of explicitly reified times and events. One of the most widely adopted attempts to model action and change in the early days of AI was the situation calculus [48,49].

One of the most influential developments of this approach to temporal representation and reasoning is [1]. In this system, temporal intervals are considered primitives and constraints (on actions, etc.) are expressed as relations between intervals. There is no branching into the future or the past. In Allen's interval algebra, there are 13 basic (binary) interval relations, where six are inverses of the other six, excluding equality.

- (21) a. before (b), after (bi);
 b. overlap (o), overlappedBy (oi);
 c. start (s), startedBy (si);
 d. finish (f), finishedBy (fi);
 e. during (d), contains (di);
 f. meet (m), metBy (mi);
 g. equality (eq).

We will motivate and illustrate these relation values through linguistic examples, beginning with *equality* in (25), as illustrated in (25).

- (25) a. $\text{equality}(x, y)$: the intervals x and y completely co-extend on the timeline, where neither x nor y extends beyond the other.

The ordinal relation of *before* (b) along with its inverse *after* (bi) is defined as follows:

- (26) a. $\text{before}(x, y)$: the interval x completely precedes the interval y with no contact or connection between x and y .
 b. $\text{after}(x, y)$: the interval x completely follows the interval y with no contact or connection between x and y .

These are illustrated by the examples in (27).

- (27) a. The rains **destroyed** the house. The owners are **filing** for flood insurance.
 b. The Senate **rejected** the judge after **learning** of his past criminal activities.

When an ordinal relation of *before* exists, $b(x, y)$, and there is no interval between x and y , we say that x *meets* y .

- (28) a. $\text{meet}(x, y)$: the interval x precedes the interval y where the final point of x touches the initial point of y .
 b. $\text{metBy}(x, y)$: the interval x follows the interval y where the final point of y touches the initial point of x .

This is illustrated below in (29).

- (29) The book **fell** to the floor. It **sat** there for days.

If the *before* relation holds for only the initial part of interval x relative to interval y , we have an *overlap* relation.

- (30) a. $\text{overlap}(x, y)$: the interval x partially precedes and partially intersects the interval y .

- b. `overlappedBy(x, y)`: the interval x partially intersects and partially follows the interval y .

The example in (31) illustrates this.

- (31) Bill **ate** a big breakfast. He was **full** before he was done.

When x and y have the same begin point but different end points, where x stops earlier than y , we have a *start* relation, defined below and illustrated in (33).

- (32) a. `start(x, y)`: the interval x begins at the same moment as interval y and ends before y terminates.
- b. `startedBy(x, y)`: the interval x begins at the same moment as interval y and continues on after x has terminated.

- (33) **The sunrise** occurred at 6:30 am **this morning**.

Similarly, when x and y have the same end point but different begin points, where x ends earlier than y , we have a *finish* relation, defined below with an example in (35).

- (34) a. `finish(x, y)`: the interval x begins at the same moment as interval y and ends before y terminates.
- b. `finishedBy(x, y)`: the interval x begins at the same moment as interval y and continues on after x has terminated.

- (35) They **reached** the summit of the mountain at noon. **The hike** took four hours.

Finally, consider the relation of complete temporal containment and its inverse, *during*.

- (36) a. `during(x, y)`: the interval x completely precedes the interval y with no contact or connection between x and y .
- b. `contains(x, y)`: the interval x completely follows the interval y with no contact or connection between x and y .

The example in (37) illustrates the *during* relation.

- (37) A baby **cried** during **the concert**.

These interval relations are illustrated graphically in the following table.

In Sect. 5.2.5, we show how to translate this approach to representing temporal relations into a model for linguistic annotation.

4.5 Subordinating Relations

In this section we address the final issue falling within the scope of the phenomena being modeled, namely how events are subordinated within diverse contexts of aspect, intention, belief, desires, plans, factuality, and other modalities. Thus far our discussion has focused on how to represent the basic meaning of times, events, and how they are anchored or ordered. In order to develop such representations, we have ignored the rich contexts within which events are typically embedded in language. Consider the following examples to illustrate this point.

- (38) a. John **might** have *bought* some wine.
 b. Mary **wanted** John to *buy* some wine.
 c. Bill **plans** to *visit* Paris next summer.

These cases reveal that in contexts where verbs are modally subordinated, or occur as arguments in intensional constructions, they cannot straightforwardly be taken as denoting real events. For reasoning purposes, from sentences (38a) and (38b), we do not know whether wine was purchased or not. Similarly, in (38c) there is inherent uncertainty in reference to a future event that has yet to occur.

There are, however, many contexts where the event which the subordinated verb denotes is guaranteed to have occurred, as shown below in (39) [36].

- (39) a. The man **forgot** that he had *locked* his car.
 b. Mary **regrets** that she didn't *marry* John.
 c. John **managed** to *leave* the party.

Complicating matters is that there are contexts that guarantee that the subordinated event does not occur, sometimes embedded by the same verb, as with the verb *forget* in (40a).

- (40) a. The man **forgot** to *lock* his car.
 b. Mary was **unable** to *marry* John.
 c. John **prevented** the *divorce*.

The examples above illustrate just a few of the types of *existential subordination* induced by specific predicative expressions: *modality* in (38), and *factivity* in (39) and (40). In fact, there is another subordinating relation that determines the *provenance* and the *veridicity* (truthfulness) of event statements. Consider the sentences below.

- (41) a. Five other U.N. inspection teams *visited* a total of nine other sites, the agency **reported**.
 b. U.S. officials **claim** they have *destroyed* the enemy's weapons.
 c. The witness **denied** that he *stole* the money.

The examples in (41) indicate how the source of a proposition describing an event must be recognized and classified, in order to perform the appropriate inferencing. [33] discusses the relevance of veridicity for Information Extraction and entailment tasks. Factuality is also critical in the area of opinion detection [74], given that the same situation can be presented as a fact in the world, a mere possibility, or a counterfact according to different sources. This is called the veridicity or factuality of the event in question.

The veridicity of the event referred to by the italicized word in each example is affected by the fact that it is embedded under the underlined verb. The sentence does not simply represent the event as being part of the actual past or present, or projected future. Instead, it expresses the event in qualified terms. This is very similar to modality, mentioned above. In (41a), for instance, the underlined event is qualified by being the argument of *report*. Its veracity depends on the reliability of the reporting agent.

The relations expressed between subordinated events and the events that subordinate them are not temporal relations, per se, (though they may have temporal implications); nevertheless, it is crucial that they are represented in a model for annotation. In order to effectively answer a question about an event it is very important to know whether the writer has presupposed its veracity, deferred responsibility for its veracity to another party, or presupposed its falsity. Thus, a language for modeling temporal information in texts should have some way to represent the different sorts of subordination relations that can be expressed. In Sects. 5.2.3 and 5.3, we present a complete set of relations for this purpose.

5 Developing a Preliminary Model

In this section, we focus on the process involved in creating an initial model and specification for the temporal phenomena that were identified for inclusion from the previous discussion.

The developers of TimeML adopted the MATTER cycle methodology described in Sect. 2. Perhaps most relevant for the initial development was the MAMA sub-cycle, where initial proposals for the model are tested with small annotation experiments. The iteration of *Model* and *Annotate* helps to refine the appropriate specification language for a specific task. Initially, the developers reviewed the various theoretical models for temporal expressions, tense and aspect, event semantics, and temporal relations mentioned in the previous section. Previous computational projects were also studied to determine what aspects of the overall goals had already been implemented in some form.

The TimeML annotation effort settled on four types of tags with different sets of attributes. The designers were careful to focus on the larger consequences of the annotation scheme, and to ensure that the logic of the resulting system was consistent so that the annotations could be used for both temporal inferencing and temporal entity recognition. Once it was determined which theories and implementations were

most useful to incorporate into TimeML, a corpus was selected and work was started on the model and specification of the tasks, implementing both the MAMA cycle and CASCADES model, as described below.

5.1 Identifying the Basic Elements

5.1.1 Temporal Expressions

Given the complexity of the inter-related phenomena as described in the previous section, it was decided that the model should treat both temporal and event expressions as denoting intervals of time. This greatly simplified how relations would be defined in the model, and reduced the cognitive load for the annotators as well.

Most of the previous work in this area focused on temporal expression identification and parsing. For example, the TIMEX tag emerged out of the Named Entity tagging subtask for DARPA's Message Understanding Conference (MUC-6) [51]. This task included the identification of persons, organizations, dates, locations, times. In the context of this task the TIMEX tag was introduced. The scope of TIMEX tagging extended only to absolute time expressions, where a specific date or time period was given (*May 1, 1901, Fall 2015*, etc.), while ignoring relative temporal expressions. These were added in the next competition, MUC-7 [14], which included phrases such as *last year, next week*, and so on.

As pointed out in [44], one of the limitations of the way time was modeled in both MUC-6 and MUC-7 tasks was that, while relative expressions were tagged, they were marked as dates and not given a relative reference. The modified specification introduced as TIMEX2 [75] deals with this problem, but still stopped short of actually providing a relative representation for the meaning of such expressions. In fact, it was fixing this limitation that was one of the main goals in the development of TimeML. To this end, a new tag, TIMEX3, was introduced that incorporated the encoding of the functional content of temporal expressions.

The initial inspiration for the TIMEX3 tag is obviously that of TIMEX2, and many of the attributes from that specification were adopted, with some modifications. There are four types of temporal expressions captured in the TIMEX3 tag: TIME, DATE, DURATION, and SET, corresponding to the types described in Sect. 4.3 above. An expression that receives the TIME type is one that refers to a time of the day, even if in a very indefinite way. The easiest way to distinguish a TIME from a DATE is to look at the granularity of the expression. If the granularity of the expression is smaller than a day, then the expression is a TIME. For example, *3:15 pm* and *late last night* are both TIME expressions, while *the summer of 1964* and *October 1, 1999* are DATE expressions. A TIMEX3 is typed as a DURATION if it explicitly describes some extent of time. Examples of this are: *three weeks, all last night*, and *two days*. Finally, the SET type was proposed for expressions that describe a set of regularly reoccurring times. Examples include: *twice a week, every month*.

5.1.2 Event Expressions

From examination of previous efforts at event annotation, it was determined that the approach coming closest to the goals of TimeML was that of [69], whose work provided a platform from which the TimeML group was able to develop a richer and more expressive model.⁴ Its four tag types of EVENT, Timex, SIGNAL, and DOA (Date of Article), as well as some of the basic attributes for EVENT were adopted by TimeML.

Unlike previous named entity annotation efforts such as ACE and MUC, where event markup was focused on thematic domains and “events of interest” [70], the goal of TimeML was to annotate eventualities broadly, as characterized by the linguistic theories covered in Sect. 4. Hence, any predicate denoting a state, process, achievement, or accomplishment, could potentially be annotated with an EVENT tag. Furthermore, this tag must be part-of-speech agnostic; that is, it must be able to markup verbs, event nominals, and event-denoting adjectives.

The attributes for the EVENT tag in STAG encode inherent properties of an event: the type of event (occurrence, perception, etc.) and tense and aspect attributes characterizing the verb form. In addition, it encoded “related ToEvent” and “related-ToTime” attributes that designated what other event and/or time in the text the item being annotated was related to, and eventRelType and timeRel Type attributes that indicated how the marked event was related to the indicated event or time. However, the TimeML working group felt that this latter attribute was inappropriate for an entity element. Unlike attributes that have a value from a fixed numeric or sortal array, such as tense and aspect, this attribute encodes relations to other events and times, and an arbitrarily large number at that. Because of these concerns, it was proposed that a new LINK tag be created to capture the relational information between events and times without it being embedded within the attribute value of any specific time or event. This will be discussed below.

5.1.3 Temporal Signals

Signals are explicit markers indicating a temporal relation between times and events. These include mainly temporal prepositions, such as *at*, *on*, *before*, and *after*.

- (42) a. Mary left_{e₁} Boston *on* Thursday_{t₁} . during(*e₁*, *t₁*)
 b. The children slept_{e₁} *before* they ate_{e₂} dinner. before(*e₁*, *e₂*)

There are some instances where the verbal predicate itself signals a temporal relation, such as the verbs *precede* and *follow*, as used in their temporal sense. In these cases, the verb denotes the temporal ordering relation between the events occurring as arguments to the relation.

⁴Setzer’s work came to be known as STAG (Sheffield Temporal Annotation Guidelines) by the working group.

- (43) a. The wedding_{e₁} *precedes* the reception_{e₂}. before(*e₁*, *e₂*).
 b. New Year's_{e₁} *follows* Christmas_{e₂}. after(*e₁*, *e₂*).

The SIGNAL tag existed in STAG, and it was not changed significantly when it was adopted by TimeML. The only attribute was an ID that could be referenced by other tags.

5.1.4 Links

Three types of LINK relations were studied and introduced by the TimeML working group: temporal relations (TLINK); aspectual relations for events (ALINK); and existential subordination links (SLINK).

As mentioned above, the original STAG specification contained no explicit tag for temporal relations: rather, all relations were encoded as attributes to EVENT instances. The TimeML working group decided that all overt relations between times and events should be represented as transparently as they are in the temporal interval algebra of [1]. This gave rise to the TLINK tag. The TLINK tag was given attributes that allowed it to represent the temporal relationship between two events, events and times, or two times and those attributes were removed from the EVENT tag. The TLINK has three ID attributes: **eventID**, **timeID**, and **signalID**. It also has two attributes to indicate what type of object was being linked to: **relatedToEvent** and **relatedToTime**; and a **relType** attribute that contained an expanded set of relationships (based on the Allen relations discussed in Sect. 4.4).

Not only did the TLINK tag take the burden of expressing information about temporal relationships off of the EVENT tag, but it also made it possible for a (theoretically) unlimited number of relationships to be expressed in connection to a single event or time: this allowed for much more expressive temporal relations, and far more complete reasoning about the relationships between time and events.

In addition to the TLINK, it was decided that aspectual information associated with an event should be captured by a unique tag, called an ALINK. This facilitated the markup of verbs that refer to phases of an event, such as: **initiation**, **culmination**, **termination**, and **continuation**. These represent a relation between an aspectual event and its embedded event predicate as with: *begin to cry*; *finish eating*, and so on, discussed briefly in Sect. 4.2 and more extensively in [63].

Finally, it was determined that, in accordance with the reasoning demands on the domain, some mention of modally subordinated events needed to be made explicit, indicating what degree of actuality, factuality, or evidentiality is to be associated with the event [31,32]. Hence, a subordinating link, SLINK, was created, with the following values: **modal**, **factive**, **counter-factive**, **evidential**, and **conditional**.

5.2 Details of the Initial Model

5.2.1 What Not to Include in the Model

Before we discuss the structure and content of the major components of TimeML, it is worth reviewing the manner in which the scope and extent of a model is determined. From the outset, it was clear that events, time expressions, and the relations between them would be the core elements of the specification language. However, as mentioned in the previous section, there are some linguistic phenomena (and their associated syntactic contexts) that were incorporated into TimeML, which are not typically thought of as strictly temporal in nature, such as the modal embedding relations mentioned above, e.g., factivity and reporting contexts. These were seen as critical because of their role in subsequent inferencing, reasoning, and question answering tasks that are dependent on events and how they are ordered.

There were other linguistic phenomena, however, which were seen as falling outside the scope of TimeML. Among these was the identification of event participants. Consider, for example, the sentences in (44).

- (44) a. Fido *chewed*_{*e*₁} a bone.
 b. Mary *bought*_{*e*₁} a car.

The event semantic frameworks reviewed in Sect. 3 would suggest that these sentences would have logical interpretations such as those given in (45).

- (45) a. $\exists x[\text{chew}(e_1, \text{fido}, x) \wedge \text{bone}(x)]$
 b. $\exists x[\text{buy}(e_1, \text{mary}, x) \wedge \text{car}(x)]$

Nevertheless, it was decided that, since annotated resources for verb-argument listings already existed, such as PropBank [53], FrameNet [2,68], and VerbNet [37], it was not necessary to include such information in the schema for TimeML. Any information recoverable by virtue of linking or association with an existing resource, it was felt, was redundant and only added to the cognitive load of the annotation task. Hence, in (44), only the event spans (*chew* and *bought*) are identified by the specification for TimeML: event participant information can be retrieved by dependency parsing or access to a lexical resource that encodes such information.

In the remainder of this section, we examine the major elements that were created in TimeML, along with the syntax for expressing them.

5.2.2 TIMEX3

The core of any specification aiming to provide temporal understanding is a rich language for representing temporal expressions, which TimeML models with the TIMEX3 tag. As we discussed in Sect. 5.1.1, there are four types of temporal expressions captured in TIMEX3: TIME, DATE, DURATION, and SET. The syntax for major attributes of the TIMEX3 tag is shown below.

```

attributes ::= tid type (value | valueFromFunction) [mod] temporalFunction
               [anchorTimeID] [functionInDocument]  tid ::= t<integer>
type ::= 'DATE' | 'TIME' | 'DURATION' | 'SET'
temporalFunction ::= ('true' | 'false') {default, if absent, is 'false'}
anchorTimeID ::= t<integer>
ValueFromFunction ::= t<integer>
functionInDocument ::= 'CREATION_TIME' | 'EXPIRATION_TIME' |
                       'MODIFICATION_TIME' | 'PUBLICATION_TIME' |
                       'RELEASE_TIME' | 'RECEPTION_TIME' | 'NONE'

```

The type of a temporal expression is represented in the tag along with a specific value for the time expression. A temporal expression's value is annotated with an extension of the ISO 8601 standard. For example, a fully specified temporal expression such as the one in (46a) has a value of "2004-11-22". A TimeML annotation produces XML as in example (46b).

- (46) a. November 22, 2004
 b. <TIMEX3 tid="t1" type="DATE" value="2004-11-22"> November
 22, 2004
 </TIMEX3>

When a temporal expression is not fully specified, placeholders can be used in the value attribute. For example, an expression such as *March 14* provides no year information, but can be given a value of XXXX-3-14. In the case of times and dates, these placeholders are generally removed in favor of a more complete annotation provided by temporal functions.

The first attribute value of note for durations is contained in value. Durations are required to have a particular format in this attribute because they represent a period of time. A sample annotation for a simple duration is given in (47).

- (47) <TIMEX3 tid="t1" type="DURATION" value="P3D"> three days
 </TIMEX3>

Durations may also use two additional TIMEX3 attributes: beginPoint and endPoint, which are used to model *anchored durations*. For example, the expression *a week from Monday* has a begin point, namely, the tid for *Monday*. With this information, the actual date that the full phrase refers to can be calculated. TimeML allows for an additional TIMEX3 to be created to annotate the missing point.

- (48) <TIMEX3 tid="t1" type="SET" value="P1W" quant="EACH"
 freq="3D"> 3 days each week </TIMEX3>

Perhaps the most innovative aspect of the TIMEX3 tag over TIMEX2, as discussed in the previous section, is how underspecified expressions are handled. As is clear from the above examples, many temporal expressions are missing information critical to a full specification and interpretation. TimeML introduced the notion of a *temporal*

function to signify that an expression was dependent on other temporal markers or context for a full interpretation.

For example, news articles typically include a specific document creation time (DCT) as part of the document metadata. If the text refers to *today*, that expression is anchored to the DCT to complete its specification. In the same manner, an expression such as *July 9* is underspecified until the appropriate year is supplied. Since such information can often be extracted from the DCT, it is anchored to that TIMEX3 and the correct year is added to the value of the *July 9* TIMEX3. This is done through an attribute called `temporalFunction`. When an expression requires an anchoring to be completely specified, `temporalFunction` receives a “true” value. The underspecified TIMEX3s still have three core attributes: `tid`, `type`, and `value`. When a temporal function is also used, three more attributes are added:

- (49) a. `temporalFunction` – a boolean attribute that indicates that a function is necessary
- b. `anchorTimeID` – the `tid` of another TIMEX3 that provides information to the temporal function
- c. `valueFromFunction` – the `tfid`, or temporal function ID, of the function that completes the TIMEX3 specification

Note that both the `value` and `valueFromFunction` attributes are used above, since expressions that require functions, by definition, do not contain enough information to provide a value. However, it is not always the case that the expression lacks any specific temporal information at all. In cases such as *today*, the tagged item cannot lend any information to the `value` attribute and the temporal function must do all the work. Still, cases such as *Wednesday* do contain specific information that should be captured by the TIMEX3 tag. In the former case, the `value` must be something like “XXXX-XX-XX”, where the X-placeholder is used to show that the format of this value should be that of a DATE, but that no other information has been provided. In the latter case, though, it is useful to capture that the expression makes use of specific temporal information by giving a value of “XXXX-WXX-3”.

Consider how the underspecified temporal expressions in (50) are represented as functional terms.

- (50) a. The conference was *this week*.
- b. Mary arrives *this week*.

By interpreting *this week* as a temporal function, we return the enclosing time period of the specified type given in `scale`, namely the type of time period (granularity); “hour, minute, day, year”. This permits an interval containing the speech time of the present, so that both (50a) and (50b) can be included within this week.

- (51) `<TIMEX3 tid=“t1” type=“DURATION” value=“P1W” temporalFunction=“true” value-FromFunction=“tf1” anchorTimeID=“t0”> this week < /TIMEX3> <CoerceTo tfid=“tf1”`

```
argumentID="t0" scale="WEEK"/ >
<Predecessor/Successor tfid= argumentID= count= signalID= / >
```

Similarly, the expression in (52) specifies the scale and the time point which anchors the interval into the past.

- (52) a. They traveled to Boston *four weeks ago*.
 b. <TIMEX3 tid="t1" type="DURATION" value="P4W" temporalFunction="true" valueFromFunction="tf1" anchorTimeID="t0"> 4 weeks </TIMEX3> <SIGNAL sid="s1"> ago </SIGNAL> <CoerceTo tfid="tf2" argument="tf1" scale="WEEK"/ > <Predecessor tfid="tf1" argument="tf2" count="4" signalID="s1"/ >

5.2.3 EVENT

The EVENT tag is used to annotate those elements in a text that describe what is conventionally referred to as an eventuality (see Sect. 4.2). Syntactically, events can be expressed as inflected verbs, event nominals, as well as adjectival phrases. The syntax and definition for the EVENT tag is shown below.

```
attributes ::= id offset pred class type pos tense aspect
              polarity mood [modality] [comment]
class ::= 'OCCURRENCE' | 'STATE' | 'PERCEPTION' | 'REPORTING' |
          'ASPECTUAL' | 'I_STATE' | 'I_ACTION'
type ::= 'STATE' | 'PROCESS' | 'TRANSITION'
pos ::= 'ADJECTIVE' | 'NOUN' | 'VERB' | 'PREPOSITION' | 'OTHER'
tense ::= 'FUTURE' | 'PAST' | 'PRESENT' | 'IMPERFECT' | 'NONE'
aspect ::= 'PROGRESSIVE' | 'PERFECTIVE' | 'IMPERFECTIVE'
          | 'PERFECTIVE_PROGRESSIVE' | 'IMPERFECTIVE_PROGRESSIVE' | 'NONE'
vform ::= 'INFINITIVE' | 'GERUNDIVE' | 'PARTICIPLE' | 'NONE'
polarity ::= 'NEG' | 'POS' {default, if absent, is 'POS'}
mood ::= 'SUBJUNCTIVE' | 'NONE'
         {default, if absent, is 'NONE'}
modality ::= CDATA
```

Much like the TIMEX3 tag, TimeML captures several different types of event. The type of event is encoded in the `class` attribute. These types are discussed in Sect. 4.2. Some of the categories under the class attribute go beyond purely temporal notions. As discussed in Sect. 4.5, to determine whether an event has occurred, one has investigate the modalities connected to veridicity. The TimeML event classes reflect this [64].

- (53) a. **Reporting:** When a person or organization declares something, narrates an event, or informs about an event, the event that describes that action is of the REPORTING class. These are generally verbs such as: *say, report, tell, explain, state*.
 b. **Perception:** This class includes events that involve the physical perception of another event. Such events are typically expressed by verbs like: *see, watch, glimpse, behold, view, hear, listen, overhear*.

- c. **Aspectual:** In languages such as English and French, there is a grammatical device of aspectual predication, which focuses on different phases of an event. This is accomplished in other languages morphosyntactically, through affixation on the matrix verb, and is part of a more complex Tense-Aspect-Mood (TAM) system in some languages. The event phases modeled in TimeML are: Initiation: *begin, start*; Reinitiation: *restart, reinitiate, reignite*; Termination: *stop, cancel*; Culmination: *finish, complete*; Continuation: *continue*. Events that are of this class also participate in a particular kind of TimeML link called an ALINK (for “Aspectual Link”) so that the relationship between the ASPECTUAL event and the one it predicates over can be shown.
- d. **I_Action:** An I_ACTION is an Intentional Action. An I_ACTION introduces an event argument, which must be in the text explicitly. The event argument describes an action or situation from which we can infer something given its relation with the I_ACTION. For instance, the events introduced as arguments of some I_ACTIONS may not necessarily have occurred when the I_ACTION takes place. Explicit performative predicates are also included here. Note that the I_ACTION class does not cover states as they have their own associated classes. For the most part, events that are tagged as I_ACTIONS are in a closed class. The following list provides a sampling of this class: *attempt, try, scramble, investigate, investigation, look at, delve, delay, postpone, defer, hinder, set back, avoid, prevent, cancel, ask, order, persuade, request, beg, command, urge, authorize, promise, offer, assure, propose, agree, decide, swear, vow, name, nominate, appoint, declare, proclaim, claim, allege, suggest*.
- e. **I_State:** I_STATE events are similar to the previous class. This class includes states that refer to alternative or possible worlds, which can be introduced by subordinated clauses, nominalizations, or untensed VPs. Here is a list of events that fall into this category: *believe, think, suspect, imagine, doubt, feel, be conceivable, be sure, want, love, like, desire, crave, lust, hope, expect, aspire, plan, fear, hate, dread, worry, be afraid, need, require, demand, be ready, be eager, be prepared, be able, be unable*.
- f. **State:** STATES describe circumstances in which something obtains or holds true. However, only certain events in this category are annotated in TimeML: those that are identifiably changed over the course of the document being marked up. Remember that TimeML’s chief concern is to annotate temporal events. If a STATE is deemed persistent throughout the event line of the document, it is factored out and not annotated. Conversely, if a property is known to change during the course of events represented or reported in the article, that property is marked as a STATE.
- g. **Occurrence:** This class includes all the many other kinds of events describing something that happens or occurs in the world. Essentially, this is a catch-all category for events that participate in the temporal annotation, but do not fit into any of the above categories.

The annotation of an EVENT is quite simple as it only includes the `class` attribute and a tag that identifies it. TimeML at first distinguished between event **tokens** and event **instances** or realizations. The tag `MAKEINSTANCE` was used to create the actual realization of an event. The motivation for this distinction came from examples like *John taught on Monday and Tuesday*, where one verb represents two events. In order to be able to annotate such cases, it is necessary to create two **instances** of *taught*, representing the two different event occurrences. `MAKEINSTANCES` are created in addition to the event annotation (which marks up the event token). As we will see, however, this tag was abandoned in the move to ISO-TimeML as unnecessary.

5.2.4 SIGNAL

A signal is a textual element that makes explicit either the relation holding between two entities (time and event, time and time, or event and event) Examples of SIGNALS are shown below:

- (54) a. **Temporal prepositions:** *on, in, at, from, to, before, after, during*, etc.
 b. **Temporal conjunctions:** *before, after, while, when*, etc.

Unlike verbal predicates in a semantic role annotation task, the temporal relation values that hold for the arguments to a temporal preposition are encoded in the `TLINK`, which we discuss below.

5.2.5 Modeling Temporal Relations with TLINK and ALINK

Probably the biggest change from earlier approaches to temporal annotation is in how relations between events and times are encoded. The incorporation of the `LINK` tag at the very beginning was a big change, but in the stable version of the TimeML specification, there were actually three different types of link tags: `TLINKs`, `ALINKs`, and `SLINKs`. `TLINKs` encode relationships between temporal objects: event-event links, time-event links, event-time links, and time-time links. The syntax for the major attributes for `TLINK` is shown below.

```
leventInstanceID ::= IDREF
{eventInstanceID ::= EventInstanceID}
timeID ::= IDREF
{timeID ::= TimeID}
signalID ::= IDREF
{signalID ::= SignalID}
relatedToEventInstance ::= IDREF
{relatedToEventInstance ::= EventInstanceID}
relatedToTime ::= IDREF
{relatedToTime ::= TimeID}
relType ::= 'BEFORE' | 'AFTER' | 'INCLUDES' | 'IS_INCLUDED' | 'DURING' |
'SIMULTANEOUS' | 'IAFTER' | 'IBEFORE' | 'IDENTITY' |
'BEGINS' | 'ENDS' | 'BEGUN_BY' | 'ENDED_BY' | 'DURING_INV'
```

TLINKs are the most general-purpose of the links, and they also incorporate information about any signals that are influencing the relationship between the two objects. The relTypes are based on the temporal relationships defined by Allen [1] and discussed above in Sect. 4.4.⁵

The other temporal relation introduced is the ALINK, or aspectual link tag. These are used to take care of sentences such as “The boat began to sink” that we discussed before: they mark that the link being annotated has a temporal relationship, but they also mark what phase of the event is being discussed.

```

lid ::= ID
{lid ::= LinkID
LinkID ::= l<integer>}
eventInstanceID ::= ID
{eventInstanceID ::= EventInstanceID}
signalID ::= IDREF
{signalID ::= SignalID}
relatedToEventInstance ::= IDREF
{relatedToEventInstance ::= EventInstanceID}
relType ::= 'INITIATES' | 'CULMINATES' | 'TERMINATES' |
'CONTINUES' | 'REINITIATES'
comment ::= CDATA
syntax ::= CDATA

```

5.3 Modeling Subordination with SLINK

As mentioned in previous sections, The SLINK is a subordination link that is used for contexts involving modality, evidentials, and factives. An SLINK is used in cases where an event subordinates another event with this modal force. These are cases where a verb takes a complement and subordinates the event instance referred to in this complement.

```

lid ::= ID
{lid ::= LinkID
LinkID ::= l<integer>}
origin ::= CDATA
eventInstanceID ::= IDREF
{eventInstanceID ::= EventInstanceID}

```

⁵The relType value ‘IDENTITY’ is actually not part of Allen’s calculus, but was used for event coreference.

```

subordinatedEventInstance ::= IDREF
{subordinatedEventInstance ::= EventInstanceID}
signalID ::= IDREF
{signalID ::= SignalID}
relType ::= 'MODAL' | 'EVIDENTIAL' | 'NEG_EVIDENTIAL' |
'FACTIVE' | 'COUNTER_FACTIVE' | 'CONDITIONAL'
comment ::= CDATA
syntax ::= CDATA

```

Initially, SLINKs were also used to mark matrix modal modification and predicative negation, but sentences such as “John may not want to teach on Monday” would have had three SLINKs and proved far too difficult to annotate effectively or accurately.

5.4 TimeML Becomes ISO-TimeML

Shortly after the completion of a stable TimeML specification, the TimeBank corpus [62] was released through the LDC. This, in turn, generated considerable interest in the specification within the community [4,45,46,65]. In addition, ISO noticed the effort and successfully adopted it as an initial draft into their TC37/SC4 Semantic Annotation Framework. The transformation of TimeML into an ISO standard did not happen overnight. In fact, it took several years for the specification to be finalized and finally approved. In this brief discussion, we give an overview of some of the major changes that emerged as TimeML was shaped into an international standard [66].

The first and probably biggest change for the specification was that it had to be made more abstract: the model for the original TimeML was rooted in the idea that all annotations using the TimeML specification would be using an XML format for their data, but that assumption couldn't be made for an international standard. Therefore, the ISO-TimeML model had to be expanded so that it could be represented in any number of formats, even very different ones such as a UML (Unified Modeling Language) diagram, or in different programming languages such as Lisp or Prolog. Doing this meant that the ISO-TimeML working group had to be able to clearly express the relationships between tags and their attributes, and how they could be connected to one another, so that those relationships could be modeled in other representations.

The next change that was needed was to make ISO-TimeML compliant with other ISO standards, such as the Linguistic Annotation Framework (LAF) [27]. Since the heavy lifting of creating a more abstract model had already been done, this primarily involved modifying the tags so that they could be used in a stand-off annotation format. The one used for ISO-TimeML is a token-based (rather than character-based) annotation.

Following ISO DIS 24612 (*Language resource management - Linguistic annotation framework*) and [26,27], the transformation to ISO-TimeML required adopting a

fundamental distinction between the concepts of *annotation* and *representation*. The term ‘annotation’ is used to refer to the process of adding information to segments of language data, or to refer to that information itself. This notion is independent of the format in which this information is represented. The term ‘representation’ is used to refer to the format in which an annotation is rendered, for instance in XML, independent of its content. According to the proposed international standard (LAF), *annotations* are the proper level of standardization, not representations. Hence, ISO-TimeML defines a markup language for annotating documents with information about time and events at the level of annotations. ISO 24617 SemAF consists of a series of standards on semantic annotation. While SemAF-Time, namely Part 1, deals with temporal and event-related annotation, Part 2 SemAF-Dacts treats dialogue acts in dialogue material and other proposed work items deal with other aspects of semantic annotation such as semantic roles and named entities. With its graph-theoretic model, LAF offers a pivotal frame that ties together all these different parts of semantic annotation as well as other annotation schemes for language resource management into an interoperable system. Such an interoperable system is, however, efficiently constructed if and only if the representation scheme is also constructed and provided uniformly. The XML-based representation scheme of ISO-TimeML is designed to satisfy such requirement by conforming to LAF and other standards [28].

In terms of the tags and their attributes for ISO-TimeML, the ways that temporal relationships were handled had to be expanded so that the specification provided a more expressive and systematic treatment for three major phenomena:

- (55) a. ORDER: How an event or time is positioned in a timeline relative to other events or times;
- b. MEASUREMENT: The size of a temporal entity, e.g., an event’s duration or the length of a temporal interval;
- c. QUANTITY: The number of events denoted in an expression.

The original versions of TimeML actually handled the “order” characteristic quite well: TimeML already had a full set of temporal relations in the *relType* of the different link tags, so those remained unchanged in the ISO specification. However, the “measure” characteristic of the text was not so adequately covered. The original TimeML had a *type* = ‘DURATION’ option for the TIMEX3 tag, but that did not fully capture the different types of meanings that a duration could imply. Consider the different interpretations of these two sentences: “Before leaving the house, I slept for two hours” and “Before getting my pilot’s license, I flew for 300 h.” In the first sentence, we can reasonably interpret that the speaker slept for the full two-hour span, without any breaks. However, the same assumption decidedly cannot be made for the second sentence, despite the fact that both sentences have the same basic syntax. To more fully express the differences, a new type of link tag, the MLINK (measure link), was introduced in ISO-TimeML. Essentially, the MLINK is used to explicitly state that a TIMEX3 expression is used to measure the duration of an event.

The “quantity” characteristic was also somewhat underspecified in TimeML: as we mentioned in previous discussions, phrases where a single extent indicated that multiple events were taking place, such as “teach every Tuesday,” are difficult to annotate. This is solved in ISO-TimeML by adding a “scopes” attribute to the TIMEX3 tag. This allows the tag to have a relationship with the “teaches” event that is not limited to the relType of the link, but rather provides a more open (but still semantically clear) interpretation of the expression.

One other major change from TimeML to ISO-TimeML is the removal of the MAKEINSTANCE tag. Annotators found the MAKEINSTANCE tag difficult to annotate, and so the attributes from that tag were placed back into the EVENT tag, which allowed for easier annotation, and the other additions to ISO-TimeML made up for the difference in how the different expressions were annotated.

Finally, since ISO-TimeML is, in fact, an international standard, some modifications had to be made to allow for the qualities of different languages besides English. For example, in Chinese, aspectual markers (words such as begins, ends, etc. in English) are not separate words, but rather are usually verbal suffixes. Also, some languages, such as Spanish, combine tense and aspect in a single verb form, rather than using modifying phrases. These and other cross-linguistic differences were accounted for in the ISO-TimeML standard.

6 The Resulting Model for ISO-TimeML

6.1 Structural Properties

The specification of ISO-TimeML consists of three components, mirroring the LAF distinction of abstract annotations and concrete representations: (1) an abstract syntax of ISO-TimeML annotations; (2) a format for representing these annotations in XML (a concrete syntax); and (3) a semantics of ISO-TimeML.

The abstract syntax of ISO-TimeML defines the set-theoretical structures that constitute the information about time and events that may be contained in annotations. The definition of the abstract syntax consists of two parts:

- (56) a. a specification of the elements from which these structures are built up, called a ‘conceptual inventory’; and
- b. a set of syntax rules which describe the possible combinations of these elements into pairs, triples, and other set-theoretic structures, called ‘annotation structures’.

What these combinations mean, i.e. which information they capture, is specified by the semantics associated with the abstract syntax.

The concrete syntax consists of the specification of names for the various sets forming the conceptual vocabulary, plus a listing of specific named elements of

these sets, and a specification of how to represent ISO-TimeML annotation structures defined by the syntax rules of the abstract syntax mentioned above. A particular XML-based syntax for temporal annotation was defined in the TimeML effort [61, 64] and has been adopted by ISO-TimeML, with the addition of a few significant changes, in conformance with LAF and the abstract syntax, which we will discuss below. For the present discussion we will focus on three object types and four relation types, as shown below.

- (57) a. EVENT: those elements in a text that describe what is conventionally referred to as an *eventuality*. Syntactically, events are typically appear as inflected or uninflected verbs, nominals, and adjectival phrases.
- b. TIMEX3: those elements in a text what are explicit temporal expressions, such as times, dates, durations, and quantified temporal expressions.
- c. SIGNAL: those elements denoting a temporal relation between events or time expressions.
- (58) a. TLINK: a relation that establishes the ordering of an event or temporal interval relative to another event or interval;
- b. ALINK: a relation that establishes an aspectual relationship between two events;
- c. SLINK: a relation that introduces a semantically subordinating context, such as that introduced by modality or reporting predicates;
- d. MLINK: a relation that establishes a measuring relation between a temporal expression and the event it measures.

The final component of ISO-TimeML consists of a specified semantic interpretation of the XML representations provided by the concrete syntax. There are currently two semantic fragments: one using Interval Temporal Logic, a first-order logic for reasoning about time [56]; the other one uses a DRT-like, event-based semantics for the abstract ISO-TimeML syntax [9]. This semantics is developed in the form of a mapping of entity structures and link structures into mini-DRSs (discourse representation structures [30]) and their merging into a single DRS.

6.2 Stand-Off Annotation

One of the most significant structural changes introduced by the ISO-TimeML specification is the move from in-line to stand-off annotation. This is in accordance with the general methodology to create interoperable annotation languages that do not modify the text being annotated.

ISO-TimeML conforms to the following three ISO standards: ISO 24610-1:2006 FSR (jointly developed with the TEI Consortium), ISO DIS 24611 MAF, and ISO DIS 24612 LAF. A proper management of stand-off annotation requires dealing with identifiers (xml:id) and pointers in conformance to most recent XML technologies and articulate these mechanisms with the XML elements provided by the other ISO standards for linguistic annotation. For instance, MAF specifies how a text

is segmented into tokens and how these tokens are represented in XML (element <token>)). In turn, ISO-TimeML annotations may point to such tokens as illustrated in the example sentence below.

(59) Mia visited Seoul to look me up yesterday.

This data is now segmented into word forms, as follows:

(60) TOKENIZATION:

```
<maf xmlns:"http://www.iso.org/maf">
  <seg type="token" xml:id="token1">Mia</seg>
  <seg type="token" xml:id="token2">visited
    </seg>
  <seg type="token" xml:id="token3">Seoul
    </seg>
  <seg type="token" xml:id="token4">to</seg>
  <seg type="token" xml:id="token5">look
    </seg>
  <seg type="token" xml:id="token6">me</seg>
  <seg type="token" xml:id="token7">up</seg>
  <seg type="token" xml:id="token8">yesterday
    </seg>
</maf>
```

As is specified in LAF, this inline segmentation may also be replaced by an offline identification of tokens through spans based, for instance, on character shifts: e.g., <seg ... form="Mia"/> is replaced by <seg ... from 0 to 3/>. Note here that the complex verb “looked ... up” is treated as a single word segment, consisting of two discontinuous tokens, “looked” and “up”. On the basis of the segmented text, ISO-TimeML can now annotate the given text in a stand-off manner, as represented below:

(61) STAND-OFF ANNOTATION:

```
<isoTimeML
  xmlns:"http://www.iso.org/isoTimeML">
  <TIMEX3 xml:id="t0" type="DATE"
    value="2009-10-20"
    functionInDocument="CREATION_TIME" />
  <EVENT xml:id="e1"
    target="#token2"
```

```

        class="OCCURRENCE" tense="PAST" />
<EVENT xml:id="e2"
  target="#token5 #token7"
  class="OCCURRENCE"
  tense="NONE" vForm="INFINITIVE" />
<TIMEX3 xml:id="t1" type="DATE"
  value="2009-10-19" />
<TLINK eventID="#e1" relatedToTime="#t0"
  relType="BEFORE" />
<TLINK eventID="#e1" relatedToTime="t1"
  relType="ON_OR_BEFORE" />
<TLINK eventID="#e2" relatedToTime="#t1"
  relType="IS_INCLUDED" />
</isoTimeML> <tei-isoFSR xmlns:
  "http://www.iso.org/tei-isoFSR">
<fs xml:id="t0">
  <f name="Type" value="2009-10-20" />
</fs> </tei-isoFSR>

```

Note that the temporal expression “yesterday” is interpreted as referring to the date “2009-10-19” on the assumption that the creation time for the text is 2009-10-20. Further, the event time of Mia’s visiting Seoul is understood as taking place in the past, “yesterday” or earlier.

6.3 Representing Relational Constraints

As mentioned above in (55), when modeling the behavior of events and temporal entities, there are essentially three characteristics that must be accounted for within an interpreted specification language such as ISO-TimeML: order, measurement, and quantity. Here we discuss how ISO-TimeML addresses each of these issues. While temporal ordering relations were adequately accounted for in TimeML, both measuring and quantifying temporal entities were issues that were not handled in a very transparent manner. ISO-TimeML has remedied this, as discussed below.

Regarding the first issue, that of ordering events and times, ISO-TimeML distinguishes the same domains as TimeML, over which ordering relations are performed.

- (62)
- a. A relation between two events;
 - b. A relation between two times;
 - c. A relation between a time and an event.

Examples of these three types of relations are shown in (63) below, where (63a) shows an ordering between two events, (63b) between two times, and (63c) an ordering of an event relative to a time.

- (63) a. John [taught]_{e1} before Mary [arrived]_{e2}.
 b. [the first Wednesday]_{t1} after [today]_{t2}
 c. John [taught]_{e1} on [Tuesday]_{t1}.

Currently, the ISO-TimeML framework deals with order by assuming a calculus of interval relations, that of Allen's interval algebra, adopted for TimeML.

- (64) a. before (b), after (bi);
 b. overlap (o), overlappedBy (oi);
 c. start (s), startedBy (si);
 d. finish (f), finishedBy (fi);
 e. during (d), contains (di);
 f. meet (m), metBy (mi);
 g. equality (eq).

The TLINK relation specifies the particular temporal ordering or anchoring of event predicates interpreted as intervals. This is described in detail in [64]. Briefly, we assume a function, τ , that inputs an event individual and returns the temporal trace of that event as an interval. The interpretation of the TLINK associated with the annotation of (63a), as shown in (65),

(65) <TLINK evID="e1" relToEvent="e2" sigID="s1" relType="BEFORE"/>

results in the event-event ordering in (66).

- (66) a. $teach = e_1, arrive = e_2$
 b. $\exists e_1 \exists e_2 [teach(e_1) \wedge arrive(e_2) \wedge \tau(e_1) < \tau(e_2)]$

Similarly, the time-time ordering in (63b) involves two TIMEX3 expressions, and a TLINK relation, as illustrated in (67).

(67) <TLINK evID="t1" relToTime="t2" sigID="s1" relType="AFTER"/>

The interpretation of this annotation is shown in (68).

- (68) a. $Wednesday = t_1, today = t_2$
 b. $\exists t_1 \exists t_2 [Wednesday(t_1) \wedge today(t_2) \wedge t_1 > t_2]$

Finally, the event-time relation illustrated in (63c) involves the annotation in (69), along with the interpretation given in (70).

(69) <TLINK evID="e1" relToTime="t2" sigID="s1" relType="IS_INCLUDED"/>

- (70) a. $teach = e_1, tuesday = t_2$

$$\text{b. } \exists e_1 \exists t_2 [\text{teach}(e_1) \wedge \text{tuesday}(t_2) \wedge \tau(e_1) \subseteq t_2]$$

In the discussion that follows, we will indicate how measure and quantity can be represented within ISO-TimeML.

6.4 Measuring Events

Another significant change introduced by ISO-TimeML is in the treatment of temporal durations. The TimeML DURATION type is based on the TIMEX2 treatment of durations, which is interpreted as a contiguous temporal interval. Consider, for example the examples shown below:

- (71) a. John slept for 2 h.
b. a three-day vacation

It was assumed that the interpretation in such event readings situated the event completely within a specific and named interval. For this reason, it was thought adequate to treat such cases with a TLINK relation; namely, the SIMULTANEOUS relType, as shown below:

```
<EVENT id="e1" pred="SLEEP"/>
<TIMEX3 id="t1" type="DURATION" value="P2H"/>
<TLINK eventID="e1" relatedToTime="t1" relType="SIMULTANEOUS"/>
```

This is inadequate, however, on two accounts. First, it is descriptively incomplete, in that this is not always the desired interpretation for a duration phrase. For example, consider the sentence below.

- (72) John taught for three hours on Tuesday.

In this case, the interpretation is ambiguous. Did John teach without stopping for three hours sometime during the day or did he teach for an hour, take a break, teach again, and so forth? Either interpretation is possible, so it would be incorrect to commit the interpretation to the contiguous (convex hull) interval reading. The second problem with this treatment is that it fails to characterize the temporal expression as a measurement of the event, as expressed in the abstract syntax for the language (as mentioned above).

To deal with this problem, ISO-TimeML reifies the role that certain expressions in the language play in measuring over a domain; that is, a new link is introduced for measuring out events, called MLINK, with the inherent relation type of MEASURE. A temporal expression such as *3 h* is expressed as a TIMEX3 of type DURATION, with the interpretation of a “time amount” [9]. This can be used in either non-contiguous or contiguous interpretations. A measure is equal to the sum of all times

that add up the desired period of time (ex. $P3H = \forall i[\Sigma i = P3H]$). This reflects more transparently the abstract syntax specified within ISO-TimeML, where the distinction is made between an interval and the measure of an interval. The annotation fragment is illustrated below.

```

%%% space deleted
<EVENT id="e1" pred="TEACH"/>
<TIMEX3 id="t2" type="DURATION" value="P3H"/>
%%% carriage return deleted
<MLINK eventID="e1" relatedToTime="t2" relType="MEASUREMENT"/>

```

Formally, we assume that a measure function, μ , such as introduced in [5], can be used interpret this relation, as represented below. The details of this proposal are more fully presented in [9].

- (73) a. $teach = e_1, tuesday = t_2, m = 3\ h$
 b. $\exists e_1 \exists t_2 \exists v [teach(e_1) \wedge \mu(\tau(e_1)) = v \wedge v = 3_h \wedge tuesday(t_2) \wedge \tau(e_1) \subseteq t_2]$

6.5 Counting Events

Anchoring and ordering relations in ISO-TimeML intrinsically quantify the event participating in the relation. But as has been pointed out, there is no clear way to embed an event within a temporal quantifier expression [9, 56]. Consider again the sentence mentioned above:

- (74) John taught on Tuesday.

In TimeML, the translation of the distinct XML elements is given below:

- (75) a. EVENT tag introduces a quantified event expression $\implies \exists e_1[teach(e_1)]$;
 b. TIMEX3 tag introduces the temporal expression $\implies \exists t_2[tuesday(t_2)]$;
 c. TLINK introduces the ordering relation $\implies \lambda y \lambda x [\tau(x) \subseteq y]$.

Assuming approaches to the semantics of TimeML as taken in [34, 56], the resulting semantics of the sentence is a conjunction of these relations:

- (76) $\exists e_1 \exists t_2 [teach(e_1) \wedge tuesday(t_2) \wedge \tau(e_1) \subseteq t_2]$

Now, what happens if we have a quantified temporal expression, such as *every Tuesday* as in (77)?

(77) John taught every Tuesday in November.

In TimeML, it is clear how to annotate such an expression. The TIMEX3 *every Tuesday* will be identified as `type = 'SET'`, with the attribute `quant = "every"`. As before, the translation between the distinct elements in this sentence would be given as follows:

- (78) a. EVENT tag introduces a quantified event expression $\implies \exists e_1[teach(e_1)];$
 b. TIMEX3 tag introduces the temporal expression $\implies \exists t_1[tuesday(t_1)];$
 c. TIMEX3 tag introduces the temporal expression $\implies \exists t_2[november(t_2)];$
 d. TLINK introduces an ordering relation $\implies \lambda y \lambda x [\tau(x) \subseteq y];$

But this does not give us the right scope and interpretation. This results in an interpretation where one event of teaching occurs over every Tuesday in November. [9] explore the option of explicitly marking the distributive property [5] of the quantification in the annotation directly. This would allow us to then scope the temporal expression over the event predicate, as illustrated below in (79).

- (79) $\forall t_1 \exists e_1 \exists t_2 [(Tuesday(t_1) \wedge November(t_2) \wedge t_1 \subseteq t_2) \rightarrow (teach(e_1) \wedge \tau(e_1) \subseteq t_1)]$

To account for the behavior of quantified temporal expressions as generalized quantifiers, ISO-TimeML introduces a attribute within the quantified term that indicates what it takes scope over. This attribute is called SCOPES and it establishes a scoping relation between the expression calling it and the argument to the attribute. For example, the new TIMEX3 annotation for *every Tuesday* is shown below in (80).

- (80) `< TIMEX3 id = t1 type = SET quant = "every" scopes = e1 >`

This introduces the relation $scopes(t_1, e_1)$, which, together with the lexical semantics of the quant value for “every”, allows us to identify the proper scope, as shown in (79).

Note that the “scoping” attribute, SCOPES, is also necessary for quantified nominal event expressions, such as *every lecture* in (81) below.

- (81) Mary read during every lecture.

This is because they behave in a similar fashion to quantified temporal expressions, relative to the scope of the matrix verbal event in the sentence. The annotation for this example is shown below in (82), along with the appropriate semantic interpretation in (83).

- (82) `< EVENT id = e1 pred = "READ" >`
`< EVENT id = e2 pred = "LECTURE" type = SET quant = "every" scopes = e1 >`

$$(83) \quad \forall e_2 \exists e_1 [lecture(e_2) \rightarrow [read(e_1) \wedge \tau(e_1) \subseteq \tau(e_2)]]$$

Some of the details of how quantification is best expressed in the annotation specification are still being worked out; the abstract syntax of ISO-TimeML, however, does allow us to express such scope relations in the concrete syntax directly.

6.6 Specifying the Metamodel

Regarding the temporal information in a document, a distinction can be made between (1) the temporal metadata, regarding when the document was created, published, distributed, received, revised, etc., and most importantly (2) the temporal properties of the events and situations that are mentioned in the document. The former type of information is associated with the document as a whole; information of the latter type will be associated in annotations with text segments in the primary data ('source text' in Fig. 5).

Since semantic annotations typically occur at a relatively high level in a layered annotation structure, they do not refer directly to segments in the primary data, but rather refer to structures in other annotation layers, such as the output of a tokenizer or a syntactic parser. Such structures do define a stretch of primary data, and contain additional information such as morphosyntactic features and references to entries in a lexicon or an ontology. The generic term *markable* is used to refer to the structures that the annotations are associated with. The metamodel shown in Fig. 5 reflects this situation in locating 'markables' between 'source text' and the the various kinds of entities and relations that make up an actual ISO-TimeML annotation.

Markables are derived from documents, which will have certain metadata that are particularly important for the interpretation of temporal annotations. For interpreting the tenses of verb forms and adverbial temporal deixis in a text, for instance, one must know when the text was produced. This will often be defined by the document creation time, and more precisely by the combination of a creation time and a creation location, since the latter defines the time zone within which the creation time is precisely defined. The time and place of the document creation may be the same for all the markables associated with the source text, but it may also happen that the text introduces other times and places relative to which the annotations of the markables should be understood. A reasonable strategy would seem to be to assume that each markable has a time and place (or time zone), which by default is that of the document in which it is defined.

A markable may refer to more than one, related event, as in *She started to laugh* (two aspectually related events); *John drove to Boston after the concert* (two temporally related events); or *Will you attend the meeting on Tuesday?* (one event having a subordination relation to another). For expressing such relations the metamodel in Fig. 5 includes the corresponding classes of relations, showing up as inter-event links. The same relations, discussed in Sect. 4.2.2, that may hold between temporal intervals may also be used as temporal relations between events, hence they show up at both places in the metamodel.

Temporal objects and relations have been studied from logical and ontological points of view; well-known studies include [1,24,25,57]; see also the collection of papers in [44]. The most common view of time, which underlies most natural languages, is that time is an unbounded linear space running from a metaphorical ‘beginning of time’ at minus infinity to an equally metaphorical ‘end of time’ at plus infinity. This linear space can be represented as a straight line, the points of which correspond to moments in time; following [25] we will also use the term ‘instant’ to refer to points of time.

From a mathematical point of view, the points on the time line are line segments of infinitesimally small size, corresponding to the intuition that a moment in time can in principle be determined with any precision that one may wish. A time zone, like Greenwich Mean Time (GMT) can be seen as a way of segmenting the time line into named segments of particular lengths, such as (calendar) years, months, days, hours, and minutes. Accordingly, time zones show up in the metamodel in Fig. 4 as functions that map a calendar year (‘2016’), or a combination of a calendar year and a calendar month (‘May 2016’), or a date (‘May 28, 2016’), or a date plus a clock time (‘May 28, 2016, 12.30 p.m.’) onto a temporal interval (in the latter case, an instant).

Natural languages offer speakers the possibility to express themselves as if something occurs at a precise instant (like *I will call you at twelve o’clock*), although every activity, process, achievement, or other type of event that occurs in reality or in someone’s mind requires more than an infinitesimally short time. Since instants are formally a special case of intervals, a consistent approach to modeling the time that an event occurs is to always use intervals, where it may happen that the interval associated with a particular event is regarded as an instant, having zero length. This is reflected in the metamodel presented in Fig. 5, which uniformly relates eventualities to temporal intervals.

The length of an interval can also be mentioned without being associated with an eventuality, as in *The time difference with Hong Kong is six hours*, where it indicates a temporal distance rather than the duration of an event; it is the temporal equivalent of spatial distance (*six kilometres*). Temporal distances and event durations are amounts of time which, as outlined in Sect. 6.4, are regarded in ISO-TimeML as a kind of temporal objects which can be related to an eventuality by means of the MLINK link. Conceptually, an amount of time is defined as an equivalence relation of sets of pairs consisting of a non-negative numerical value and a unit of measurement (where the equivalence relation is defined by a conversion function relating different units of measurement, such as 1 day = 24 h; 1 h = 60 min, etc.. For more details see [5]). This view is reflected in example (74b) by the use of the temporal object *3_hour* and in the metamodel in Fig. 5 where “amounts of time” figure as a distinct type of temporal object.

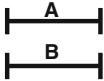
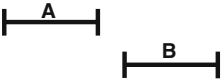
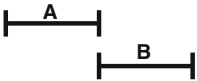
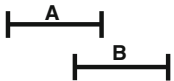

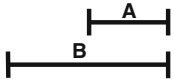
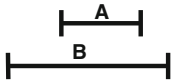
	A EQUALS B
	A is BEFORE B; B is AFTER A
	A MEETS B; B is MET BY A
	A OVERLAPS B; B is OVERLAPPED BY A
	A STARTS B; B is STARTED BY A
	A FINISHES B; B is FINISHED BY A
	A is DURING B; B CONTAINS A

Fig. 4 The interval relations as defined by Allen [1]

6.7 Abstract Syntax

The abstract syntax of ISO-TimeML defines the set-theoretical structures (like pairs and triples) called *annotation structures* that form the information about time and events that may be contained in annotations. As mentioned above (Sect. 6.1) the definition of the abstract syntax consists of two parts: (a) a specification of the concepts from which annotation structures are built up, called a ‘conceptual inventory’; and (b) a set of syntax rules which describe the possible combinations of these elements. What these combinations mean, i.e. which information they capture, is specified by the semantics associated with the abstract syntax.

6.7.1 Conceptual Inventory

The conceptual inventory of concepts used to build ISO-TimeML annotation structures fall into five categories, all formed by finite sets, plus the concepts of real number and natural number. Natural numbers are needed for capturing the

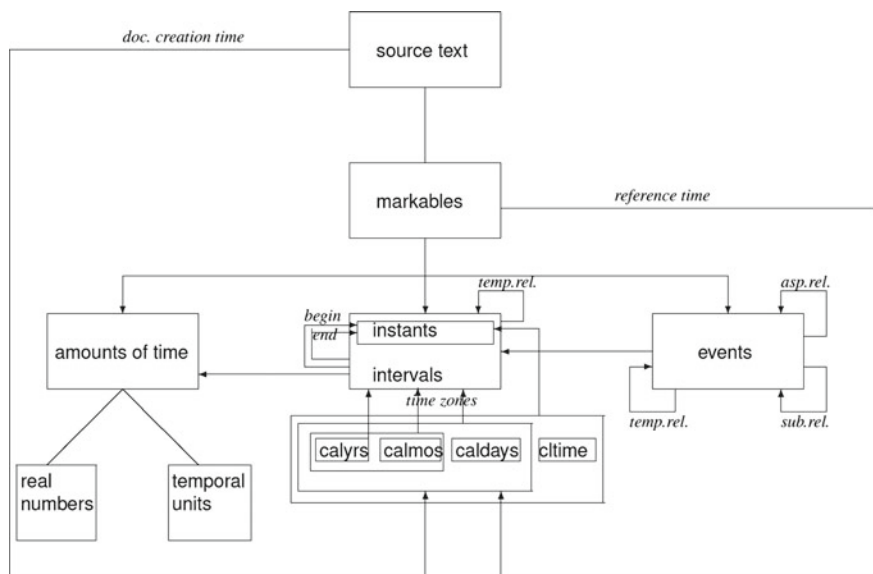


Fig. 5 ISO-TimeML metamodel

information expressed in English by *twice* and *three times*; real numbers are needed for cases such as *two and a half hours*. More specifically, the ISO-TimeML conceptual inventory is formed by:

- sets of elements called ‘event classes’, ‘tenses’, ‘aspects’, ‘polarities’, and ‘set-theoretic types’;
- finite sets of elements called ‘temporal relations’, ‘duration relations’, ‘numerical relations’, ‘event subordination relations’, and ‘aspectual relations’;
- a set of elements called ‘time zones’;
- sets of elements called ‘calendar years’, ‘calendar months’, ‘calendar day numbers’, ‘clock times’ (natural numbers ranging from 0000 to 0059; from 0100 to 0159; ... from 2300 to 2400);
- a set of ‘temporal units’.

6.7.2 Syntax Rules

Annotation structures consist of two kinds of smaller structures, called entity structures and link structures. An entity structure contains semantic information about a segment of source text, while a link structure describes a semantic relation between segments of source text by means of links between entity structures.

An entity structure is a pair $\langle m, \alpha \rangle$ associating the semantic information α with the segment of source text identified by the markable m . A link structure in

ISO-TimeML⁶ is a triple $\langle \epsilon_1, \epsilon_2, R \rangle$ consisting of two entity structures and a relation. More formally, an annotation structure is a pair consisting of a set of entity structures and a set of link structures that link the entity structures together through temporal and inter-event relations.

Entity structures:

There are five types of entity structures, containing information about (1) events; (2–4) temporal objects (intervals, instants, and amounts of time); and (5) explicit temporal relations (as for instance expressed in English by temporal prepositions). The component α in an entity structure $\langle m, \alpha \rangle$ can be one of the following five structures:

1. An *event structure* is a 7-tuple $\langle C, T, A, \Sigma, N, P_N, V \rangle$ where C is a member of the set of event classes; T and A are a tense and an aspect, respectively; Σ is a set-theoretical type (such as *individual object* or *set of individual objects*); N is a natural number (e.g. the number 2 for dealing with such examples as “*John kissed Mary twice*”); P_N is an amount of time (such as two and a half hours, for such examples as “*John called Mary twice every two and a half hours*”), and V is a veracity (claimed truth or falsity, corresponding to positive or negative polarity in natural language).
2. The following set-theoretical structures are *interval structures*:
 - a. An *instant structure*: either a triple $\langle \text{time zone}, \text{date}, \text{clocktime} \rangle$, where a *date* is a triple consisting of a calendar year, a calendar month, and a calendar day number; or a triple $\langle \text{time-amount structure}, \text{instant structure}, \text{temporal relation} \rangle$ (“*half an hour before midnight*”);
 - b. a pair $\langle t_1, t_2 \rangle$ of two interval structures, corresponding to the beginning and end points of an interval (“*nine to five*”);
 - c. a triple $\langle \text{time-amount structure}, \text{interval structure}, \text{temporal relation structure} \rangle$ (“*three weeks before Christmas*”; “*two years from today*”);
 - d. a triple $\langle t_1, t_2, R_d \rangle$ where t_1 and t_2 are interval structures, and where R_d is a duration relation (“*from '92 till 95*”; “*from 1882 through 1995*”).
3. A *time-amount structure* is a pair $\langle n, u \rangle$, where n is a real number and u a temporal unit, or a triple $\langle R_n, n, u \rangle$, where R_n is a numerical relation and n and u as before (“*six years*”; “*more than five minutes*”).
4. A *temporal relation structure* is just a temporal relation.

Link structures:

There are five types of link structures in ISO-TimeML: for temporal *anchoring* of events in time; for temporal *ordering of events and/or intervals (including instants)*

⁶This is because all temporal relations in ISO-TimeML are binary.

relative to each other; for *measuring* the length of an interval; for *subordination relations* between events, and for *aspectual relations* between events.

1. A *temporal anchoring link structure* is a triple:
 $\langle \text{event structure}, \text{interval structure}, \text{temporal anchoring relation} \rangle$;
2. A *temporal relation link structure* is a triple
 $\langle \text{event structure}, \text{event structure}, \text{temporal relation} \rangle$, or a triple
 $\langle \text{interval structure}, \text{interval structure}, \text{temporal relation} \rangle$
3. A *time measurement link structure* is a pair $\langle \text{event structure}, \text{time} - \text{amount structure} \rangle$ or a pair $\langle \text{interval structure}, \text{time} - \text{amount structure} \rangle$;
4. A *subordination link structure* is a triple $\langle \text{event structure}, \text{event structure}, \text{subordination relation} \rangle$;
5. An *aspectual link structure* is a triple $\langle \text{event structure}, \text{event structure}, \text{aspectual relation} \rangle$.

It should be noted that the abstract syntax distinguishes five types of link structure, while the ISO-TimeML concrete syntax distinguishes only four types of link: TLINK, MLINK, SLINK and ALINK. This is because both temporal anchoring link structures and temporal relation link structures are represented by means of TLINKs. This is no problem, since the types of the arguments of a TLINK allows one to reconstruct the corresponding abstract link structures, hence this does not affect the ‘unambiguity’ of the representations (see Sect. 2.2).

Further, the event structures defined above may contain two elements that are currently not part of the concrete syntax, namely a set-theoretical ‘signature’ and a cardinality. These elements have been included to be able to deal with certain forms of quantification that are currently not covered by ISO-TimeML (“*John kissed Mary twice*”; “*Everybody will die*”, with two possible scopings of events and individuals), but are planned to be included in a future extension, as discussed in [9].

7 Conclusion

In this chapter we have elaborated both the process and the methodology for converting the observations and insights of linguistic theories for specific language phenomena into models for developing language annotation specifications. We focused on a single domain, temporal information, in order to trace the development path from theory to model through the adoption of both the MAMA subcycle of MATTER and the CASCADES model. We demonstrated the necessary steps and decisions in this process by examining the creation, modification, and formalization of an actual working specification language, TimeML, and its ISO standardized form as ISO-TimeML.

The details of how the development of TimeML follows the methodology outlined in this chapter will not necessarily apply to all language annotation tasks. However,

because of the complexity of the linguistic phenomena associated with temporal information, and the extent of prior theoretical modeling in the domain, a carefully examined case study helps to demonstrate the role of the annotation development models, MATTER and CASCADES, in the creation of a specification. The methodology outlined here exemplifies what we believe is best practices applied to specification design and creation for natural language phenomena. This establishes the foundation and specificational infrastructure for the topic covered in the next chapter “[Designing annotation schemes: From model to representation](#)”. In this chapter, Ide et al. focus on the actual *physical representation* of the information being annotated as modeled in the abstract syntax.

Acknowledgements We would like to express our thanks for the many people involved in the development of TimeML and ISO-TimeML. In particular, we would like to thanks Kiyong Lee, Jessica Moszkowicz, Roser Saurí, Marc Verhagen, Bran Boguraev, Bob Knippen, Inderjeet Mani, Graham Katz, Rob Gauzauskis, Andrea Setzer, Jerry Hobbs, Ian Pratt-Harman, Drago Radev, Tommaso Caselli, and members of the ISO community, including Nancy Ide, Alex Chengyu Fang, Rainer Osswald, Haihua Pan, Yuzhen Cui, Haihua Pan, Manigo Kit, and Amanda Schiffrin.

References

1. Allen, J.: Towards a general theory of action and time. *Artif. Intell.* **23**, 123–154 (1984)
2. Baker, C., Fillmore, C., Cronin, B.: The structure of the Framenet database. *Int. J. Lexicogr.* **16**(3), 281–296 (2003)
3. Beavers, J.: Scalar complexity and the structure of events. In: Dölling, J., Heyde-Zybatow, T., Schäfer, M. (eds.) *Event Structures in Linguistic Form and Interpretation*, pp. 245–265. Mouton de Gruyter, Berlin (2008)
4. Boguraev, B., Pustejovsky, J., Ando, R., Verhagen, M.: Timebank evolution as a community resource for timeml parsing. *Lang. Resour. Eval.* **41**(1), 91–115 (2007)
5. Bunt, H.C.: *Mass Terms and Model-Theoretic Semantics*. Cambridge University Press, Cambridge (1985)
6. Bunt, H.: Semantic annotations as complementary to underspecified semantic representations. In: *Proceedings of the Eighth International Conference on Computational Semantics*, pp. 33–44. Association for Computational Linguistics (2009)
7. Bunt, H.: Introducing abstract syntax+ semantics in semantic annotation, and its consequences for the annotation of time and events. In: Lee, E., Yoong, A. (eds.) *Recent Trends in Language and Knowledge Processing*, pp. 157–204. Hankukmunhwasa, Seoul (2011)
8. Bunt, H.: On the principles of interoperable semantic annotation. In: *Proceedings of the 11th Joint ACL-ISO Workshop on Interoperable Semantic Annotation*, pp. 1–13 (2015)
9. Bunt, H., Pustejovsky, J.: Annotating temporal and event quantification. In: *Proceedings of 5th ISA Workshop* (2010)
10. Bunt, H., Fang, A.C., Ide, N., Webster, J.: A methodology for designing semantic annotation languages exploiting syntactic-semantic isomorphisms (2010)
11. Bunt, H., Alexandersson, J., Choe, J.W., Fang, A.C., Hasida, K., Petukhova, V., Popescu-Belis, A., Traum, D.R.: Iso 24617-2: a semantically-based standard for dialogue annotation. In: *LREC*, pp. 430–437. Citeseer (2012)

12. Carlson, G.N.: Reference to kinds in English. Ph.D. thesis, Linguistics Department, University of Massachusetts, Amherst, Massachusetts (1977)
13. Chierchia, G.: Structured meanings, thematic roles and control. In: *Properties, Types and Meaning*, pp. 131–166. Springer, Berlin (1989)
14. Chinchor, N., Robinson, P.: MUC-7 named entity task definition. In: *Proceedings of the 7th Conference on Message Understanding*, p. 29 (1997)
15. Comrie, B.: *Tense*. Cambridge University Press, Cambridge (1985)
16. Davidson, D.: The logical form of action sentences. In: Rescher, N. (ed.) *The Logic of Decision and Action*, pp. 81–95. Pittsburgh Press, Pittsburgh (1967)
17. Diesing, M.: Bare plural subjects and the derivation of logical representations. *Linguist. Inq.* **23**, 353–380 (1992)
18. Dowty, D.R.: *Word Meaning and Montague Grammar: The Semantics of Verbs and Times in Generative Semantics and in Montague's PTQ*, vol. 7. Springer Science & Business Media, Berlin (1979)
19. Dowty, D.: On the semantic content of the notion of thematic role. *Prop. Types Mean.* **2**, 69–130 (1989)
20. Dowty, D.: Thematic proto-roles and argument selection. *Language* **67**, 547–619 (1991)
21. Grimshaw, J.: *Argument Structure*. MIT Press, Cambridge (1990)
22. Hay, J., Kennedy, C., Levin, B.: Scalar structure underlies telicity in 'degree achievements'. In: Matthews, T., Strolovitch, D. (eds.) *Proceedings of Semantics and Linguistic Theory IX*, pp. 127–144. Cornell University, Ithaca (1999)
23. Higginbotham, J.: On semantics. *Linguist. Inq.* **16**, 547–593 (1985)
24. Hobbs, J.R., Pustejovsky, J.: Annotating and reasoning about time and events. In: Doherty, P., McCarthy, J., Williams, M.A. (eds.) *Working Papers of the 2003 AAAI Spring Symposium on Logical Formalization of Commonsense Reasoning*, pp. 74–82. AAAI Press, Menlo Park (2003)
25. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. *ACM Trans. Asian Lang. Inf. Process. (TALIP)* **3**(1), 66–85 (2004)
26. Ide, N., Romary, L.: Outline of the international standard linguistic annotation framework. In: *Proceedings of the ACL 2003 Workshop on Linguistic Annotation: Getting the Model Right*, vol. 19, pp. 1–5. Association for Computational Linguistics (2003)
27. Ide, N., Romary, L.: International standard for a linguistic annotation framework. *Nat. Lang. Eng.* **10**(3–4), 211–225 (2004)
28. Ide, N., Suderman, K.: The linguistic annotation framework: a standard for annotation interchange and merging. *Lang. Resour. Eval.* **48**(3), 395–418 (2014)
29. Jäger, G.: Topic-comment structure and the contrast between stage level and individual level predicates. *J. Semant.* **18**(2), 83–126 (2001)
30. Kamp, H., Reyle, U.: *From Discourse to Logic; Introduction to the Model-theoretic Semantics of Natural Language*. Springer, Berlin (1993)
31. Karttunen, L.: Implicative verbs. *Language* **47**, 340–358 (1971)
32. Karttunen, L.: Some observations on factivity. *Res. Lang. Soc. Interact.* **4**(1), 55–69 (1971)
33. Karttunen, L., Zaenen, A.: Veridicity. In: *Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik* (2005)
34. Katz, G.: Towards a denotational semantics for TimeML. In: *Annotating, Extracting and Reasoning about Time and Events*, pp. 88–106. Springer, Berlin (2007)
35. Kennedy, C., Levin, B.: Measure of change: the adjectival core of degree achievements. *Adjectives and adverbs: Syntax, semantics and discourse* pp. 156–182. Oxford University Press, Oxford (2008)
36. Kiparsky, P., Kiparsky, C.: Fact. In: *Progress in Linguistics*, pp. 143–173. Mouton, The Hague (1971)

37. Kipper, K.: Verbnets: a broad-coverage, comprehensive verb lexicon. Ph.D. dissertation, University of Pennsylvania, PA (2005). <http://repository.upenn.edu/dissertations/AAI3179808/>
38. Krifka, M.: Thematic relations as links between nominal reference and temporal constitution. *Lex. Matters* **2953**, 30–52 (1992)
39. Krifka, M.: The origins of telicity. In: Rothstein, S. (ed.) *Events and Grammar*. Kluwer, Dordrecht (1998)
40. Levin, B., Hovav Rappaport, M.: *Argument Realization*. Cambridge University Press, Cambridge (2005)
41. Lin, J.W.: Time in a language without tense: the case of chinese. *J. Semant.* **23**(1), 1–53 (2006)
42. Mani, I., Wilson, G.: Robust temporal processing of news. In: *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL2000)*, pp. 69–76. New Brunswick (2000)
43. Mani, I., Wilson, G., Sundheim, B., Ferro, L.: Guidelines for annotating temporal information. In: *Proceedings of HLT 2001, First International Conference on Human Language Technology Research* (2001)
44. Mani, I., Pustejovsky, J., Gaizauskas, R.: *The Language of Time: A Reader*. Oxford University Press, Oxford (2005)
45. Mani, I., Verhagen, M., Wellner, B., Lee, C.M., Pustejovsky, J.: Machine learning of temporal relations. In: *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 753–760. Association for Computational Linguistics, Sydney (2006). <http://www.aclweb.org/anthology/P/P06/P06-1095>
46. Mani, I., Wellner, B., Verhagen, M., Pustejovsky, J.: Three approaches to learning TLINKs in timeml. Technical Report CS-07-268, Brandeis University, Waltham, United States (2007)
47. Manna, Z., Pnueli, A.: *Temporal Verification of Reactive Systems: Safty*. Springer, Berlin (1995)
48. McCarthy, J.: Situations, actions, and causal laws. Technical Report, DTIC Document (1963)
49. McCarthy, J., Hayes, P.J.: Some philosophical problems from the standpoint of artificial intelligence. *Readings in artificial intelligence* pp. 431–450 (1969)
50. Moens, M., Steedman, M.: Temporal ontology and temporal reference. *Comput. Linguist.* **14**(2), 15–28 (1988)
51. MUC: *Proceedings of the Sixth Message Understanding Conference (MUC-6)*. Morgan Kaufmann, California (1995)
52. Naumann, R.: Aspects of changes: a dynamic event semantics. *J. Semant.* **18**(1), 27–81 (2001)
53. Palmer, M., Gildea, D., Kingsbury, P.: The proposition bank: an annotated corpus of semantic roles. *Comput. Linguist.* **31**(1), 71–106 (2003)
54. Parsons, T.: *Events in the Semantics of English*, vol. 5. MIT Press, Cambridge (1990)
55. Partee, B.H.: Some structural analogies between tenses and pronouns in english. *J. Philos.* **70**(18), 601–609 (1973)
56. Pratt-Hartmann, I.: From TimeML to Interval temporal logic. In: *Proceedings of the Seventh International Workshop on Computational Semantics*, pp. 166–180 (2007)
57. Prior, A.N.: *Past, Present and Future*, vol. 154. Clarendon Press, Oxford (1967)
58. Pustejovsky, J.: The geometry of events. In: Tenny, C. (ed.) *Studies in Generative Approaches to Aspect*. Lexicon Project Working Papers vol. 24. MIT, Cambridge (1988)
59. Pustejovsky, J.: The syntax of event structure. *Cognition* **41**(1), 47–81 (1991)
60. Pustejovsky, J., Stubbs, A.: *Natural Language Annotation for Machine Learning*. O'Reilly Media, Inc., USA (2012)
61. Pustejovsky, J., Castano, J., Ingria, R., Saurí, R., Gaizauskas, R., Setzer, A., Katz, G.: TimeML: robust specification of event and temporal expressions in text. In: *IWCS-5, Fifth International Workshop on Computational Semantics* (2003). <http://www.timeml.org>

62. Pustejovsky, J., Hanks, P., Saurí, R., See, A., Gaizauskas, R., Setzer, A., Radev, D., Sundheim, B., Day, D., Ferro, L., Lazo, M.: The TimeBank corpus. In: *Proceedings of Corpus Linguistics*, pp. 647–656 (2003)
63. Pustejovsky, J., Ingria, B., Sauri, R., Castano, J., Littman, J., Gaizauskas, R., Setzer, A., Katz, G., Mani, I.: The Specification Language TimeML. *The Language of Time: A Reader*. Oxford University Press, Oxford (2004)
64. Pustejovsky, J., Knippen, R., Littman, J., Saurí, R.: Temporal and event information in natural language text. *Lang. Resour. Eval.* **39**, 123–164 (2005)
65. Pustejovsky, J., Littman, J., Sauri, R.: Arguments in TimeML: events and entities. In: Katz, F., Pustejovsky, J., Schilder, F. (eds.) *Annotating, Extracting and Reasoning about Time and Events*, vol. 4795, pp. 107–126. Springer, Berlin (2007)
66. Pustejovsky, J., Lee, K., Bunt, H., Romary, L.: Iso-TimeML: an international standard for semantic annotation. In: *LREC* (2010)
67. Rappaport Hovav, M., Levin, B.: An event structure account of english resultatives. *Language* **77**(4), 766–797 (2001)
68. Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C., Scheffczyk, J.: *FrameNet II: Extended Theory and Practice* (2006). <http://framenet.icsi.berkeley.edu/framenet>
69. Setzer, A.: Temporal information in newswire articles: an annotation scheme and corpus study. Ph.D. thesis, University of Sheffield, UK (2001)
70. Sundheim, B.M.: Overview of results of the MUC-6 evaluation. In: *Proceedings of a Workshop on Held at Vienna, Virginia: May 6–8, 1996, TIPSTER '96*, pp. 423–442. Association for Computational Linguistics (1996)
71. Tenny, C.: The aspectual interface hypothesis. 31. *Lexicon Project*, Center for Cognitive Science, MIT (1989)
72. Van Lambalgen, M., Hamm, F.: *The Proper Treatment of Events*, vol. 6. Wiley, New York (2008)
73. Vendler, Z.: Verbs and times. *Philos. Rev.* **66**, 143–160 (1957)
74. Wiebe, J., Wilson, T., Cardie, C.: Annotating expressions of opinions and emotions in language. *Lang. Resour. Eval.* **39**(2–3), 165–210 (2005)
75. Wilson, G., Mani, I., Sundheim, B., Ferro, L.: A multilingual approach to annotating and extracting temporal information. In: *Proceedings of the Workshop on Temporal and Spatial Information Processing*. vol. 13, p. 12. Association for Computational Linguistics (2001)

Handbook of Linguistic Annotation

Ide, N.; Pustejovsky, J. (Eds.)

2017, IX, 1459 p. 264 illus. In 2 volumes, not available separately., Hardcover

ISBN: 978-94-024-0879-9