Capturing Telic/Atelic Temporal Data Semantics: Generalizing Conventional Conceptual Models

Vijay Khatri, Sudha Ram, Richard T. Snodgrass, and Paolo Terenziani

Abstract—Time provides context for all our experiences, cognition, and coordinated collective action. Prior research in linguistics, artificial intelligence and temporal databases suggests the need to differentiate between temporal facts with goal-related semantics (i.e., telic) from those are intrinsically devoid of culmination (i.e., atelic). To differentiate between telic and atelic data semantics in conceptual database design, we propose an annotation-based temporal conceptual model that generalizes the semantics of a conventional conceptual model. Our temporal conceptual design approach involves: 1) capturing "what" semantics using a conventional conceptual model; 2) employing annotations to differentiate between telic and atelic data semantics that help capture "when" semantics; 3) specifying temporal constraints, specifically non-sequenced semantics, in the temporal data dictionary as metadata. Our proposed approach provides a mechanism to represent telic/atelic temporal semantics using temporal annotations. We also show how these semantics can be formally defined using constructs of the conventional conceptual model and axioms in first-order logic. Via what we refer to as the "semantics of composition," i.e., semantics implied by the interaction of annotations, we illustrate the logical consequences of representing telic/atelic data semantics during temporal conceptual design.

Index Terms—temporal database, conceptual modeling, data semantics, temporal conceptual model, database design.

1 Introduction

Many applications, e.g., health care (patient care), insurance (claims and accident histories), reservation systems and scientific data repositories, require capturing some aspect of time in organizing their information. Consequently, there is a recent call for research that focuses on "evolving and changing world state" [21]. Over the last two decades, the database community has devised many approaches to model the time validity of data: over 2000 papers on temporal databases have been published (see [73] for a cumulative bibliography). Recent temporal database research highlights the significance of distinguishing between facts with goal-related semantics (referred to as *telic*) from those that are devoid of culmination (referred to as *atelic*) [64, 65]. The telic/atelic data semantics—dating back to Aristotle [6]—is rooted in linguistics [68] and cognitive science [15]. In this paper, we suggest that the telic/atelic distinction is important in designing *temporal applications*, i.e., database applications that need to organize data by time.

Conceptual database design is widely recognized as an important step in the development of database applications [12, 26, 57] including the temporal applications described above. During conceptual database design, a *conceptual model* provides a notation and formalism that can be used to construct a high-level representation of the real world—referred to as a *conceptual schema*—independent of the implementation details. By enabling analysts to work at a higher level of abstraction [32], conceptual modeling—part of conceptual design— impacts the ability to meet users' requirements [12], affects integration with other systems [50], promotes understanding of the "real world" domain [27], and supports communication between analysts and users [56]. While "static" or "time agnostic" conceptual models, referred to as *conventional conceptual models* (e.g., the *E*ntity-*R*elationship (ER) model [20]), have been used successfully for the last three decades, recently there has been interest in incorporating time into the conceptual models [21]. Considering that temporal data is finding its way into traditional applications, e.g., retail, health care, insurance, there is a need for an overarching methodology that integrates design of temporal databases with conventional conceptual design. Thus, one of the requirements of the proposed approach that differentiates telic/atelic data semantics is that it should be compatible with existing general purpose methodologies [12, 20, 26].

Recent research suggests that temporal requirements need careful consideration during conceptual design (see, for example, [8, 11, 30, 37, 61]). As indicated in an excellent survey [31], several temporal conceptual models have been proposed in prior literature. Prior approaches to temporal conceptual modeling (for example, [11, 25, 29, 30, 37, 49, 74]) provide different levels of support for modeling temporal data semantics including timestamping constraints, evolution and transition constraints, and lifespan cardinality constraints [33]. *Timestamping constraints* distinguish between those elements of the schema that change over time from those that do not, *evolution and transition constraints* focus on migration of entities from one class to another, and *lifespan cardinality constraints* focus on cardinalities over the lifespan of entities. Prior approaches to temporal conceptual modeling have focused on different types of temporal constraints. For example, while $\mathcal{ER}_{\mathrm{VT}}$ [8, 11], TERC+ [74] focus on all three types of constraints, TimeER [29, 30] and ST-USM [36, 37] focus on semantics of temporal data as captured on a single schema and, therefore, focus on only timestamping and lifespan cardinality constraints. However, none of the prior approaches have focused on timestamping constraints that help distinguish between telic-atelic data semantics [39, 40] in a conceptual schema.

While prior research [64] has examined the implications of the telic/atelic dichotomy on the temporal aspects of the logical (specifically, *relational*) data model and algebra, this dichotomy has until now not been considered within the conceptual model. Other work [36, 37] has considered conceptual modeling for temporal and geospatial data and applied an annotation-based approach to the *U*nifying *S*emantic *M*odel (USM)[1] [52]—an extended version of the ER Model [20]—to propose the geo*S*patio-*T*emporal *U*nifying *S*emantic *M*odel (ST USM). That work likewise did not consider the telic/atelic dichotomy. By proposing an approach that *generalizes* the semantics of concepts in a conventional conceptual model for temporal applications, this paper extends prior research in conceptual design. To the best of our knowledge, this is the first work to incorporate the telic/atelic dichotomy in a conceptual model.

To delineate the scope of our work, we state our assumptions.

- In this research, we employ annotations to capture telic/atelic data semantics. The objective of this research is not syntax (e.g., textual annotations), but rather temporal data semantics that need to be captured during conceptual design. Prior research in temporal conceptual design has presented several different ways of rendering temporal data semantics via annotations. For example, temporal data semantics can be represented textually (see, for example, [37]) or graphically (see, for example, [74]); they can be represented in the schema (see, for example, [37]) or outside the schema (see, for example, [59]). However, the objective of this research is to explicate the telic/atelic data semantics and not delve into how "best" to render those data semantics as that is an important topic that has been considered elsewhere (see, for example, [41]).
- In our research on conceptual modeling, we also do not focus on how semantics are *physically* implemented, as that is an involved topic in its own right and has been considered elsewhere in the context of the relational model [64].
- Facts can interact with time in two orthogonal ways resulting in transaction time and valid time [60]. While *valid time* denotes when a fact is true in the real world and implies the storage of histories related to facts, *transaction time* links an object to the time it is current in the database and implies the storage of versions of a database object [33]. In this paper, we focus only on valid time because transaction time is intrinsically atelic.
- While database schema can evolve with time and *schema versioning* [53, 54] is an important area of research, we do not focus on schema versioning, in part because most schema versioning is focused on transaction time. Instead, we focus on the semantics of data, as captured in a single schema.
- In this work we differentiate between *data* and *query semantics*, and focus on the former. Note that data has its own semantics—independently of any query language and query operators—and that queries are merely an operational way of making such semantics explicit. However, data has semantics even if *no* query is asked. Because conceptual models provide a mechanism to capture data semantics, we focus on capturing (telic/atelic) data semantics.[2]

In summary, this paper exemplifies how the semantics of constructs in a conventional conceptual model can be generalized to define a temporal conceptual model. We focus on differentiating between telic and atelic data semantics during conceptual modeling via a temporal conceptual design approach that involves: 1) capturing "what" semantics using a conventional conceptual model; 2) employing annotations to differentiate telic/atelic data semantics that help capture "when" semantics; and 3) specifying temporal constraints, specifically non-sequenced semantics, in the temporal data dictionary as metadata. Our proposed approach provides a mechanism to represent telic/atelic temporal semantics using temporal annotations. We also show how these semantics can be formally defined using constructs of the conventional conceptual model and axioms in first-order logic. Via what we refer to as the "semantics of composition," i.e., semantics implied by the interaction of annotations, we illustrate the logical consequences of explicating telic/atelic data semantics during temporal conceptual design. In summary, much like prior research [7, 9], we focus on both formalization of constructs and logical implications that are associated with such formalization.

The rest of this paper is organized as follows. The following section motivates the specific research question addressed in this paper through examples that illustrate the differences between telic and atelic data semantics. In Section 3, we provide a context for this research in areas such as linguistics, artificial intelligence and temporal databases. In Section 4, we present our proposed annotation-based approach for capturing the telic/atelic dichotomy in a conceptual model; this approach focuses first on "what" (e.g., using the ER model or USM) is important for an application in the real world, then considers "when" semantics, and finally captures the temporal constraints. The basis of a modeling language is the semantics (or meaning) of its constructs; Section 5 formally defines the meaning of telic/atelic temporal entity classes, attributes, relationships and superclass-subclass. We further describe the semantics of composition, i.e., the semantics implied by the interaction of annotations, and provide a mechanism to capture temporal constraints. We conclude by highlighting the contributions of this research.

[1] Much like the ER model, USM includes constructs such as entity class, attribute and relationship. It includes different types of classes (e.g., entity and groupings) and different types of relationships (e.g., grouping). In this paper, we focus only on those concepts that overlap with the traditional ER Model.

[2] On the other hand, query semantics is of significance in the logical (relational) model, and is considered elsewhere [64].

2 Motivating Example

To motivate this research, we provide an intuitive example below. As shown in Figure 1, let us assume that a company has the following lifespan associated with their contracts, `IBM's billing project (from 2002 to 2006)`, two `Dell's accounting projects` (one from `2002` to `2003`, and the other from `2004` to `2005`) and `Cisco's database project (from 2005 to 2`[illegible]rly, let us assume that the same comp[illegible]ollowing contractors who exist over time, `Jim (from` [illegible]`06)`, `Bob` (from `2002` to `2003`, and [illegible]`04 to 2005`) and `Frank (from 2005 to 2006)`. C[illegible]lic facts, which involve a specific goal [illegible]on. Thus, if a contract (say, `IBM's` [illegible]ccomplished in a [illegible] 2006 `[2002-2006]`, it cannot be in[illegible]ed in any other time period [illegible]at is, the contract `IBM's billing project` has bee[illegible]ed in the whole period starting from 20[illegible]g in 2006, and in no other period. M[illegible]ts with Dell mee[illegible]should not be merged into one. On t[illegible]ctors is construed [illegible]e any specific goal or culmination. As a consequence, if a [illegible]isted for a period of time, s/he also ex[illegible]ub-parts of the time period. For example, while `Jim` existed as a contractor for a project from 2002-2006, one can appropriately infer that he existed (as a contractor) during 2004-2005.

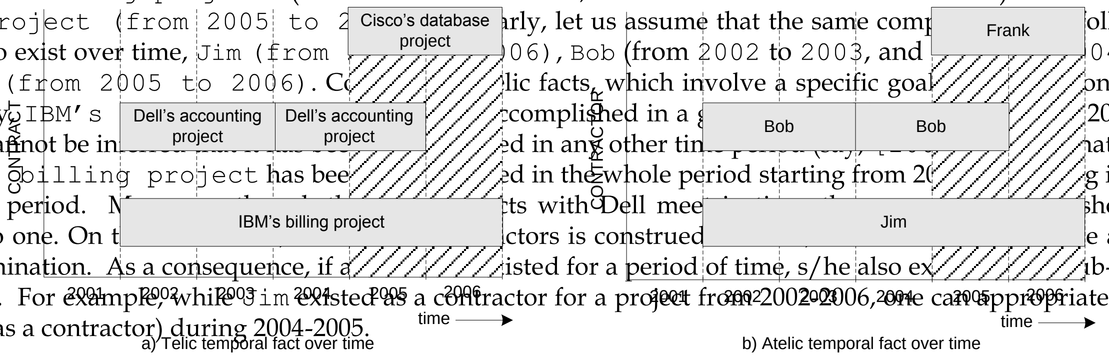

Figure 1: Motivating example using contract application

For compliance purposes, let us assume that an organization wants to audit only the contracts that were accomplished during the time period 2005-2006; similarly, let us assume that the same organization wants to audit the contractors that existed during the time period 2005-2006. (See the shaded region in Figure 1.)

Based on telic/atelic interpretation of the modeled data, Table 1 shows the results of the contracts/contractors that should be audited. (In this table, the second column refers to telic interpretation of the question ("accomplished"); the third column refers to atelic interpretation ("that existed").) As is evident from the presented results, there would be significant problems if there were no mechanism that differentiated between telic and atelic facts. For example, if all facts (in the example, both contracts and contractors) are interpreted as telic (row 1 of Table 1), then two contractors (`Jim` and `Bob`) will *not* be audited; such oversight has legal ramifications for the organization. On the other hand, if all facts are interpreted as atelic (row 2 of Table 1), then three contracts (`IBM's billing project`, `Dell's accounting project`, and `Cisco's database project`) will be erroneously audited, thus, wasting resources in the organization.

	Contracts that were accomplished in the time period 2005–2006	**Contractors that existed in the time period 2005–2006**
Ascribing Telic Semantics	`Cisco's database project` (*correct*)	`Frank` (*incorrect*)
Ascribing Atelic Semantics	`IBM's billing project`, `Dell's accounting project`, `Cisco's database project` (*incorrect*)	`Jim`, `Bob`, `Frank` (*correct*)

Table 1: Audit results based on telic and atelic interpretation of data

Having intuitively described the telic/atelic distinction, we now present prior research in linguistics that is the basis for this classification. We next show how prior research in artificial intelligence and temporal databases has operationalized this distinction that originated in linguistics.

3 Telic/Atelic Data Semantics

The distinction between *telic* and *atelic* dates back to Aristotle [6], who first noticed that facts can be partitioned into two main classes depending on whether they are goal-oriented (an example of a telic fact is "Bob built a house;" *telos* means goal in Greek) or not (an example of an atelic fact is "Bob is asleep;" in Greek "*a*" is used as a prefix to denote negation). Further, such a distinction has played a major role in linguistics and cognitive science, and lately, artificial intelligence and temporal databases, which we review below.

3.1 Linguistics

As a useable communication medium, language is generally thought of as the raw material from which data is created. The distinction between telic and atelic facts has been widely explored in linguistics (see, for example, [72]) and cognitive science (see, for example, [15]). Within the linguistics community, it is commonly agreed that sentences can be classified into different *aktionsart* classes (also called *aspectual* classes [72]) depending on their linguistic behavior or their semantic properties. The ontologic basis of this classification into stative (e.g., "asleep") and kinesis (e.g., "built a house") sentences is rooted in causation and consequence: for a kinesis sentence, there is a "culmination" after which a consequent state ensues [46]. While in the example of "built a house," there is a clearly defined "culmination," that in the example of "asleep" is not. Prior research employs the following semantic criteria to distinguish between stative (or atelic) and kinesis (or telic) statements [23]:

- A sentence φ is *stative* iff it follows from the truth of φ at an interval I that φ is true at all subintervals of I, e.g., if Bob was asleep from 1:00 to 2:00 PM, then he was asleep at all subintervals of this interval; asleep is, thus, stative. This property is referred to as *downward inheritance* by the AI community [55]; see also the next section.
- A sentence φ is *kinesis* iff it follows from the truth of φ at an interval I that φ is false at all subintervals of I, e.g., if Bob built a house from June 1 to September 1, then it is false that he built a house in any subinterval of this interval.

The linguistic community agrees that although all base facts can be classified as telic/atelic, a telic-to-atelic (or atelic-to-telic) coercion can always be performed using explicit linguistic tools; e.g., a present perfect sentence [69] can always be converted into a progressive form (cf., [23, 46, 68]). For instance, although "Bob built a house from June 1 to September 1" is a telic fact and one cannot infer that "Bob built a house on July 1," one can correctly assert that "Bob was building a house on July 1" because a progressive form has been used in the telic-to-atelic coercion.

3.2 Artificial Intelligence

Since "one of the most crucial problems in any computer system that involves representing the world is the representation of time" [4], the treatment of the telic/atelic dichotomy has had a significant impact on the artificial intelligence (AI) literature. While the ontologic basis for the telic/atelic distinction is rooted in causation and consequence, the AI literature (see, for example, [55]) distinguishes between telic and atelic facts on the basis of *upward* and *downward inheritance*; note that upward inheritance has been adapted from the linguistic literature (property of stative sentence) described in the prior section:

- The *downward inheritance* property implies that one can infer from the temporal fact f that holds at valid time t (where t is a time period) that t holds in any sub-period of t.
- The *upward inheritance* property implies that one can infer from the temporal data f that holds at two consecutive or overlapping time periods t_1 and t_2 that f holds in the union time period $t_1 \cup t_2$.

In AI, the telic/atelic dichotomy (using a different terminology) was first explicitly dealt with in Allen's reified logic [3] which models a general formal ontology to deal with time and causation and includes the famous Interval Algebra [2]. Recently, AI approaches in the development of formal ontologies pay specific attention to the telic/atelic dichotomy.

On the other hand, the treatment of the telic/atelic dichotomy has been considered by the database community only recently.

3.3 Temporal Databases

Several database management systems (DBMSs) offer support for valid and transaction time: Oracle 11g, IBM DB2 10 for z/OS, and Teradata Database 14. Part 2 (SQL Foundation) of SQL:2011 was just released, with system versioned tables (similar to transaction time) and application time period tables (similar to valid time) [1]. Recently, the telic/atelic distinction has been defined in a relational data model and algebra [64].

Continuing with the example in the prior section, we summarize the approach proposed in [64]. Figure 2 illustrates an example of a relation CONTRACT that includes attributes such as ID, task, budget and the time over which the contract was accomplished (history is represented by two columns, start VT to provide the beginning valid time, and end VT with year as the temporal granularity. From a linguistic perspective, the data associated with a contract is telic because there is a "culmination" when the contract is accomplished, e.g., 2006 for the contract, `IBM's billing project`. Based on the AI/database operationalization of telic facts, upward and downward inheritance would <u>not</u> apply for contracts. For example, it is <u>not</u> the case that contract `IBM's billing project` was accomplished in the year 2005 (downward inheritance). It is also <u>not</u> the case that two Dell's accounting projects that are contiguous and have the same task and budget would suggest that there is a single contract for `Web design` for three years from 2002 to 2005 (upward inheritance). Figure 2 also illustrates how ascribing telic or atelic semantics to the lifespan of a CONTRACT affects the "meaning" of the lifespan of a CONTRACT. Ascribing atelic data semantics (see Figure 2a) to the relation CONTRACT would incorrectly imply that contracts that were accomplished in 2005-2006 were `IBM's billing project`, `Dell's accounting project` and `Cisco's database project`, and that the two CONTRACTs with Dell are merged together into one, starting in 2002 and ending in 2005. Notice that such a merge causes a non-recoverable loss of information (indeed, this would have a significant practical impact, since only a $20000 budget would be considered for the "merged" contract, starting in 2002 and ending in 2005). On the other hand, ascribing telic data semantics (see Figure 2b) would correctly imply that only `Cisco's database project` was accomplished in that time period and would not cause any loss of information, since two distinct contracts with Dell would be recorded.

ID	Task	budget	start VT	end VT
IBM's billing project	SAP implementation	500000	2002	2006
Dell's accounting project	Web design	20000	2002	2003
Dell's accounting project	Web design	20000	2004	2005
Cisco's database project	Web design	75000	2005	2006

CONTRACT

a) **Ascribe atelic semantics to the lifespan of CONTRACT:**

```
2002 → <IBM's billing project, SAP implementation, 500000>[3]
2002 → <Dell's accounting project, Web design, 20000>
2003 → <IBM's billing project, SAP implementation, 500000>
2003 → <Dell's accounting project, Web design, 20000>
2004 → <IBM's billing project, SAP implementation, 500000>
2004 → <Dell's accounting project, Web design, 20000>
2005 → <IBM's billing project, SAP implementation, 500000>
2005 → <Dell's accounting project, Web design, 20000>
2005 → <Cisco's database project, Web design, 75000>
2006 → <IBM's billing project, SAP implementation, 500000>
2006 → <Cisco's database project, Web design, 75000>
```

b) **Ascribe telic semantics to the lifespan of CONTRACT:**

```
[2002-2006] → <IBM's billing project, SAP implementation, 500000>[4]
[2002-2003] → <Dell's accounting project, Web design, 20000>
[2004-2005] → <Dell's accounting project, Web design, 20000>
[2005-2006] → <Cisco's database project, Web design, 75000>
```

Figure 2: Example of temporal semantics for the CONTRACT relation

Recently, a two-sorted data model has been proposed in which telic data are related to time periods (conceived as atomic non-decomposable entities; see Figure 2, part (b)), so that upward and downward inheritance are not enforced on them. In such a model, atelic data are coped with in the traditional "point-based" way, so that both atelic and telic data are supported [64]. While stored data is either telic or atelic, additional `telic-to-atelic()` (or `atelic-to-telic()`) coercion functions, which are the counterpart of linguistic coercion operators, have been provided *at the query level*. (As the present paper is concerned with conceptual design and thus with data semantics associated with telic/atelic dichotomy, we do not consider such query coercion functions (query semantics) further as that would be in the purview of logical design; the reader is referred to [64] for details.) Considering the significance of differentiating telic/atelic data semantics, we next outline our proposed approach for augmenting a conventional conceptual model.

4 Annotation-based Approach

Prior research in *temporal conceptual design* divides temporal conceptual modeling into two phases: 1) first capture the "current" reality using a conventional conceptual model *without* considering the temporal aspects; and subsequently 2) annotate the schema with the temporal data semantics of the application [36, 37], if required. Based on human associative memory (see, [5]) that segregates "what" from "when," the annotation-based temporal conceptual design approach employs the generic problem-solving approach of "divide and conquer." Instead of using additional constructs to represent the temporal aspects, this approach uses annotations to represent "when" on top of the initial abstraction, which represents "what." In the second phase, we suggest that a database analyst needs to distinguish between telic and atelic data semantics because, as we saw above, an inability to do so will lead to a database that is not an accurate reflection of the "real world."

In this paper, we adapt the annotation-based approach exemplified via ST USM [36, 37]. Via annotations, we provide a means to differentiate between telic and atelic data semantics. While prior research [36, 37] employed annotations to help capture atelic semantics, in this paper we extend the annotations to include telic semantics. Our overall approach and guiding principle is schematically shown in Figure 3. USM—a conventional conceptual model that helps to capture the data semantics related to entity classes, attributes, relationships and superclass-subclass—instantiates the non-temporal data semantics (cf. Section 4.1); note that our proposed approach is not specific to USM and can be applied to any conventional conceptual model (e.g., [20, 26]). Annotations are used to instantiate both telic and atelic data semantics (cf. Section 4.2 for syntax and Section 5 for semantics). Finally, we suggest that temporal non-sequenced constraints, which we will define shortly, need to be captured in a temporal data dictionary (that is, as metadata).

[3] History, *h*, is a mapping from time structure (*T*) to value structure (*V*), i.e., *h*: *T* → *V*. The fact denoted by `<IBM's billing project, SAP implementation, 500000>` exists in the time point, `2002`.

[4] The fact denoted by `<IBM's billing project, SAP implementation, 500000>` existed in the time period `[2002-2006]`.

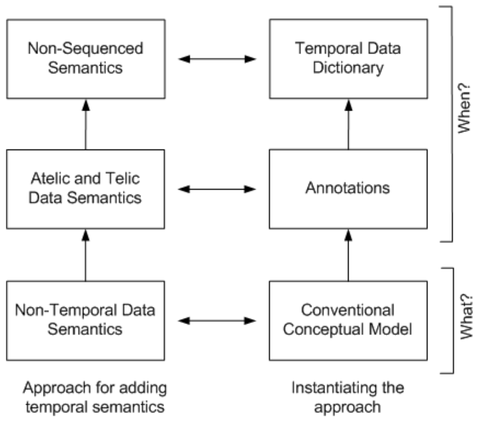
Non-Sequenced Semantics
Temporal Data Dictionary
Atelic and Telic Data Semantics
Annotations
Non-Temporal Data Semantics
Conventional Conceptual Model
When?
What?
Approach for adding temporal semantics
Instantiating the approach

Figure 3: Inducing temporal data semantics

4.1 The Underlying Conventional Conceptual Model

We summarize below how to represent "what" data semantics that can be captured using a conventional conceptual model (see, for example, [12, 20, 52, 57]), specifically USM [52], utilizing the contract application (outlined in the prior sections) as an example.

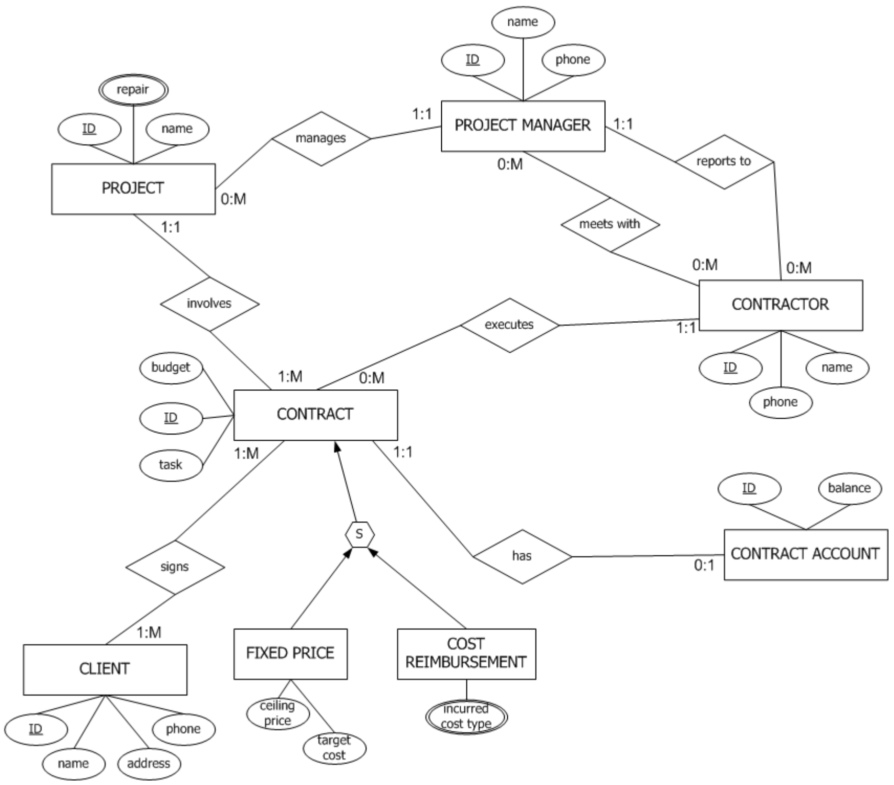

Figure 4: A conventional conceptual Schema (in USM) for a contract application

USM includes concepts such as entity class, attribute, relationship, and superclass-subclass. While the representation of real world objects is referred to by the term *entities*, their characteristics are referred to as *attributes*. While an *entity class*, sometimes also referred to as an *entity type*, is a collection of entities that have the same attributes [26], the set of instantiations of an entity class is referred to as an *entity set*. As shown in Figure 4, CONTRACTOR is an entity class with attributes such as ID, name, and phone. A relationship connects members of one entity class to members of one or more entity classes. For example, executes is a relationship between CONTRACTOR and CONTRACT. The cardinalities on the schema imply that, for example, a CONTRACTOR executes a minimum of 0 and a maximum of many (0:M) CONTRACTs and that a given CONTRACT is executed by exactly one CONTRACTOR (1:1). Also, repair is a multi-valued attribute of the entity class, PROJECT, which refers to repairs that were undertaken to fix damages that may have occurred during the execution of the project. Based on the degree and timing of the responsibility assumed by a contractor for the costs of performance, there are two type of CONTRACTs (referred to as a *superclass*), FIXED PRICE and COST REIMBURSEMENT (referred to as *subclasses*).

While the former provides for a firm price and include both a ceiling price (i.e., price that is not subject to any adjustment) and target cost, the latter provides for payment of allowable incurred costs, referred to as incurred cost type.

Figure 4 shows a conventional conceptual schema for our application, which includes entity classes, attributes, relationships, and superclass-subclass. At this point, because the temporal aspects have not yet been considered, the schema may be referred to as non-temporal. We will later extend this schema to a temporal USM schema. But before doing so, we need to describe the basis for telic/atelic data semantics and the annotations that are employed to capture these semantics.

4.2 Associating Facts with Time

In the following, we summarize extant definitions associated with how facts can interact with time [26, 52-54]. Next, we continue the motivating example to show the need for differentiating between telic and atelic data semantics. We present an enhanced annotation syntax that would help capture the telic data semantics and then apply the annotation-based approach to our motivating example.

4.2.1 Temporal Primitives

The pair (T, ≤) is used to denote a *time domain*, where T is a nonempty set of *time instants* and "≤" is the total order on T. We can assume the time domain is either *discrete* or *dense*. While there is no general agreement if the time domain is dense or discrete, the temporal database community agrees that a discrete model of time is generally adequate for representing reality [34]. For example, a discrete time domain is represented by ($\mathbf{Z}$, ≤) where instants are isomorphic to integers, implying that every instant has a unique successor. [5] Additionally, time is assumed to be bounded at both ends, i.e., the past and the future [58].

The time between two instants, e.g., from January 1, 2008 to May 31, 2010, is referred to as a *time period*. An unanchored contiguous portion of the time line, e.g., one day, is called a *time interval*. An interval is relative while an instant is absolute [59]. A non-decomposable time interval of fixed minimal duration, say, microsecond, is referred to as a *chronon*. *A temporal granularity*, a measure of the time datum [13, 14], is intrinsic to temporal data and provides a mechanism to hide details that are not known or not pertinent for an application. Some examples of temporal granularities are Gregorian day (or day) and business week.

4.2.2 The Telic/Atelic Distinction

With respect to valid time, prior research has operationalized the presence or absence of goal-related semantics via the properties of upward and downward inheritance [64].

Definition (telic facts). Telic facts as those for which upward and downward inheritance do *not* hold.

Definition (atelic facts). Atelic facts are those for which upward and downward inheritance holds.

Consistent with prior database literature, we define *atelic facts* as ones that are characterized by properties of downward and upward inheritance. On the other hand, *telic facts* are defined as facts for which neither property holds. While upward and downward inheritance holds for the lifespan of a CONTRACTOR, neither of those properties hold for CONTRACT (in Figure 4); thus, the former is atelic and the latter is telic. In summary, whereas time periods are primitive non-decomposable units for telic valid time, those for atelic valid time are time instants.

The telic/atelic dichotomy (and, in particular, the issues on whether upward and downward inheritance hold or not) is not relevant for instantaneous facts (usually called *achievements* in the linguistic literature [68] and *events* in the temporal database area [33]), i.e., facts that inherently occur at an instant in time. Events such as signing of contracts can be naturally associated to time instants.

4.2.3 The Sequenced and Non-Sequenced Distinction

As the telic/atelic distinction concerns whether downward and upward inheritance apply, it is specific to facts represented in the conceptual schema. There is another useful distinction, sequenced and non-sequenced constraint [17, 59], which concerns the entity class-attribute and entity class-relationship pairs.

Definition (sequenced). A temporal constraint is defined as *sequenced* if it is applied "independently at each instant in time."

Definition (non-sequenced). A *non-sequenced* constraint is one that does not apply independently at each instant of time.

Note that the telic-atelic distinction applies to *stored data* whereas the sequenced/non-sequenced distinction applies to *queries* and their close relative, *integrity constraints*.

The entity class-attribute (or entity class-relationship) pair can be considered to be sequenced if the interaction between the two occurs at every instant of the existence of each of the participants. For example, balance (an attribute of CONTRACT ACCOUNT) may be time-varying. As each CONTRACT ACCOUNT has a balance at *each instant of time*, this interaction is sequenced. This term reflects a view of time as a "sequence" of instants [19] with the values of the balance aligning exactly in time instants with the existence (valid) time of the CONTRACT ACCOUNT. Hence, if there was an instant when the CONTRACT ACCOUNT existed but there was no balance, or an instant when there was a balance but the CONTRACT ACCOUNT did not exist, then that sequenced constraint would be violated. Similarly, PROJECT MANAGER and PROJECT play a role

[5] An *instant* is a time point on the time line.

in the relationship manages, wherein a PROJECT MANAGER manages a PROJECT at *each instant in time*. Thus, the entity class and relationship pair, PROJECT MANAGER-manages and PROJECT-manages, is sequenced.

The entity class-attribute (and entity class-relationship) pair is considered *non-sequenced* if there is a constraint on the interaction that does not involve every instant in time, such as *before*, *meets* (these are a few of the Allen's predicates [2]) or involves some more complex constraint between the time of the participants. As an example, when a CLIENT signs a CONTRACT, the signing would happen "before" the start of the contract; there is no instant-by-instant correspondence between the two (signs and CONTRACT). Similarly, the signing of a CONTRACT by a CLIENT happens "during" the existence of a CLIENT. These non-sequenced constraints can be defined using thirteen Allen's predicates: before, before^{-1}, meets, meets^{-1}, overlaps, overlaps^{-1}, finished, finished^{-1}, during, during^{-1}, starts, starts^{-1} and equals [2].

4.2.4 Continuing the Example

We continue with the contract example to illustrate the concepts related to telic/atelic data semantics. Several aspects of the contract application need to be organized with respect to time. For example, two key entity classes of interest in the application are CONTRACT and CONTRACTOR. Both these entity classes need to be referenced with time. While the existence of a CONTRACT entails culmination (and does not satisfy upward and downward inheritance), that of a CONTRACTOR does not. The lifespan of a CONTRACT—with temporal granularity of day—needs to be represented as telic, while that of a CONTRACTOR—with the granularity of day—needs to be represented as atelic. Some of the attributes of a CONTRACT are tasks that are agreed upon in the CONTRACT and budget allocated for the CONTRACT. Two types of CONTRACTs (i.e., FIXED PRICE and COST REIMBURSEMENT) inherit the temporal properties of the CONTRACT. Before the start of a CONTRACT, a CLIENT signs it at an instant (event) in time (i.e., day).

To financially manage a CONTRACT, each CONTRACT has a CONTRACT ACCOUNT. The CONTRACT ACCOUNT has a lifespan (atelic temporal) with a temporal granularity of day. To encourage a portfolio-based approach, each CONTRACT is organized such that it is involved with exactly one (1:1) PROJECT (telic temporal with granularity of day). Each PROJECT in turn is managed by a PROJECT MANAGER. A CONTRACTOR reports to a PROJECT MANAGER and the period of time during which this reporting relationship holds needs to be captured (as atelic time). Additionally, to effectively manage a PROJECT, PROJECT MANAGERs meet with CONTRACTORs. Because meetings do not support upward and downward inheritance, the relationship meets with between a CONTRACTOR and a PROJECT MANAGER needs to be represented as telic temporal with a temporal granularity of minutes.

In summary, capturing data semantics related to, e.g., PROJECT MANAGER, PROJECT, CONTRACTOR, CONTRACT ACCOUNT, CONTRACT, CLIENT, requires a proposed temporal conceptual model to: 1) allow a data analyst to model non-temporal aspects of the application in a straightforward manner; 2) provide a mechanism to differentiate between atelic-telic temporal aspects of the application; 3) provide a common framework that applies to entity classes, attributes and relationships for expressing the structure of temporal data; 4) include a mechanism to represent multiple granularities in a conceptual schema (e.g., day, minute); and 5) enable a data analyst to specify sequenced and non-sequenced temporal constraints.

4.2.5 Annotation Syntax

Annotations provide a common framework for *optionally* specifying the temporal data semantics associated with entity classes, attributes, relationships, and superclass-subclass.

As shown in Figure 5, the overall structure of an *annotation phrase* is "⟨temporal annotation⟩". The temporal annotation first specifies the valid time (or existence time/lifespan) followed by the transaction time. The temporal annotation for valid time and transaction time is segregated by a forward slash (/). Any of these aspects can be specified as not being relevant to the associated conceptual construct with "-". The valid time can be modeled as an event (E) or a state which in turn can be an atelic state (State or Atelic State or in short S) or a telic state (Accomplishment or in short Acc); each event or state has an associated temporal granularity. For example, "S(day)/-" associated with a CONTRACTOR denotes that a CONTRACTOR exists (i.e., has an associated existence time/lifespan or valid time), and the temporal granularity of the atelic states (S) is day. Additionally, we do not need to capture transaction time (hence, "-") associated with the CONTRACTOR. Similarly, annotation phrase "E(day)/-" associated with signs implies that a contract was signed on a day. PROJECT.repair is a telic temporal attribute having "Acc(day)/-" as an associated annotation phrase because repair has culmination.

⟨annotation⟩	::=	ϵ | ⟨temporal annotation⟩
⟨temporal annotation⟩	::=	ϵ | ⟨valid time⟩ / ⟨transaction time⟩
⟨valid time⟩	::=	⟨state⟩ (⟨g⟩) | ⟨event⟩ (⟨g⟩) | -
⟨state⟩	::=	⟨telic state⟩ | ⟨atelic state⟩
⟨transaction time⟩	::=	T | Transaction | -
⟨telic state⟩	::=	Acc | Accomplishment
⟨atelic state⟩	::=	S | State | Atelic State
⟨event⟩	::=	E | Event
⟨g⟩	::=	⟨day⟩ | ⟨hour⟩ | ⟨minute⟩ | ⟨second⟩ | ⟨user defined⟩
⟨day⟩	::=	day
⟨hour⟩	::=	hr | hour

⟨minute⟩	::=	min | minute
⟨second⟩	::=	sec | second

Figure 5: Annotation syntax in BNF

In summary, annotations provide a succinct mechanism for denoting a complex assemblage of semantics: telic or atelic, event or state, valid or transaction time along with the associated granularities. Note that an annotation phrase can be uniformly applied to entity classes, attributes, relationships and superclass-subclass. Also note that this approach does not propose new "constructs" for capturing temporal data semantics. Considering that an annotation phrase can be empty (note ϵ for ⟨annotation⟩ in Figure 5), that is, any construct of a conventional conceptual model can be *optionally* annotated, the annotation-based approach *generalizes* the semantics of concepts in a conventional conceptual model, e.g., USM, using annotations.

4.2.6 The Example Revisited

Figure 6 shows the annotated temporal schema for the contract application. Note that our annotation-based approach is not specific to USM and can be applied to any conventional conceptual model [20, 26].

To develop the annotated schema (e.g., Figure 6), which is based on a non-temporal schema (e.g., Figure 4), the data analyst asks the domain expert three particular questions. First, do you want to capture history (lifespan) or only current values of the facts (objects)? Second, does the fact need to be modeled as an event or state (i.e., durative facts)? And finally, does upward and downward inheritance hold for the durative facts (i.e., telic or atelic facts)? Accordingly, the data analyst annotates the schema. Note how Figure 6 augments the schema shown in Figure 4 with temporal annotations. Some entity classes, attributes and relationships remain as non-temporal. This does not imply that they are non-temporal in reality, but rather that one is not interested to deal with their temporal features (if any) in the application or the "miniworld."

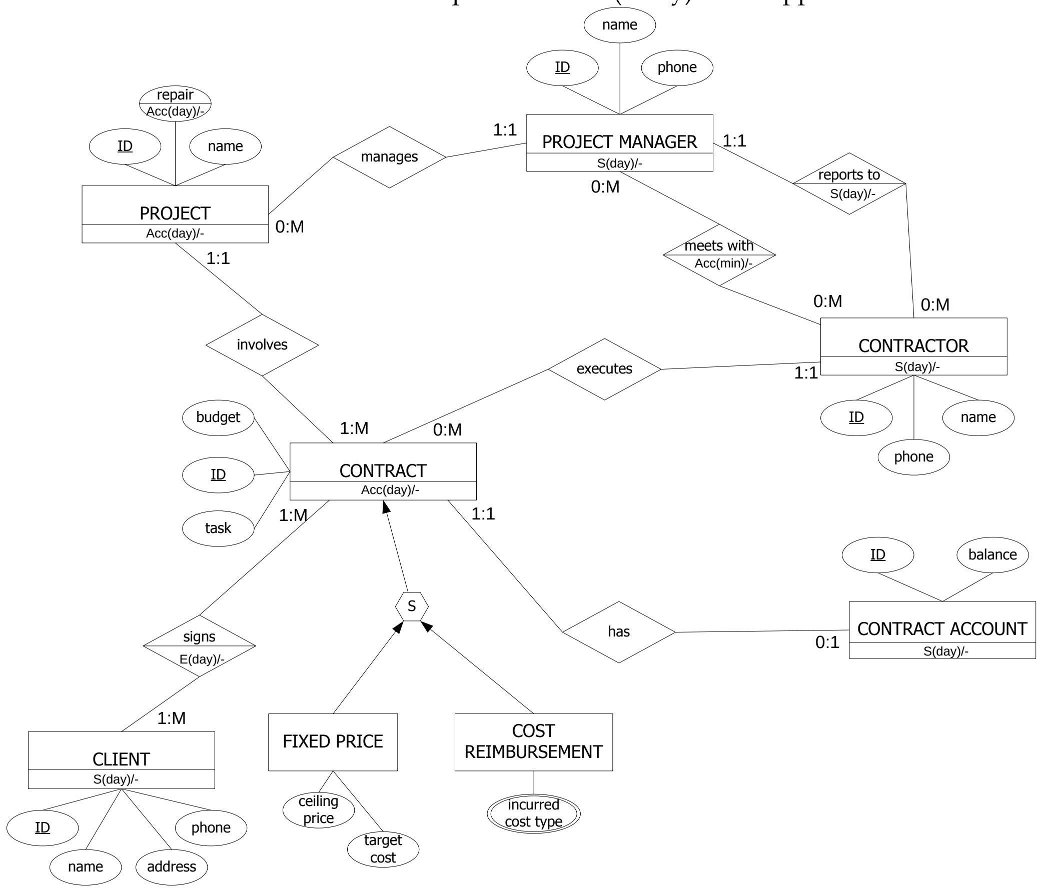

Figure 6: An annotated temporal schema for the contract application

4.2.7 Specifying Non-sequenced Semantics

As shown in Figure 3, our approach for adding temporal semantics proceeds in three steps. Figure 4 provides an example of "what" semantics, while Figure 6 is an example of "when" (along with "what") semantics using annotations. We next turn to the third step, which involves capturing the non-sequenced semantics.

Attribute/Relationship	Entity class(es)	Non-sequenced constraint	Comment
signs	CONTRACT	before	Signing of the contract must happen before the

			lifespan of the contract.
meets with	CONTRACTOR	during	The time period of a meeting must be during the lifespan of a contractor.
meets with	PROJECT MANAGER	during	The time period of a meeting must be during the lifespan of a project manager.
reports to	CONTRACTOR	during	The time period of the reporting relationship must be during the lifespan of a contractor.
reports to	PROJECT MANAGER	during	The time period of the reporting relationship must be during the lifespan of a project manager.
repair	PROJECT	during	The time period of a repair must occur during the lifespan of a project.

Table 2: Specifying non-sequenced constraints

Besides the semantics that are implied by annotations, a data analyst can optionally define other explicit temporal constraints that capture temporal relationships; such constraints are based on Allen's predicates [2] such as before, meets, overlap, finished, during, starts, and equal. See Table 2 for an example of non-sequenced constraints in a temporal data dictionary. A temporal constraint can be specified between any two annotated constructs, e.g., between the temporal relationship signs and the temporal entity class CONTRACT or between the temporal entity class PROJECT and the temporal attribute repair. As shown in Table 2, the constraint before between CONTRACT ("Acc(day)/-") and signs ("E(day)/-") implies that the event signs must happen "before" the lifespan of the CONTRACT. Note that the table does not directly differentiate between before and before^{-1} (or *after*); this is indicated in the comment column instead.

Having discussed how annotation phrases and non-sequenced temporal constraints can be specified, we next examine the semantics of annotations as they apply to entity classes, attributes, relationships and superclass-subclass.

5 Explicating Temporal Semantics

We first formally define telic/atelic data semantics. We then employ examples to illustrate how to specify the semantics of telic/atelic annotations on entity classes, attributes, relationships and superclasses-subclasses using the conventional USM; we do so using an approach that employs inheritance.

5.1 Semantics of Annotations

As shown in Figure 7 below, VT_EVENTUALITY and TT_EVENTUALITY are superclasses in our superclass/subclass hierarchy that are associated with valid time and transaction time, respectively. Each VT_EVENTUALITY and TT_EVENTUALITY is associated with a TEMPORAL_GRANULARITY. TEMPORAL_GRANULARITY (described in detail in [37]) includes the recursive relationships groups_into and anchor_gran, which helps create the *granularity graph* [24].

The valid time eventuality[6] (VT_EVENTUALITY) has three subclasses that represent events (VT_EVENT), atelic state (VT_STATE) and telic state (VT_ACCOMPLISHMENT). On the other hand, transaction time eventuality (TT_EVENTUALITY) has one subclass (TT_STATE) that indicates that transaction time is always atelic (i.e., there is *no* TT_ACCOMPLISHMENT subclass). We adopt maximal convex time periods to represent the valid time of atelic facts [23, 64], thus implying that the fact holds at each instant (chronon) within the maximal time period. Each TT_STATE holds in a set of maximal convex time periods, where each time period is represented with indexes begin and end. While VT_EVENT occurs in time points, VT_STATE holds over maximal_periods (representing a convex maximal set of time instants). The association of telic facts to time must be dealt with in a different way with respect to atelic facts. Convex maximal periods, interpreted as convex sets of time instants, cannot be used since upward and downward inheritance would be inappropriately enforced. On the other hand, telic semantics in which time periods are atomic—that is, ones that cannot be divided or merged together and time periods *can* overlap—and need to be associated with these facts. Hence, VT_ACCOMPLISHMENT holds over telic_periods (representing atomic indivisible entities). We define the inherent semantics of temporal concepts by stating formal axioms. These axioms are predicates that are implied by the annotations the data analyst applies to a conceptual schema. For the sake of generality and compactness we widely exploit inheritance: axioms concern the "high-level" entity classes in Figure 7 (e.g., VT_ACCOMPLISHMENT), and are inherited by all their specializations (e.g., CONTRACT). In these predicates, the values of attributes of instances of classes are denoted functionally. So if e is a specific instance of VT_ACCOMPLISHMENT, then VT_ACCOMPLISHMENT(e, telic_period) denotes the value of the telic_period attribute of e. A set of entities of an entity class, E, is represented as $S(E)$.

[6] Following the linguistic literature (see, for example, [68]), we use the term *eventuality* for all aktionsart statements.

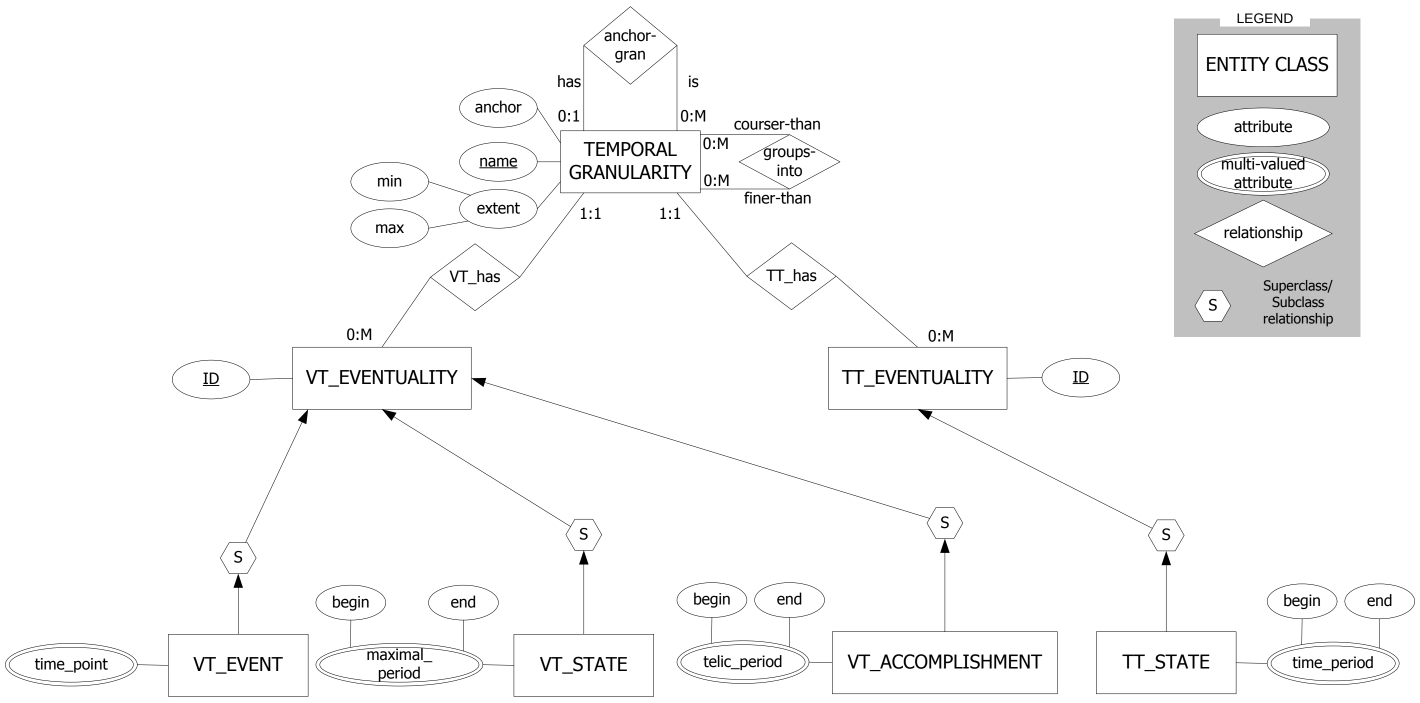

Figure 7: Semantics of temporal annotations

Axiom 1: The following axioms characterize the maximal_period and telic_period attributes. Note how a well-formed telic period (in the first statement) differs from that of an atelic period (in the second and third statements).

a) Telic periods (i.e., the valid times of VT_ACCOMPLISHMENTs) are well-formed.
$\forall e \in S(\text{VT\_ACCOMPLISHMENT}), \forall p \in \text{VT\_ACCOMPLISHMENT}(e, \text{telic\_period}), \text{begin}(e, p) < \text{end}(e, p)$

b) Maximal temporal periods (i.e., the valid time of VT_STATEs) are well-formed.
$\forall e \in S(\text{VT\_STATE}), \forall p \in \text{VT\_STATE}(e, \text{maximal\_ period}), \text{begin}(e, p) < \text{end}\ (e, p)$

c) Maximal temporal periods (i.e., the valid time of VT_STATEs) cannot overlap in time. Such a constraint does *not* hold for telic periods.
$\forall e \in S(\text{VT\_STATE}), \forall p_1, p_2 \in \text{VT\_STATE}(e, \text{maximal\_ period}), \text{begin}(e, p_1) < \text{begin}(e, p_2) \Rightarrow \text{end}\ (e, p_1) < \text{begin}\ (e, p_2)$

Axiom 2: Upward and downward inheritance holds for atelic state (VT_STATE) but not for telic accomplishment (VT_ACCOMPLISHMENT). In order to state the properties of VT_STATE and VT_ACCOMPLISHMENT, we need to explicitly model the association between each valid time eventuality (VT_EVENTUALITY) and its time of occurrence. We thus introduce "hold" as an inferred attribute that relates each VT_EVENTUALITY to its valid time. For instance, considering the example in Figure 1, we have VT_STATE (`IBM's billing project`, hold) = `[2002-2006]`.

$$\begin{aligned} &\forall e \in S(\text{VT\_EVENTUALITY}), \\ &\quad \text{VT\_EVENT}(e, \text{time\_point}) \Rightarrow \text{VT\_EVENTUALITY}(e, \text{hold}) = \text{to\_element}\ (\text{VT\_EVENT}(e, \text{time\_point})\ ) \\ &\quad \wedge\ \text{VT\_STATE}\ (e, \text{maximal\_period}) \Rightarrow \text{VT\_EVENTUALITY}(e, \text{hold}) = \text{VT\_STATE}(e, \text{maximal\_period}) \\ &\quad \wedge\ \text{VT\_ACCOMPLISHMENT}\ (e, \text{telic\_period}) \Rightarrow \text{VT\_EVENTUALITY}(e, \text{hold}) = \text{VT\_ACCOMPLISHMENT}(e, \text{telic\_period}) \end{aligned}$$

where to_element is a function that takes as input a time point (instant) and coerces it to a one-chronon period.

The *VT_Down* and *VT_Up* axioms in the following specify that the properties of downward and upward inheritance that apply to VT_STATE.

a) VT_Down

$$\forall e,\ p_1, p_2, \text{VT\_STATE}(e) \wedge \text{VT\_STATE.hold}(e, p_1) \wedge p_2 \subseteq p_1 \Rightarrow \text{VT\_STATE.hold}(e, p_2)$$

b) VT_Up

$$\begin{aligned} &\forall e,\ p_1, p_2, \text{VT\_STATE}(e) \wedge \text{VT\_STATE.hold}(e, p_1) \wedge \text{VT\_STATE.hold}(e, p_2) \wedge \\ &\quad (\text{MEETS}(p_1, p_2) \vee \text{MEETS}^{-1}(p_1, p_2) \vee \text{OVERLAPS}(p_1, p_2) \vee \text{OVERLAPS}^{-1}(p_1, p_2) \end{aligned}$$

$$\lor \text{FINISHED}(p_1, p_2) \lor \text{FINISHED}^{-1}(p_1, p_2) \lor \text{DURING}(p_1, p_2)$$
$$\lor \text{DURING}^{-1}(p_1, p_2) \lor \text{STARTS}(p_1, p_2) \lor \text{STARTS}^{-1}(p_1, p_2) \lor \text{EQUALS}(p_1, p_2)$$
$$) \Rightarrow \text{VT\_STATE.hold}(e, p_1 \cup p_2)$$

In the above, $\cup$ denotes set-union over maximal periods, i.e., over sets of time points and the temporal relations in the disjunction are relations of Allen's Interval Algebra [2] that indicate that p_1 and p_2 are either contiguous or intersecting in time (i.e., they are not temporally disjoint); see [2] for a schematic of Allen's predicates. The absence of analogous axioms for VT_ACCOMPLISHMENT means that downward and upward inheritance *do not* hold for accomplishments. Indeed, that is the central telic/atelic distinction we wish to capture.

5.2 Entity Class, Attribute, Relationship and Superclass-Subclass

In the following, using examples we formally define the semantics of an annotated temporal entity class, relationship, attribute and superclass-subclass, collectively referred to as *ERAS*. Specifying the semantics of a telic/atelic fact is rather straightforward: the semantics of each annotated ERAS can be obtained by classifying it as a subclass of an appropriate temporal entity class such as VT_EVENT, VT_STATE, VT_ACCOMPLISHMENT and TT_STATE; see Figure 7. The temporal semantics of the annotated ERAS are thus automatically obtained through inheritance of the attributes and thus the axioms on those attributes from such superclasses. The schema that explicitly shows the semantics of the annotations is referred to as the *translated USM schema*; see the bottom of Figure 8 for an example. We present an example of the translated USM schema (along with the automatically generated axioms) for a (shaded) fragment of the annotated temporal schema in Section 6.

5.2.1 Temporal Entity Class

We describe the semantics of a telic temporal entity class using the example of CONTRACT. As discussed above, CONTRACT is telic since neither downward nor upward inheritance applies. Figure 8 shows the semantics of the telic temporal entity class, CONTRACT. Note that the associated annotation for this entity class is "Acc(day)/-".

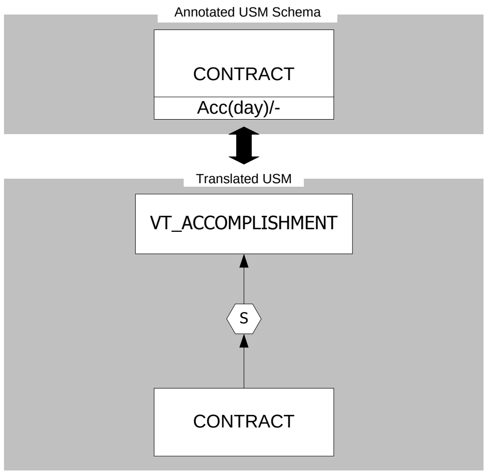

Figure 8: Semantics of telic temporal entity class

The semantics of CONTRACT is explicated by relating it to the appropriate superclass, which is VT_ACCOMPLISHMENT. This implies that CONTRACT inherits all the properties (i.e., attributes and relationships) of its superclasses, VT_ACCOMPLISHMENT and VT_EVENTUALITY (i.e., only axiom 1a will apply). Similarly, the semantics of an atelic entity class (e.g., CONTRACTOR) can be obtained by classifying it as a subclass of VT_STATE; in this way, the upward and downward properties are obtained through inheritance of the "maximal-period" attribute; thus axioms 1b, 1c, and 2 apply. For instance, a contract `IBM's billing project` with a lifespan `[2002-2006]` from VT_STATE (`IBM's billing project`) and axiom 2 (first part), one gets VT_STATE (`IBM's billing project`, hold) = `[2002-2006]`, which in conjunction with Axiom 2a (VT_Down) and that `[2004,2004]` $\subseteq$ `[2002,2006]` implies that the contract `IBM billing project` was valid in 2004, i.e., VT_STATE (`IBM's billing project`, hold) = `[2004-2004]`.

Bi-temporal entity classes would be entity classes that are a specialization of both VT_EVENTUALITY (VT_EVENT, VT_STATE or VT_ACCOMPLISHMENT) and TT_EVENTUALITY (TT_STATE).[7]

Finally, the following axiom related to granularity is similar to that in [36].

Axiom 3: All the entities in CONTRACT have the same associated temporal granularity (day).

$$\forall e \in S\,(\text{CONTRACT}),\ \text{CONTRACT.VT\_ACCOMPLISHMENT.VT\_EVENTUALITY. VT\_has.TEMPORAL\_GRANULARITY}(e, \text{name}) = \text{day}$$

[7] Compared with the previous approach [35, 36], employing superclasses and subclasses to represent temporal data semantics is more parsimonious.

5.2.2 Temporal Attribute

The semantics of temporal attributes can be explicated through the introduction of an auxiliary "constructed" entity class (in the translated USM schema), which is then associated with the superclass VT_STATE, VT_EVENT or VT_ACCOMPLISHMENT. The auxiliary constructed entity class is related to the entity class to which the attribute refers via an "auxiliary constructed relationship". The cardinalities to the original entity class is 1:1 for standard attributes and 0:M for multi-valued attributes. Additionally, axiom 1a and that similar to axiom 3 would apply.

Figure 9 shows the semantics of a telic temporal attribute PROJECT.repair in Figure 6. As shown in Figure 9, the auxiliary "constructed" entity class SEM_ATTR_repair_PROJECT is a subclass of VT_ACCOMPLISHMENT and is related to the entity class PROJECT by the "auxiliary constructed relationship" ATTR_OF_repair_PROJECT. Each PROJECT is related to a minimum of 0 and a maximum of many (M) SEM_ATTR_repair_PROJECTs during the lifetime of that PROJECT.

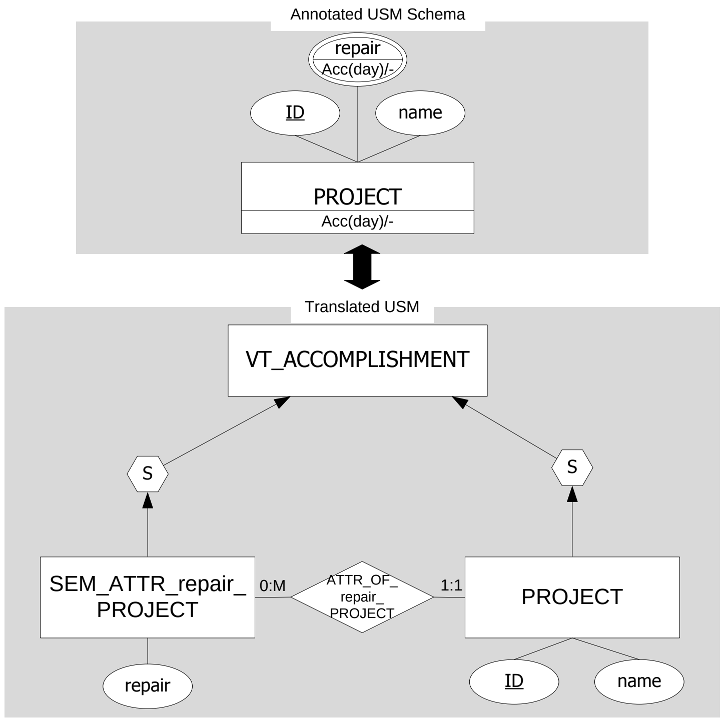

Figure 9: Semantics of telic temporal attribute

5.2.3 Temporal Relationship

The explication of the semantics of a temporal relationship is similar to that of an attribute described above. For example, Figure 10 shows the semantics of a telic temporal relationship (meets_with). The semantics is explicated through the introduction of auxiliary "constructed" entity class, SEM_REL_meets_with. While the "constructed" entity class (SEM_REL_meets_with) inherits attributes and relationships from VT_ACCOMPLISHMENT superclass, PROJECT MANAGER and CONTRACTOR inherit attributes and relationships from VT_STATE because their associated annotations are "Acc(min)/-" and "S(day)/-", respectively. Additionally, axiom 1a and one similar to axiom 3 would apply.

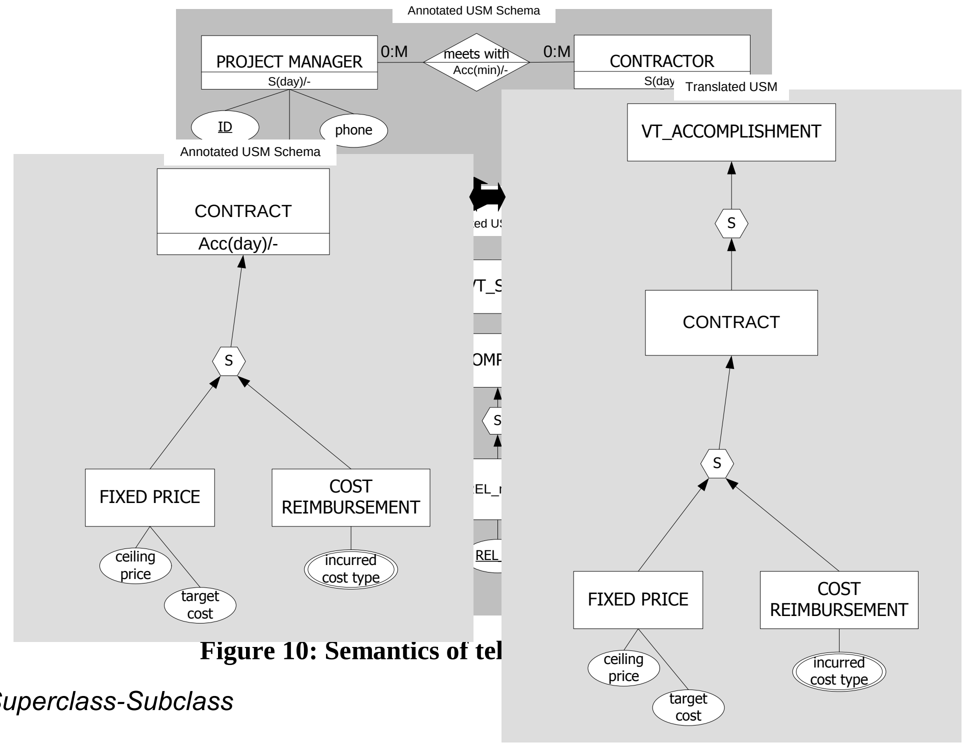

Figure 10: Semantics of tel

5.2.4 Temporal Superclass-Subclass

Figure 11: Semantics of telic superclass-subclass

Subclass and superclass represent a type of abstraction that allows modeling the same real-world object at different levels of abstraction. As shown in Figure 11, an application may require capturing the lifespan of the generic abstraction (i.e., the superclass CONTRACT) and not for that of the specific abstraction (i.e., the subclasses, FIXED PRICE and COST REIMBURSEMENT). As shown in the figure above, the subclasses inherit the properties of all the superclasses (CONTRACT as well as VT_ACCOMPLISHMENT). We describe other possible types of superclass-subclass interactions in Section 5.3.1.

5.3 Semantics of Composition

In Section 5.2, we discussed how the telic/atelic distinction can be captured via annotations; however, we presented the semantics of annotations as related to ERASs "in isolation." In this section, we take into account constraints on the interactions (sequenced/non-sequenced) between a pair of ERASs, i.e., entity class-relationship, entity class-attribute and superclass-subclass. Thus, sequenced/non-sequenced distinction relevant to *constraints* on the interactions between time-based entity classes is separate from the telic/atelic distinction associated with the time-based entity classes (that is, the semantics of the instances/*data* of these classes).

As shown in Figure 6, signs (represented as an event) is a relationship between entity classes CLIENT (which is atelic) and CONTRACT (which is telic). While the relationship between signs and CLIENT is sequenced which implies that signing must be performed at one instant during the existence of the client, that between signs and CONTRACT is non-sequenced (see Table 2) since signing must be performed "before" the lifespan of a contract. This example illustrates an important feature of our approach: the interplay of semantics between ERASs can be captured in a pair-wise compositional way.

Consistent with existing approaches, we assume that constraints are sequenced by default, while non-sequenced constraints can be explicitly specified by the data analyst in the temporal data dictionary (see Table 2 for example). In the rest of this section, we first provide the default (sequenced) semantics with the understanding that a data analyst can override these semantics by providing alternative constraints explicitly. We then consider non-sequenced semantics.

5.3.1 Sequenced Semantics

In the sequenced interpretation, an exact correspondence (for each chronon of time) is required between the pair of entity class-attribute or entity class-relationship or superclass-subclass.

As shown in Table 3, non-temporal, atelic and telic temporal attributes/relationships can interact with non-temporal, atelic, and telic temporal entity classes. Table 3 indicates that 16 cases must be taken into account; however, just a few general principles are adequate to summarize all the possible combinations. Note that the sequenced interpretation requires a chronon-by-chronon correspondence, which, in turn, requires an atelic interpretation of the involved ERAs. Therefore, telic ERAs must be coerced into atelic ones (column "coercion to atelic" in Table 3). Here, it is important to stress that, since the sequenced (and non-sequenced) feature concerns the temporal relationship between an entity class and an attribute or an entity class and a relationship, the coercions in Table 3 have a limited scope: they are used solely in the context of evaluating the temporal constraints between a pair of entities, but they have no effect the basic telic/atelic nature of such entities. In a general sense, the sequenced an non-sequenced interpretation impose "local" constraints on pairs of entities in the same way in which temporal conjunctions impose constraints on pair of sentences. For instance, the "while" conjunction establishes a sort of "sequenced" constraint between the sentences it relates, since it states that there is an "instant-by-instant" correspondence between the occurrences of the facts they describe. In case the "while" conjunction is applied to one (or two) telic sentence, the "instant-by-instant" constraint is enforced by "looking at the telic fact from the inside", i.e., by coercing it to an atelic interpretation. However, such a coercion has a "local" scope: the "while" sentence; On the other hand, it does not permantly change the telic fact into atelic (e.g., it does not permanently strip the culmination to the telic fact) [62]. Consider, for instance, the following sentences. "John wrote his first book in six months. While he was writing, he was nervous and stressed." In the "while" sentence, the telic fact that John wrote a book is looked from the inside, instant-by-instant, to say that in all such instants he was nervous and stressed. Thus, a coercion to atelic is performed in this context (indeed, the coercion is explicitly caused by the application of a progressive form). However, this coercion is "local" to the "while" sentence itself and by no means strips the culmination to the telic activity of writing the book (i.e., after the two sentences, we may still infer the fact that John finished to write the book). Similarly, the coercions in Table 3 are "local" to the interpretation of the sequenced constraint, and do not change permanently the telic/atelic nature of the involved data. The following general principles underlie the rules in Table 3.

- The sequenced interpretation forces a time point-by-time point correspondence between entity class and attribute/relationship. Thus, if they are telic (VT_ACCOMPLISHMENT), entity class and/or attribute/relationship must be coerced to atelic (just for the semantics of the interpretation of the sequenced interaction). Coercion is not needed for VT_EVENTS (as they are punctual).
- If the attribute/relationship (or entity class) is non-temporal and the entity class (or attribute/relationship) is temporal, the non-temporal attribute/relationship (or entity class) is coerced to be temporal, e.g., of type VT_STATE (for row 7 in Table 3). The rationale is that if the data analyst wanted something else, they should have specified so. As an example, the name of a PROJECT in Figure 7, which yields a constraint that a name (a non-temporal attribute) must be considered to be associated with each project during the project's existence.
- If both the entity class and the attribute/relationship are temporal and durative, the sequenced interpretation dictates that their validity times must be identical.
- If the attribute/relationship is non-durative (an event) and the entity class is durative (VT_STATE or VT_ACCOMPLISHMENT), or vice versa, the sequenced interpretation requires that the valid time of the event (a time instant) must be *during* the validity time of the durative ERA.
- Participation constraints also apply at each point in time. So an attribute must have a single value at each point in time (equivalently, the time periods of different values cannot overlap) and similarly for the "at most one" side of a relationship.

Table 3 enumerates the cases of semantics of composition, where "EC" and "A/R" refers to entity class and attribute/relationship, respectively. The first column in this table is simply the row number. Sixteen constraints are required to fully cover all the possibilities. The second and third columns identify various combinations between different temporal types of "A/R" and "EC," i.e., S, Acc, E and ϵ (as specified in syntax in Figure 5) or VT_STATE, VT_ACCOMPLISHMENT, VT_EVENT, non-temporal, respectively (as specified in semantics in Figure 7). So, for example, row 2 considers either a non-temporal attribute of a state entity class or a state entity class participating in a non-temporal relationship. The fourth column lists a conversion which is necessarily implied by seven of the combinations. The final column gives some details of the semantics and whether such a conversion is required (an empty cell just means there is nothing more to say). Note that there is semantics for every combination of constructs specified by the designer.

This table can be expressed by a set of constraints. We provide here the constraint corresponding to the row 2 of the table where: 1) the attribute/relationship (which has been specified as non-temporal) is inferred as temporal; and 2) the valid time of the attribute/relationship and that of the entity class must be equal. For each non-temporal attribute (e.g., name of the entity class PROJECT MANAGER), we first need an auxiliary entity (e.g., SEM_ATTR_PROJECT MANAGER_name) that is used to infer the ATTR_OF predicate.

$\forall e_1 \in S(\text{PROJECT MANAGER}), \forall e_2 \in S(\text{name}), \text{SEM\_ATTR\_PROJECT MANAGER\_name}\ (e_1, e_2) \Rightarrow \text{ATTR\_OF}\ (e_2, e_1)$

Then, the general constraint below is used to enforce the fact that the ATTR_OF predicate makes the (coerced) valid time of the (auxiliary entity representing the) attribute (or relationship) equal to that of the temporal entity it is associated with.

$$\begin{aligned}&\forall e, a, p_1,\ e.\text{VT\_STATE} \wedge (e.\text{VT\_STATE(hold)} = p_1) \wedge \text{ATTR\_OF}(a, e)\\&\qquad\wedge (a.\text{annotations}{\neq}\text{Acc} \wedge a.\text{annotations}{\neq}\text{E} \wedge a.\text{annotations}{\neq}\text{S})\\&\quad\Rightarrow a{\in}\ \text{S(VT\_STATE)} \wedge a.\text{VT\_STATE(hold)} = p_1\end{aligned}$$

No	Type of attribute/relationship (*A/R*)	Type of entity class (*EC*)	Coercion to atelic	Semantics
1	ϵ or non-temporal	ϵ or non-temporal	–	Conventional
2	ϵ or non-temporal	S or VT_STATE	–	Infer *A/R* is temporal
3	ϵ or non-temporal	Acc or VT_ACCOMPLISHMENT	*EC*	Infer *A/R* is temporal
4	ϵ or non-temporal	E or VT_EVENT	–	Infer *A/R* is temporal

No	Type of attribute/relationship (*A*/*R*)	Type of entity class (*EC*)	Coercion to atelic	Semantics
5	S or VT_STATE	S or VT_STATE	–	
6	S or VT_STATE	Acc or VT_ACCOMPLISHMENT	*EC*	
7	S or VT_STATE	ϵ or non-temporal	–	Infer *EC* is temporal
8	S or VT_STATE	E or VT_EVENT	–	
9	Acc or VT_ACCOMPLISHMENT	S or VT_STATE	*A/R*	
10	Acc or VT_ACCOMPLISHMENT	Acc or VT_ACCOMPLISHMENT	*EC*, A/R	
11	Acc or VT_ACCOMPLISHMENT	ϵ or non-temporal	*A/R*	Infer *EC* is temporal
12	Acc or VT_ACCOMPLISHMENT	E or VT_EVENT	*A/R*	
13	E or VT_EVENT	S or VT_STATE	–	
14	E or VT_EVENT	Acc or VT_ACCOMPLISHMENT	*EC*	
15	E or VT_EVENT	ϵ or non-temporal	–	Infer *EC* is temporal
16	E or VT_EVENT	E or VT_EVENT	–	

Table 3: Interaction of temporal/non-temporal attribute/relationship with temporal/non-temporal entity class

Similar constraints are required to express each possible case in Table 3. This table considers all combinations of an attribute associated with an entity class (and also, a relationship in which an entity class participates). Consider interaction number 2, in which an attribute is non-temporal whereas the entity class is an atelic state. An example is the phone of a PROJECT MANAGER in Figure 6. In this case, phone is considered to be time-varying as well, with a period of validity equal to that of the PROJECT MANAGER. No conversion to atelic is needed because the attribute is already considered to be atelic.)

It is worth noticing that, based on row 11 of Table 3, if an attribute/relationship of an entity class is telic (VT_ACCOMPLISHMENT), it is converted to atelic for the purpose of the chronon-by-chronon equality constraint. For example, the PROJECT entity class is telic as is its attribute repair; see Figure 6. Each PROJECT, say `P1` with lifespan `[2004–2007]`, has associated repairs, say `R1` (`[2005–2006]`) and `R2` (`[2006–2007]`), which implies that the periods of the possibly several repairs, each at different times, are converted to atelic, collated together (`[2005–2007]`), and then compared for equality with the existence time of the PROJECT `P1`.

Note that there are subtle ramifications in considering non-temporal entity classes to be temporal for the sake of the sequenced semantics. For example, if the phone attribute of CONTRACTOR is represented as atelic state (with temporal granularity of day) and the CONTRACTOR is assumed non-temporal, CONTRACTOR will be considered to be an entity class whose lifespan (valid time) is equal to the validity period of phone; see row 7 of Table 3. Should CONTRACTOR have another atelic temporal attribute, that attribute would be required by this “composed” semantics to have a validity period equal to that of CONTRACTOR, thus, that of the phone attribute. By this linkage, which we term *compositionality*, the implication is that under the sequenced interpretation all temporal attributes must have identical validity time periods. While prior research [44] suggests that “all” objects, their properties and relationships are embedded in time, data analysts can *choose* (or *not choose*) to capture the associated temporality for an application. Thus, implicit semantics (described above) elucidate the temporality “assumptions” for an application.

Non-temporal, atelic and telic temporal subclasses can be related with non-temporal, atelic and telic temporal superclasses; see Table 4. Notice that the subclass-superclass interaction requires that the instances of the subclass are also instances of the superclass. As a consequence, subclasses must be *of the same temporal type* as that of the superclass. Thus, for instance, we do not allow that a VT_STATE is a subclass of a VT_ACCOMPLISHMENT. (The reason is that atelic and telic are orthogonal to each other, and so it doesn’t make sense to state that one is a refinement of the other.) When the temporal type of the superclass matches that of the subclass, the times of subclass entities must be *equal* to the time of their superclass. Finally, in all the cases in which one of the two entities is non-temporal, and the other is temporal, we admit a temporal interpretation for the non-temporal entity.

No	Type of subclass	Type of superclass	Semantics
1	ϵ or non-temporal	ϵ or non-temporal	Conventional
2	ϵ or non-temporal	S or VT_STATE	Infer subclass is temporal
3	ϵ or non-temporal	Acc or VT_ACCOMPLISHMENT	Infer subclass is temporal
4	ϵ or non-temporal	E or VT_EVENT	Infer subclass is temporal
5	S or VT_STATE	S or VT_STATE	Allowed
6	S or VT_STATE	Acc or VT_ACCOMPLISHMENT	Disallowed
7	S or VT_STATE	ϵ or non-temporal	Infer superclass is temporal
8	S or VT_STATE	E or VT_EVENT	Disallowed
9	Acc or VT_ACCOMPLISHMENT	S or VT_STATE	Disallowed
10	Acc or VT_ACCOMPLISHMENT	Acc or VT_ACCOMPLISHMENT	Allowed
11	Acc or VT_ACCOMPLISHMENT	ϵ or non-temporal	Infer superclass is temporal
12	Acc or VT_ACCOMPLISHMENT	E or VT_EVENT	Disallowed
13	E or VT_EVENT	S or VT_STATE	Disallowed
14	E or VT_EVENT	Acc or VT_ACCOMPLISHMENT	Disallowed
15	E or VT_EVENT	ϵ or non-temporal	Infer superclass is temporal

No	Type of subclass	Type of superclass	Semantics
16	E or VT_EVENT	E or VT_EVENT	Allowed

Table 4: Interaction of temporal/non-temporal subclass with temporal/non-temporal superclass

5.3.2 Non-Sequenced Semantics

As discussed in Section 4.2.7, data analysts may specify non-sequenced constraints between ERAs. Besides the constraints that are *implied* by annotations, a data analyst can optionally define other explicit temporal constraints that capture the temporal relationship; such constraints are based on Allen's predicates [2] such as before and meets. The semantics of such non-sequenced constraints is just that of these predicates applied to the relevant timestamps. Moreover, the special constraints "no_constraint" represents the absence of temporal constraints, and may be used to relate non-temporal entities.

5.4 Temporal Propagation

In the prior section we showed that in both sequenced and non-sequenced interpretations, temporal constraints are imposed on the validity time of the related ERASs (with the only exception of the "no_constraint" option). Such pairwise temporal constraints propagate in a compositional way ma. Specifically, a chain of sequenced and non-sequenced interactions (except the "no_constraint" temporal ERAS recursively propagate the inference that all the ERASs in the chain are tempora se no_constraint interaction is specified. Additionally, ea raction (except "no_constra aint between the related ERASs. Specifically, the co int is licitly specif d in the c , meets). On the other hand, if a sequenced interaction involves two du VT_STA two instantaneous ERASs (VT_EVENT) the t nt is that their valid ti equenced interaction between an instant rative ERAS imposes th t be during the valid time of the durative one.

For example, consider the entity classes PROJECT and PROJE ges between them. Based on the annotated schema shown in Figure 6, PROJECT, PROJ epresented as telic, atelic and nontemporal, resp Moreover (by default), the re ages and between manages and PROJECT MANAGER are both *sequenced*. Conside T a Table 3 applies, so that PROJECT is coerced to atelic, manages is coerced to atelic and their valid he same. Consider again PROJECT MANAGER a anages, where rule 2 in Table 3 applies, so that lic, and its validity time is impos validity time of the two m lly *implied* by the underlyi een ERASs.

Constrai notated schema. In partic k the overall consistency c used in order to check that ma (see, e.g., [63]).

6. CONTRAC

To illustrate ontinue with the example ed portion of the annotate wing axioms would be au to a translated USM sche

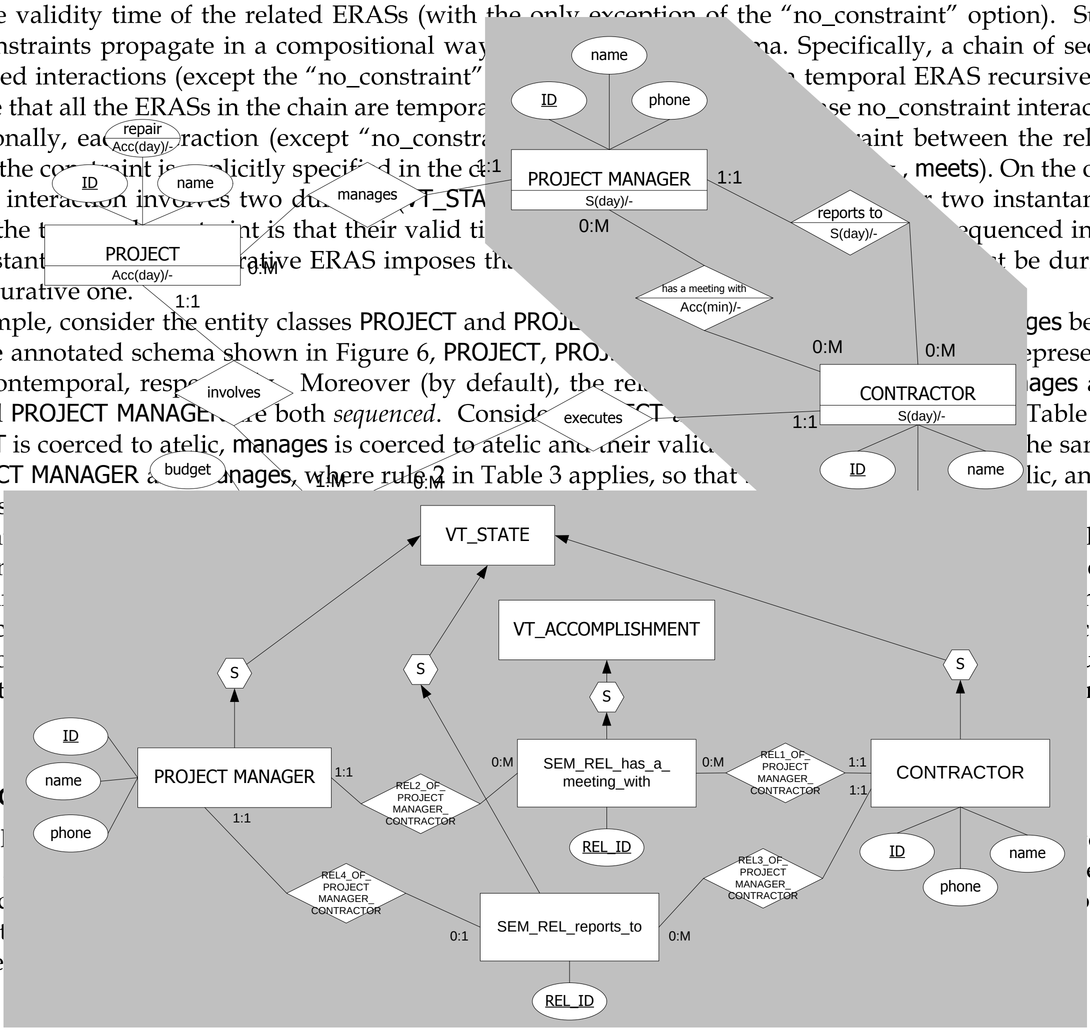

Figure 12: Semantics of the (shaded) fragment of the annotated schema for the CONTRACT application

Axiom 1

a) $\forall e \in S(\text{VT\_ACCOMPLISHMENT}), \forall p \in \text{VT\_ACCOMPLISHMENT}(e, \text{telic\_period}),$
$\text{begin}(e, p) < \text{end}(e, p)$

b) $\forall e \in S(\text{VT\_STATE}), \forall p \in \text{VT\_STATE}(e, \text{maximal\_ period}), \text{begin}(e, p) < \text{end}(e, p)$

c) $\forall e \in S(\text{VT\_STATE}),$
$\forall p_1, p_2 \in \text{VT\_STATE}(e, \text{maximal\_ period}), \text{begin}(e, p_1) < \text{begin}(e, p_2) \Rightarrow$
$\text{end}(e, p_1) < \text{begin}(e, p_2)$

Axiom 2

$\forall e \in S$(VT_EVENTUALITY),

VT_EVENT(e, time_point)$\Rightarrow$ VT_EVENTUALITY(e, hold) = to_element (VT_EVENT(e, time_point)) $\wedge$
VT_STATE (e, maximal_period) $\Rightarrow$ VT_EVENTUALITY(e, hold) = VT_STATE(e, maximal_period) $\wedge$
VT_ACCOMPLISHMENT (e, telic_period) $\Rightarrow$
VT_EVENTUALITY(e, hold) = VT_ACCOMPLISHMENT(e, telic_period)

a) VT_Down

$\forall e$, p_1, p_2, VT_STATE(e) $\wedge$ VT_STATE.hold(e, p_1) $\wedge$ $p_2 \subseteq p_1 \Rightarrow$ VT_STATE.hold(e, p_2)

b) VT_Up

$\forall e$, p_1, p_2, VT_STATE(e) $\wedge$ VT_STATE.hold(e, p_1) $\wedge$ VT_STATE.hold(e, p_2) $\wedge$
(MEETS(p_1, p_2) $\vee$ MEETS^{-1}(p_1, p_2) $\vee$ OVERLAPS(p_1, p_2) $\vee$ OVERLAPS^{-1}(p_1, p_2)
$\vee$ FINISHED(p_1, p_2) $\vee$ FINISHED^{-1}(p_1, p_2) $\vee$ DURING(p_1, p_2)
$\vee$ DURING^{-1}(p_1, p_2) $\vee$ STARTS(p_1, p_2) $\vee$ STARTS^{-1}(p_1, p_2) $\vee$ EQUALS(p_1, p_2)
) $\Rightarrow$ VT_STATE.hold(e, $p_1 \cup p_2$)

Axiom 3

a) $\forall e \in S$ (CONTRACTOR),
CONTRACTOR.VT_STATE.VT_EVENTUALITY.VT_has.TEMPORAL_GRANULARITY(e, name) = day

b) $\forall e \in S$ (PROJECT MANAGER),
PROJECT MANAGER.VT_STATE.VT_EVENTUALITY.VT_has.TEMPORAL_GRANULARITY(e, name) = day

c) $\forall e \in S$ (meets with),
meets with.VT_ACCOMPLISHMENT.VT_EVENTUALITY.VT_has.TEMPORAL_GRANULARITY(e, name) =
minute

d) $\forall e \in S$ (reports to),
reports to.VT_STATE.VT_EVENTUALITY.VT_has.TEMPORAL_GRANULARITY(e, name) = day

Axiom 4

a) $\forall e_1 \in S$(PROJECT MANAGER), $\forall e_2 \in S$(name), SEM_ATTR_PROJECT MANAGER_name (e_1, e_2) $\Rightarrow$ ATTR_OF (e_2, e_1)

b) $\forall e_1 \in S$(PROJECT MANAGER), $\forall e_2 \in S$(phone), SEM_ATTR_PROJECT MANAGER_phone (e_1, e_2)
$\Rightarrow$ ATTR_OF (e_2, e_1)

c) $\forall e_1 \in S$(CONTRACTOR), $\forall e_2 \in S$(name), SEM_ATTR_CONTRACTOR_name (e_1, e_2) $\Rightarrow$ ATTR_OF (e_2, e_1)

d) $\forall e_1 \in S$(CONTRACTOR), $\forall e_2 \in S$(phone), SEM_ATTR_CONTRACTOR_phone (e_1, e_2) $\Rightarrow$ ATTR_OF (e_2, e_1)

e) $\forall e$, a, p_1, e.VT_STATE $\wedge$ (e.VT_STATE(hold) = p_1) $\wedge$ ATTR_OF(a, e) $\wedge$
(a.annotations$\neq$Acc $\wedge$ a.annotations$\neq$E $\wedge$ a.annotations$\neq$S)
$\Rightarrow a \in$ S(VT_STATE) $\wedge$ a.VT_STATE(hold) = p_1

In summary, our proposed approach employs the annotated schema for capturing telic/atelic semantics. On the other hand, the translated USM schema is used to formally define the semantics of the annotated constructs.

7. Contributions and Future Research

This research presents an approach for temporal conceptual design that helps to capture telic/atelic data semantics: 1) first capture "what" semantics using a conventional conceptual model (see, for example, Figure 4); 2) then employ annotations to differentiate between telic/atelic "when" semantics (see, for example, Figure 6); 3) finally, specify temporal constraints, specifically non-sequenced semantics, in the temporal data dictionary (see, for example, Table 2). In presenting an approach for capturing temporal data semantics, we *generalize* the semantics of concepts in a conventional conceptual model. Such an approach presents the temporal semantics *in terms of* a non-temporal conventional conceptual model (for example, the ER Model). We showed how the annotated schema captures rich semantics and that the semantics of the annotated schema can be "unwrapped." For example, see the translated USM schema as well as the automatically generated axioms in Section 6 that explicate the semantics of a small fragment of the (shaded) annotated schema. While the annotated schema can be employed to capture telic/atelic data semantics, the "unwrapped" schema (along with axioms) that explicates the semantics of the annotations can be used for developing a logical schema.[8] In Appendix A, we first present evaluation

[8] Such logical mapping would depend on the temporal support provided by the logical model and is outside the scope of this paper.

criteria that are based on linguistic goals of syntactic, sematic and pragmatic quality. We then evaluate the proposed approach in Appendix B. In this section, we present the contributions of our research and conclude with the implications of our research for both future research and for practice.

With respect to prior approaches, and, in particular ST-USM [36, 37], this work makes three key contributions. First, we include both telic/atelic and sequenced/non-sequenced dichotomies, thus, enhancing the expressiveness of the prior approach. Both dichotomies (telic/atelic and sequenced/non-sequenced) are important in temporal databases. We showed that they are orthogonal since the telic/atelic distinction concerns the temporal properties of entity classes, attributes, relationships and superclass-subclass per se, while the sequenced/non-sequenced distinction concerns the entity class-attribute/entity class-relationship pair. That said, these two distinctions interact through the semantics of composition. For example, a telic durational entity class with an atelic attribute is considered to be coerced to atelic to affect the sequenced constraint of identical temporal extent. This is in contrast to ST-USM [36, 37] as well as the prior other approaches in the literature (see, for example, [48, 66, 67]), in which atelic entity classes and properties and sequenced relationships are included. Thus the proposed approach generalizes the semantics of concepts in a conventional conceptual model.

Second, while our proposed approach is more expressive, it maintains the succinctness of the prior proposal [36, 37] because of two features: compositionality and inheritance. To keep the semantics manageable and clear, they need to be modeled in a compositional way. In our approach, the semantics of the entire conceptual schema can be split into the semantics of each entity class/attribute/relationship alone and entity class-attribute/ entity class-relationship pair considered separately. (Note that such a compositional approach perfectly fits with the orthogonality of the telic/atelic and sequenced-non-sequenced distinctions. This enhances the clarity and the compactness of the proposed approach.)

Third, to further enhance compactness and clarity, we have used inheritance throughout. In particular, in the proposed approach, we specify the semantics of a taxonomy of few "basic" entity classes (e.g., VT_STATE, VT_ACCOMPLISHMENT, VT_EVENT), so that the semantics of specific entity classes and properties in a conceptual model is simply obtained by inheritance by classifying them along the taxonomy. In such a way, we provide "direct" semantics to the annotations, "S", "Acc" and "E," which implies that the annotated entity class has to be classified as state, accomplishment and event, respectively, thus, inheriting the corresponding (implicit) temporal properties.

Our work has several implications for future research. First, we have briefly mentioned the notion of implicit constraints and of temporal propagation. Future research should provide algorithms to propagate such constraints, e.g., in order to make implied constraints explicit and/or to check the consistency of explicit and implicit constraints; see, for example, a survey [71] that describes temporal constraints propagation algorithms for temporal constraints.

Second, we focused on a specific type of temporal constraints, i.e., non-sequenced temporal constraints. Future research should investigate temporal constraints in the context of cardinality constraints (see, for example, [22]).

Third, while this research focused on timestamping—i.e., distinguishing temporal and non-temporal concepts—that differentiates two types of durative facts (telic/atelic), future research should explore how this differentiation influences evolution constraints that apply to classes (*status* and *transition constraints*) and relationships (*generation* and *cross-time constraints*) (see, for example, [10, 11]).

Fourth, the role of telic/atelic distinction in spatial data needs to be explored in the future. While several spatial conceptual models have been proposed (see, for example, [37, 48, 66, 67]), telic/atelic aspects have not been included in these models.

Fifth, while we presented a mechanism to capture non-sequenced temporal semantics, future research needs to similarly explore non-sequenced spatial semantics. Similar to temporal constraints that are based on Allen's predicates, spatial constraints that are based on topological constraints, such as meets, equals, inside and covers, can be specified [47]. Additionally, the role of the telic/atelic distinction in spatial queries needs to be examined further; this distinction has been considered for temporal queries [64]. Additionally, it would be helpful to determine which kinds of non-sequenced constraints are useful, and whether these constraints might propagate, as do the sequenced constraints (cf. compositionality).

Sixth, future research needs to explore how a design support environment can support the elicitation of telic/atelic data semantics. For example, in a design-support environment, clicking on a line could bring up a dialog box that would allow a constraint to be specified, which would then render that pair non-sequenced. Via a proof-of-concept design support environment, prior research has shown how the annotation-based approach can be embedded in extant design-support environment [38]; such an environment integrates with existing database design methodologies and results in upward compatible conceptual as well as XML schemas.

Sixth, while prior research—based on schema understanding (see, for example, [16])—suggests that annotations should be placed *in* the schema [41] rather than *outside* the schema, further research should examine how the annotations can best be rendered in the schema, including evaluating other possible graphical and textual representations. Further research should also evaluate the effect of annotations on the development (rather than schema understanding) of temporal conceptual schema (see, for example, [43]).

8. CONCLUSION

In this paper, we present a temporal conceptual modeling approach that generalizes the semantics of concepts in a conventional conceptual model. Our proposed approach differentiates between facts with goal-related semantics (telic) from those that are intrinsically devoid of culmination (atelic). Using an example, we illustrate how failing to distinguish between telic and atelic semantics can affect the interpretation of the "content" of the database. Thus, capturing the meaning of the telic/atelic data, or, data semantics, can in turn affect the overall use of the database. In this work, we have extended a prior annotation-based approach [36, 37] to provide a mechanism to help capture telic/atelic data semantics. We also formally discuss additional sequenced and non-sequenced constraints that can be captured during conceptual design, thus significantly extending the types of temporal data semantics that can be captured during conceptual modeling and enabling temporal applications to be designed within existing practice.

Acknowledgments

This work was supported in part by the National Science Foundation under grants IIS-0803229, IIS-0415101, IIS-0455993, IIS-1016205, and EIA-0080123 and by partial support from a grant from Microsoft Corporation.

References

[1] "Information technology -- Database languages -- SQL -- Part 2: Foundation (SQL/Foundation)," International Standards Organization, December 2011.
[2] J. F. Allen, "Maintaining knowledge about temporal intervals," *Communications of the ACM* (26: 11), pp. 832-843, 1983.
[3] J. F. Allen, "Toward a General Theory of Action and Time," *Artificial Intelligence* (23: 2), pp. 123-154, 1984.
[4] J. F. Allen, "Time and time again: The many ways to represent time," *International Journal of Intelligent Systems* (6: 4), pp. 341-356, 1991.
[5] J. R. Anderson and G. H. Bower, *Human Associative Memory*. Washington, DC: V. H. Winston & Sons, 1973.
[6] Aristotle, *Categories. On Interpretation. Prior Analytics*. Cambridge, MA: Harvard University Press.
[7] A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev, "Reasoning over Extended ER Models," in *26th International Conference on Conceptual Modeling (ER-07)*, Auckland, New Zealand, 2007.
[8] A. Artale, E. Franconi, and F. Mandreoli, "Description logics for modelling dynamic information," in *Logics for emerging applications of databases*, 2003.
[9] A. Artale, E. Franconi, F. Wolter, and M. Zakharyaschev, "A Temporal Description Logic for Reasoning over Conceptual Schemas and Queries," in *8th European Conference on Logics in Artificial Intelligence (JELIA-02)*, Cosenza, Italy, 2002.
[10] A. Artale, C. Parent, and S. Spaccapietra, "Modeling the evolution of objects in temporal information systems " in *4th Int. Symposium on Foundations of Information and Knowledge Systems (FoIKS-06)*, 2006, pp. 22-42.
[11] A. Artale, C. Parent, and S. Spaccapietra, "Evolving objects in temporal information systems," *Annals of Mathematics and Artificial Intelligence* (50), pp. 5-38, 2007.
[12] C. Batini, S. Ceri, and S. B. Navathe, *Conceptual Database Design: An Entity-Relationship Approach*: Benjamin/Cummings Publishing Company, 1992.
[13] C. Bettini, C. E. Dyreson, W. S. Evans, R. T. Snodgrass, and X. S. Wang, "A Glossary of Time Granularity Concepts," in *Temporal Databases: Research and Practice*, O. Etzion, S. Jajodia, and S. Sripada, Eds., Springer-Verlag, 1998, pp. 406-413.
[14] C. Bettini, S. Jajodia, and S. X. Wang, *Time Granularities in Databases, Data Mining and Temporal Reasoning*. Berlin: Springer-Verlag, 2000.
[15] L. Bloom, L. Lifter, and J. Hafitz, "Semantics of Verbs and the Developments of Verb Inflection in Child Language," *Language* (52: 2), pp. 386-412, 1980.
[16] F. Bodart, A. Patel, M. Sim, and R. Weber, "Should Optional Properties Be Used in Conceptual Modelling? A Theory and Three Empirical Tests," *Information Systems Research* (12: 4), pp. 384-405, 2001.
[17] M. Böhlen and C. S. Jensen, "Sequenced Semantics " in *Encyclopedia of Database Systems*, L. Liu and T. Özsu, Eds., Springer, 2009.
[18] M. H. Böhlen, C. S. Jensen, and R. T. Snodgrass, "Temporal Statement Modifiers," *ACM Transactions on Database Systems* (25: 4), pp. 407-456, 2000.
[19] M. H. Böhlen, R. T. Snodgrass, and M. D. Soo, "Coalescing in Temporal Databases," in *Proceedings of 22th International Conference on Very Large Data Bases*, Mumbai (Bombay), India, 1996, pp. 180-191.
[20] P. P. Chen, "The Entity-Relationship Model - Toward a Unified View of Data," *ACM Transactions of Database Systems* (1: 1), pp. 9-36, 1976.
[21] P. P. Chen, "Suggested research directions for a new frontier -- active conceptual modeling," in *ER Conference*, Tucson, AZ, 2006, pp. 1-4.
[22] F. Currim, "A Framework for Modeling Business Rules in Conceptual Database Design," University of Arizona, PhD Dissertation, Tucson, 2004.
[23] D. R. Dowty, "The Effects of the Aspectual Class on the Temporal Structure of Discourse," *Linguistics and Philosophy* (9: 1), pp. 37-61, 1986.
[24] C. E. Dyreson, W. S. Evans, H. Lin, and R. T. Snodgrass, "Efficiently Supporting Temporal Granularities," *IEEE Transactions on Knowledge and Data Engineering* (12: 4), pp. 568-587, 2000.
[25] R. Elmasri and V. Kouramajian, "A Temporal Query Language for a Conceptual Model," in *Advanced Database Systems*, N. R. Adam and B. K. Bhargava, Eds., Berlin: Springer-Verlag, 1993, pp. 175-195.
[26] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, Sixth edition. Boston, MA: Addison Wesley, 2006.
[27] A. Gemino and Y. Wand, "Evaluating modeling techniques based on models of learning " *Communications of the ACM* (46: 10), pp.

79-84, 2003.

[28] A. Gemino and Y. Wand, "A Framework for Empirical Evaluation of Conceptual Modeling Techniques " *Requirements Engineering Journal* (9 4), pp. 248-260, 2004.

[29] H. Gregersen, "The formal semantics of the timeER model," presented at the Proceedings of the 3rd Asia-Pacific conference on Conceptual modelling - Volume 53, Hobart, Australia, 2006.

[30] H. Gregersen and C. Jensen, "Conceptual Modeling of Time-Varying Information," TIMECENTER Technical Report TR-35, September 10 1998.

[31] H. Gregersen and C. S. Jensen, "Temporal Entity-Relationship Models-A Survey," *IEEE Transactions on Knowledge and Data Engineering* (11: 3), pp. 464-497, 1999.

[32] B. Hailpern and P. Tarr, "Model-driven development: The good, the bad, and the ugly," *IBM Systems Journal* (45: 3), pp. 451-461, 2006.

[33] C. S. Jensen, C. E. Dyreson, M. Bohlen, J. Clifford, R. Elmasri, S. K. Gadia, F. Grandi, P. Hayes, S. Jajodia, W. Kafer, N. Kline, N. Lorentzos, Y. Mitsopoulos, A. Montanari, D. Nonen, E. Peresi, B. Pernici, J. F. Roddick, N. L. Sarda, M. R. Scalas, A. Segev, R. T. Snodgrass, M. D. Soo, A. Tansel, R. Tiberio, and G. Wiederhold, "A Consensus Glossary of Temporal Database Concepts-February 1998 Version," in *Temporal Databases: Research and Practice*, O. Etzion, S. Jajodia, and S. Sripada, Eds., Springer-Verlag, 1998.

[34] C. S. Jensen and R. T. Snodgrass, "Semantics of Time-Varying Attributes and their use for Temporal Database Design," TimeCenter Technical Report, Tucson, Arizona TR-1, January 29 1997.

[35] V. Khatri, S. Ram, and R. T. Snodgrass, "ST-USM: Bridging the Semantic Gap with a Spatio-temporal Conceptual Model," TimeCenter Technical Report TR-64, 2001.

[36] V. Khatri, S. Ram, and R. T. Snodgrass, "Supporting User-defined Granularities and Indeterminacy in a Spatiotemporal Conceptual Model," *Annals of Mathematics and Artificial Intelligence* (36: 1-2), pp. 195-232, 2002.

[37] V. Khatri, S. Ram, and R. T. Snodgrass, "Augmenting a Conceptual Model with Geospatiotemporal Annotations," *IEEE Transactions on Knowledge and Data Engineering* (16: 11), pp. 1324-1338, 2004.

[38] V. Khatri, S. Ram, and R. T. Snodgrass, "On Augmenting Database Design-Support Environments to Capture the Geo-Spatio-Temporal Data Semantics," *Information Systems* (31: 2), pp. 98-133 2006.

[39] V. Khatri, R. T. Snodgrass, and P. Terenziani, "Atelic data" in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu: Eds., Springer, 2009, pp. 142-143.

[40] V. Khatri, R. T. Snodgrass, and P. Terenziani, "Telic Distinction in Temporal Databases " in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu: Eds., Springer 2009, pp. 2911-2914.

[41] V. Khatri, I. Vessey, S. Ram, and V. Ramesh, "Cognitive Fit between Conceptual Schemas and Internal Problem Representations: The Case of Geospatio-Temporal Conceptual Schema Comprehension," *IEEE Transactions on Professional Communication* (49: 2), pp. 109-127, 2006.

[42] J. Kim, J. Hahn, and H. Hahn, "How Do We Understand a System with (So) Many Diagrams? Cognitive Integration Processes in Diagrammatic Reasoning," *Information Systems Research* (11: 3), pp. 284-303, 2000.

[43] Y.-G. Kim and S. T. March, "Comparing Data Modeling Formalisms," *Communications of the ACM* (38: 6), pp. 103-115, 1995.

[44] W. Klein, *Time in Language*. London: Routledge, 1994.

[45] O. I. Lindland, G. Sindre, and A. Solvberg, "Understanding Quality in Conceptual Modeling," *IEEE Software* (11: 2), pp. 42-49, 1994.

[46] M. Moens and M. Steedman, "Temporal Ontology and Temporal Reference," *Computational Linguistics* (14: 2), pp. 15-28, 1988.

[47] J. Nunes, "Geographic Space as a Set of Concrete Geographical Entities," in *NATO Advanced Study Institute on Cognitive and Linguistic Aspects of Geographic Space*, Las Navas del Marques, Spain, 1990.

[48] C. Parent, S. Spaccapietra, and E. Zimanyi, "Spatio-temporal conceptual models: Data structures + space + time," in *7th ACM Symposium on Advances in Geographic Information Systems*, Kansas City, USA, 1999.

[49] C. Parent, S. Spaccapietra, and E. Zimányi, "The MurMur project: Modeling and querying multi-representation spatio-temporal databases," *Information Systems* (31: 8), pp. 733-769, 2006.

[50] J. Parsons, "Effects of local versus global schema diagrams on verification and communication in conceptual data modeling," *Journal of Management Information Systems* (19: 3), pp. 155-183, 2002.

[51] J. Parsons and Y. Wand, "Choosing classes in conceptual modeling," *Commun. ACM* (40: 6), pp. 63-69, 1997.

[52] S. Ram, "Intelligent Database Design using the Unifying Semantic Model," *Information and Management* (29: 4), pp. 191-206, 1995.

[53] J. F. Roddick, "A Survey of Schema Versioning Issues for Database Systems," *Information and Software Technology* (37: 7), pp. 383-393, 1995.

[54] J. F. Roddick and R. T. Snodgrass, "Schema Versioning," in *The TSQL2 Temporal Query Language*, R. T. Snodgrass, Ed., Boston: Kluwer Academic Publishers, 1995, pp. 427-449.

[55] Y. Shoham, "Temporal Logics in AI: Semantical and Ontological Considerations" *Artificial Intelligence* (33:1), pp. 89-104, 1987.

[56] K. Siau and X. Tan, "Improving the quality of conceptual modeling using cognitive mapping techniques," *Data & Knowledge Engineering* (55: 3), pp. 343-365, 2005.

[57] A. Silbershatz, H. Korth, and S. Sudarshan, *Database System Concepts*, Third Edition, WCB/ McGraw Hill, 1997.

[58] R. T. Snodgrass, Ed., *The TSQL2 temporal query language*. Boston: Kluwer Academic Publishers, 1995.

[59] R. T. Snodgrass, *Developing Time-Oriented Database Applications in SQL*. San Francisco: Morgan Kaufmann Series in Data Management Systems, 1999.

[60] R. T. Snodgrass and I. Ahn, "Temporal Databases," *IEEE Computer* (19: 9), pp. 35-42, 1986.

[61] S. Spaccapietra, C. Parent, and E. Zimányi, "Modeling Time from a Conceptual Perspective," in *1998 ACM CIKM International Conference on Information and Knowledge Management*, Bethesda, Maryland, 1998.

[62] P. Terenziani, "Trattamento di informazioni temporali in sistemi basati sulla conoscenza: formalismo di rappresentazione e tecniche di ragionamento automatic," Dipartimento di Informatica University of Turin, PhD dissertation, 1993.

[63] P. Terenziani and L. Anselma, "A Knowledge Server for Reasoning about temporal constraints between classes and instances of

events," *International Journal of Intelligent Systems* (19: 10), pp. 919-948, 2004.

[64] P. Terenziani and R. T. Snodgrass, "Reconciling Point-based and Interval-based Semantics in Temporal Relational Databases: A Proper Treatment of the Telic/Atelic Distinction," *IEEE Transactions of Knowledge and Data Engineering* (16: 5), pp. 540-551, 2004.

[65] P. Terenziani, R. T. Snodgrass, A. Bottrighi, M. Torchio, and G. Molino, "Extending temporal databases to deal with telic/atelic medical data," *Artificial Intelligence in Medicine* (39: 2), pp. 113-126, 2007.

[66] N. Tryfona and C. S. Jensen, "Conceptual Data Modeling for Spatiotemporal Applications," *Geoinformatica* (3: 3), pp. 245-268, 1999.

[67] N. Tryfona and C. S. Jensen, "Using Abstractions for Spatio-Temporal Conceptual Modeling," in *2000 ACM Symposium on Applied Computing Applied*, Como, Italy, 2000.

[68] Z. Vendler, "Verbs and Times," in *Linguistics in Philosophy*, New York, NY: Cornell University Press, 1967, pp. 97-121.

[69] H. J. Verkuyl, *On the Compositional Nature of The Aspects* Dorderect Riedel, 1972.

[70] I. Vessey, "Cognitive Fit: A Theory-based Analysis of Graphs Vs. Tables Literature," *Decision Sciences* (22: 2), pp. 219-240, 1991.

[71] L. Vila, "A survey on temporal reasoning in artificial intelligence " *AI Communications* (7), pp. 4-28, 1994.

[72] B. L. Webber, "Special Issue on Tense and Aspect " *Computational Linguistics* (14: 2), pp. 1-2, 1988.

[73] Y. Wu, S. Jajodia, and X. S. Wang, "Temporal Database Bibliography Update," in *Temporal Databases: research and practice* O. Etzion, S. Jajodia, and S. Sripada, Eds., Springer Verlag, 1998, pp. 338-366.

[74] E. Zimanyi, C. Parent, S. Spaccapietra, and A. Pirotte, "TERC+: A Temporal Conceptual Model," in *International Symposium Digital Media Information Base*, 1997.

BIOGRAPHY

Vijay Khatri received the BE degree (electronics) from Malaviya National Institute of Technology in 1992, the management degree from the University of Bombay in 1994, and the PhD degree from the University of Arizona in 2002. He is an associate professor of information systems in the Operations and Decision Technologies department at Kelley School of Business in Indiana University. Vijay has published articles in journals such as *Annals of Mathematics and Artificial Intelligence*, *Information Systems Research*, *Journal of Management Information Systems*, *Decision Sciences Journal*, *Communications of the ACM*, *IEEE Transactions on Knowledge and Data Engineering*, *Information Systems*, and *Photogrammetric Engineering and Remote Sensing*. His research centers on issues related to data semantics, semiotics and conceptual database design, temporal databases, and spatial databases. More specifically, his research involves developing techniques for management of data, especially for applications that need to organize data based on time and space. He serves on the editorial review board of the *Journal of Data Management* and *Journal of Association of Information Systems*, and is an associate edtor at *Information Systems Research* and *MIS Quarterly*. He is a senior member of the the IEEE and IEEE Computer Society.

Sudha Ram received the BS degree in mathematics, physics, and chemistry from the University of Madras in 1979, PGDM from the Indian Institute of Management, Calcutta in 1981, and the PhD degree from the University of Illinois at Urbana-Champaign in 1985. She is the Eller Professor of Management Information Systems in the College of Business and Public Administration at the University of Arizona. Dr. Ram has published articles in such journals as *Communications of the ACM*, *ACM Transactions on Information Systems*, *IEEE Transactions on Knowledge and Data Engineering*, *Information Systems Research*, *Information Systems*, *MIS Quarterly*, and *Management Science*. Dr. Ram's research deals with modeling and analysis of databases for manufacturing, scientific, and business domains. Her research has been funded by IBM, NCR, US ARMY, NIST, US National Science Foundation, NASA, NIH, and ORD (CIA). Specifically, her research deals with interoperability among heterogeneous database systems, semantic modeling, data allocation, schema and view integration, intelligent agents for data management, and tools for database design. Dr. Ram serves on various editorial boards including *Information Systems Research*, *Journal of Database Management*, *Information Systems Frontiers*, and the *Journal of Systems and Software*. She is also the director of the Advanced Database Research Group at the University of Arizona. She is a member of the IEEE and the IEEE Computer Society

Richard T. Snodgrass received the BA degree in physics from Carleton College and the MS and PhD degrees in computer science from Carnegie Mellon University. He joined the University of Arizona in 1989, where he is a professor of computer science. He is an ACM fellow. Dr. Snodgrass was editor-in-chief of the *ACM Transactions on Database Systems*. He chairs the ACM SIGMOD Advisory Board and has served on the editorial boards of the *International Journal on Very Large Databases* and the *IEEE Transactions on Knowledge and Data Engineering*. He chaired the Americas program committee for the 2001 *International Conference on Very Large Databases*, chaired the program committees for the 1994 *ACM SIGMOD Conference* and the 1993 *International Workshop on an Infrastructure for Temporal Databases*, and was a vice-chair of the program committees for the 1993 and 1994 *International Conferences on Data Engineering*. He was ACM SIGMOD Chair from 1997 to 2001 and has previously chaired the ACM Publications Board and the ACM SIG Governing Board Portal Committee. He received the ACM SIGMOD Contributions Award in 2002. He chaired the TSQL2 Language Design Committee, edited the book, The TSQL2 Temporal Query Language (Kluwer Academic Press), and has worked with the ISO SQL3 committee to add temporal support to that language. He authored Developing Time-Oriented Database Applications in SQL (Morgan Kaufmann), was a coauthor of Advanced Database Systems (Morgan Kaufmann), and was a coeditor of Temporal Databases: Theory, Design, and Implementation (Benjamin/Cummings). He codirects Time-Center, an international center for the support of temporal database applications on traditional and emerging DBMS technologies. His research interests include temporal databases, query language design, query optimization and evaluation, storage structures, and database design. He is a senior member of the IEEE and the IEEE Computer Society.

Paolo Terenziani received both the Laurea (in 1987) and the PhD (in 1993) degrees in computer science from the Universita di Torino, Italy. He is currently a full professor with the Dipartimento di Informatica at the Universita del Piemonte Orientale "Amedeo Avogadro," Alessandria, Italy. His research interests mainly focus on the treatment of time and time dependent phenomena in different areas, including databases (in the subareas of temporal relational databases, and of semantics) and artificial intelligence (natural language, knowledge representation, temporal reasoning, constraint propagation techniques). He has published more than 60 papers about these topics in refereed journals and conferences.

APPENDIX

Appendix A

A precursor to augmenting a conventional conceptual model with telic/atelic data semantics is identifying the conceptual modeling requirements that need to be met. Prior research suggests employing four linguistic cornerstones—*modeling language* or *model*, *domain*, *schema*, and *audience interpretation*—to present three aspects of quality, *syntactic*, *semantic* and *pragmatic* [45].[9] Modeling language, or model, *M*, refers to all statements that can be made according to the syntax. Domain, *D*, refers to all correct statements that are relevant for solving the problem. The schema, *S*, is the set of statements actually made. Audience interpretation, *A*, is the set of statements that the stakeholders think the schema contains. This framework employs linguistic concepts to present "*what* we are trying to achieve" (goals) as well as "*how* to achieve it" (means); see Table 5.

Goal	Means
Syntactic quality The syntactic goal is *syntactic correctness,* that all statements in the schema are according to the model syntax, i.e., $S \backslash M = \phi$ [45].	• Lindland et al. [45] suggest that the means to avoiding syntax errors is by using a *formal syntax* such as Backus-Naur Form (BNF).
Semantic quality Two semantic goals are: 1) *Validity*, that all statements in the schema are correct and relevant: ($S \backslash D = \phi$) and 2) *Completeness,* that the schema contains all statements about the domain that are correct and relevant: ($D \backslash S = \phi$).	• Lindland et al. [45] suggest that *consistency checking* is the means to checking the semantics of the schema. • Another means for achieving the semantic goal is by ensuring upward compatibility. *Upward compatibility* Bohlen et al. [18] refer to the ability to render a conventional conceptual model temporal without impacting or negating the semantics of the non-historical "legacy schema," i.e., the schema developed using a conventional conceptual model; such a property of a temporal conceptual model protects investments in non-historical existing (or legacy) schemas. Upward compatibility requires that with the addition of temporal aspects, the syntax and semantics of the conventional conceptual model (for example, the ER Model [20]) remain unaltered.
Pragmatic quality The pragmatic goal is *comprehension* that is based on human cognition (see, for example, [16, 28, 42, 51, 56]), i.e., that all schema projections, S_i, are understood by their relevant audience interpretation projections, A_i. $\forall i$, $S_i = A_i$.	• Pragmatic means are whatever makes the schema *easier to understand*, in terms of, e.g., inspection, explanation or filtering. • Another means to the pragmatic goal is *expressive economy*, which implies that there is a smaller schema that needs to be inspected.

Table 5: Goals and means related to syntactic, semantic and pragmatic quality

[9] As compared with [45], we employ different terminology: in our context, the *modeling language* is referred to as the "model" and "the set of statements actually made" is referred to as the "schema." We employ the notations of [45].

Appendix B

Based on *goals* and *means* associated with syntactic, semantic and pragmatic quality of conceptual models [45] (see Section 4), we evaluate our proposed approach. In our proposed approach, the temporal annotations ("when") are orthogonal to ERAS (entity class-attribute-relationship-superclass/subclass, i.e., "what"). Thus, various goals for augmented (temporal) conceptual model may be stated as in Table 5:

- Syntactic goal: *syntax correctness* ($S \backslash M \cup S^{ta} \backslash M^{ta} = \phi$)
- Semantic goals: *validity* ($S \backslash D \cup S^{ta} \backslash D^{t} = \phi$) and *completeness* ($D \backslash S \cup D^{t} \backslash S^{ta} = \phi$)
- Pragmatic goal: *comprehension* ($\forall i$, ($S_i = A_i$) $\cap$ ($S_i^{ta} = A_i^{ta}$))

where S^{ta}, M^{ta}, D^{t}, and A_i^{ta} refers to temporal statements in the schema, temporal annotation syntax (see Figure 5), all correct temporal statements that are relevant for solving the problem, and a projection of audience interpretation of the temporal domain, respectively. Note how our approach ensures that syntactic, semantic and pragmatic quality of "what" (conventional conceptual model: S, D, M, A_i) is mutually independent of "when" (annotations: S^{ta}, M^{ta}, D^{t}, A_i^{ta}).

With respect to the means of achieving syntactic quality, we formally define the temporal annotation syntax in Backus-Naur Form. Additionally, we have employed first-order logic to define the temporal data semantics. Such an approach formalizes the syntax of the modeling language, and thus in return helps reduce ambiguity, which in turn can help improve schema comprehension.

Recall that two means of achieving the semantic goals are consistency checking and upward compatibility. We present a mechanism, referred to as semantics of composition (see Section 6.3), wherein the consistency of interactions between entity class-relationship and entity class-attribute can be validated; see also Table 3 for an example of valid interactions that are provided by our proposed approach. Since our temporal extension is a strict superset that is provided by adding non-mandatory semantics, the annotation-based temporal conceptual model is upward compatible with conventional conceptual models.

The means of achieving pragmatic quality is via ease of comprehension in terms of, e.g., inspection and expressive economy. Based on the general problem-solving approach of divide-and-conquer, the annotation-based approach divides conceptual design into two phases: first capture "what" semantics and then associate "when" semantics with "what." Prior research, based on cognitive fit [70] and human associative memory [5], suggests that annotations *in* the schema result in the matching of the external problem representation (schema) with internal task representation; on the other, annotations *outside* the schema result in a mismatch [41]. Thus, prior research suggests that the annotated schemas should be straightforward to comprehend with respect to inspection. With respect to expressive economy, various conventional conceptual modeling constructs are orthogonal to temporal annotations. Thus these annotations are generic, in the sense that they apply equally to the various conceptual modeling constructs, and are minimal, in that the same kind of annotation can apply to many different constructs, rather than having a different way of stating the "when" for each construct. All interactions are permitted except for those two "disallowed" in Table 3. For example, an annotation phrase "Acc(day)/-" can be applied to an entity class, an attribute or a relationship.

In summary, our proposed approach demonstrates syntactic, semantic and pragmatic quality. We next present implications of our research, both for practice and research.