Transition Constraints for Temporal Attributes

E.A. Nasubo Ongoma[1], C. Maria Keet[2], and Thomas Meyer[1]

[1] School of Mathematics, Statistics, and Computer Science, University of KwaZulu-Natal and UKZN/CSIR-Meraka Centre for Artificial Intelligence Research, South Africa,
212562258@stu.ukzn.ac.za, tmeyer@csir.co.za

[2] Department of Computer Science, University of Cape Town, South Africa,
mkeet@cs.uct.ac.za

Abstract. Representing temporal data in conceptual data models and ontologies is required by various application domains. For it to be useful for modellers to represent the information precisely and reason over it, it is essential to have a language that is expressive enough to capture the required operational semantics of the time-varying information. Temporal modelling languages have little support for temporal attributes, if at all, yet attributes are a standard element in the widely used conceptual modelling languages such as EER and UML. This hiatus prevents one to utilise a complete temporal conceptual data model and keep track of evolving values of data and its interaction with temporal classes. A rich axiomatisation of fully temporised attributes is possible with a minor extension to the already very expressive description logic language $\mathcal{DLR}_{\mathcal{US}}$. We formalise the notion of *transition* of attributes, and their interaction with transition of classes. The transition specified for attributes are extension, evolution, and arbitrary quantitative extension.

1 Introduction

Representing temporal information has been researched for over 20 years in diverse fields, including temporal conceptual models [17, 1, 16], logic-based representation [1, 4], informal business rules on time [12], spatio-temporal models [18] and Ontology Based Data Access [2, 6]. Logic-based representation of temporal data at the conceptual level provides a link between temporal conceptual data models and temporal description logic. One of the notable achievements is the temporally extended ER language called ER_{VT} that has a logical reconstruction in the description logic language $\mathcal{DLR}_{\mathcal{US}}$, which covers temporal behaviour of classes [1, 3] and relationships [14] and more recently also temporal attributes [15].

According to [11], temporal modelling addresses the question *how does the world change?* One way is as the life cycle, the evolving state of membership of an object from its start (creation) to end (deletion), covered in [3, 14, 15] for objects, relations and attributes respectively. The second way is transition, concerning the change of object properties over time as the object migrates from one class to the next. Research on transition constraints has covered only object migration [3] and relation migration [14]. However, to efficiently represent and reason over data, we need to also look at attribute transition, which is the migration of attributes along an object's life cycle. Temporal attributes have been used and studied less in comparison to classes and relations. Yet in databases, we have a direct contact with attributes, and if those (temporal) attributes are not represented properly in our conceptual models, one will face inconsistent databases with respect to the constraints that ought to hold.

This paper looks at temporal attributes as they evolve over time along an object's lifecycle. The formalisation is done using a minor extension of the temporal description logic $\mathcal{DLR}_{\mathcal{US}}$ with temporal attributes. This work relies on previous results on transitions of objects and relations for ER_{VT}. Using the constraints described in the previous work, we represent attribute migration as evolution constraints ADEV and ADEX, persistence constraints APEX and APEV and quantitative constraints AQEV and AQEX. We also give a more rigorous definition to arbitrary quantitative transition of attributes in time (instead of merely stating the next point) in time.

The paper is organised as follows. We first introduce a few motivating scenarios for temporal attributes in Section 2. The basics of transition and the modified description logic $\mathcal{DLR}_{\mathcal{US}}$ is described in Section 3. The new semantics for quantitative evolution as well as the transition constraints for attributes, with its interaction with evolving classes is described in Section 4. We discuss the results in Section 5 and conclude in Section 6.

2 Application Areas

Attribute migration is done in practice but it is not explicitly stated and formally represented. Our aim is to introduce the notion of evolving temporal attributes to stakeholders of the various application domains. Most businesses have as aim to reduce cost-to-company and by using transition constraints companies would reduce cost as well as eliminate human error in their databases. Transition constraints also permit companies to represent history and learn from it, for example policy makers in insurance companies make their decisions based from the client's history, reservation systems use temporal databases to manage booking and medical information systems to manage patient's records. Evolving temporal attributes can therefore be used in many application areas mapped as business rules to monitor, verify and ensure that the processes run as planned and also check consistency, by reflecting violations and generate new knowledge. System administrators would then have to write down the requirements for data-centric transitions and the objects and attributes in database would evolve after the set amount of time. Consider the following application areas:

- Security features put in place to prevent a breach in access and ensure safety and dependability, for example if an employee is promoted to a branch manager but he is given the access rights of a CEO due to underspecified change constraints on the data, then there would be a violation. The attributes can be given according to the employee ranks, (say from 1-6), let's say that an employee is given the lowest access, different managers with different access rights and the highest is the CEO.
- Administration would benefit from human resource business rules that permit and manage evolution of classes. For example having the promotion of an employee, if we had a business rule that all managers must have a masters degree and an employee is promoted without one, it will cause a violation in the database.

– Medical information systems to monitor the effect drugs on patients and the status of the patients. For instance, there are sensor readings for a particular patient and when it reaches a particular state, medical intervention is required, for example when a diabetes out-patient's blood sugar reaches a certain level, he needs to be hospitalised. It is not trivial to declare evolving constraints in the medical and healthcare field, but it is possible to monitor patients' changes (e.g., in the ICU), recorded with a set of attributes, and show evolution of the patient's states against the current state, raise alarms on extreme changes, and reclassify the patient according to the symptoms.
– Situational awareness, to monitor the changes in an environment and add meaning to this change. For example, recording the change in the levels (concentration) of a toxin in the environment, or even the change in climate zone designation of a region (recorded with attributes in a GIS).

The above examples show how transition constraints of attributes are used and why we need to properly represent them in order to capture the operational semantics governing the movement and change of data in a temporal conceptual model or ontology.

3 Preliminaries

We briefly introduce what has been done before on the life cycle, status classes and transitions. The life cycle can be viewed in three ways: by status classes [3], status relations [14] and status attributes [15] which records the membership of an object, relation, or attribute as it evolves along the statuses. Status models the normal behaviour of the life cycle in the real world, items are first scheduled, then become active, can be suspended and reactivated again and finally may become disabled, when they reached their expiry. These are the four statuses used when modelling data, which are represented in an EER diagram as shown in Fig. 1; the rectangular boxes are status classes (entity types) and the ovals are status attributes, which are 'housed' in classes. An active class (C) can have any of the four status attributes, while scheduled, suspended and disabled classes can only have their corresponding attributes, i.e. scheduled, suspended or disabled only.

Transition records migration of elements from source to target element, be they objects or relations as they move along their lifecycle. They control permissible changes from source class to target, to ensure that data does not enter an impossible state because of a previous state. These objects (relations) may lose membership from the source class (relation) or acquire other memberships in the target class (relation), which may give rise to a new classification scheme. Classes have both temporal and snapshot attributes, but only temporal attributes can participate in the migration, whereas snapshot attributes always remain the same.

Object transitions give constraints that capture the movement of an object from source class to target class. These transitions can be either an evolution or an extension, the difference being that an object in an evolution ceases to be a member of the source class while in an extension, the object is still a member of the source class. Dynamic evolution DEV and dynamic extension DEX were introduced in [3], while quantitative extension TEX, quantitative evolution TEV, persistent extension PEX, persistent evolution PEV were introduced in [5]. The parallel transition for relations are relation dy-

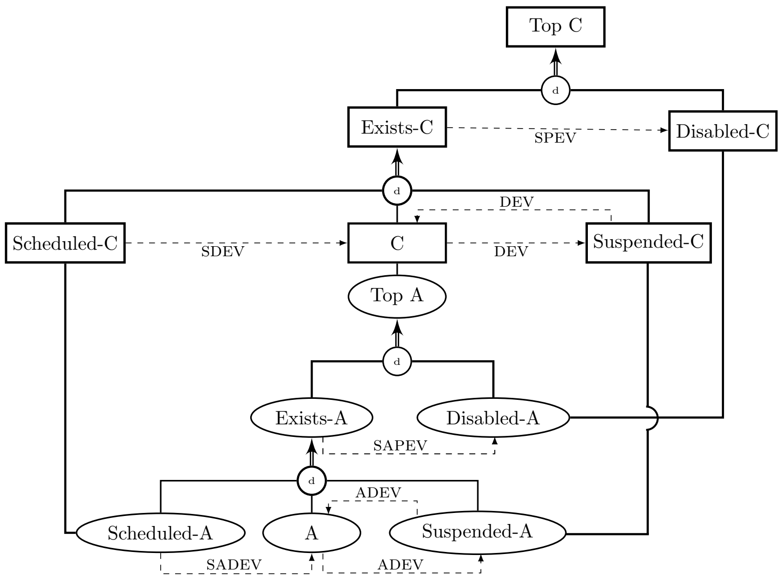

Fig. 1. EER diagram with the class hierarchy of status classes (rectangles) integrated with the attribute hierarchy of status attributes (ovals). The dashed arrows illustrate class migration and attribute migration.

namic evolution RDEV, dynamic extension for relations RDEX, quantitative extension for relations RQEX, quantitative evolution for relations RQEV, strong dynamic extension for relations SRDEX and strong dynamic extension for relations SRQEX transition relations were introduced in [14]. However, there is no support for attribute migration or on the effect of object migration on attributes, yet every object transition affects attributes.

3.1 $\mathcal{DLR}_{\mathcal{US}}$ with Attributes

The Description Logic $\mathcal{DLR}_{\mathcal{US}}$ [1] is an expressive fragment of FOL that combines the propositional temporal logic with *Since* and *Until* operators with the (non-temporal) description logic $\mathcal{DLR}$ [9] so that relationships and classes can be temporalised. We add a minor extension to $\mathcal{DLR}_{\mathcal{US}}$ to include a precise syntax and semantics for attributes, mainly because we are dealing with conceptual data models that require attributes. A temporary attribute was defined earlier in [3], as a binary relation for each attribute, $A \in \mathcal{A}$, for $\langle A, C\rangle \in \texttt{T}$, with its $\mathcal{DLR}_{\mathcal{US}}$ axiom as $\texttt{C} \sqsubseteq \neg\exists[\texttt{From}](\Box^{*}\texttt{A})$. This entailed that temporal constructors can be used in front of attributes, which, however, was not included in the $\mathcal{DLR}_{\mathcal{US}}$ syntax and semantics.

The syntax and semantics of the extended $\mathcal{DLR}_{\mathcal{US}}$ are included in Fig. 2. Details of $\mathcal{DLR}_{\mathcal{US}}$ can be found in [1, 3]; as usual, we have classes C (starting from atomic ones CN), n-ary relations R (DL roles, with $n \geq 2$, RN), binary attributes A between a class and a datatype, DL role components (U, of which F denotes a role component in an attribute, $F \subseteq U$, and $F = \{\texttt{From}, \texttt{To}\}$). The selection expression $U_i/n : C$

denotes an n-ary relation whose i-th argument ($i \leq n$) is of type C and $[U_j]R$ denotes the j-th argument ($j \leq n$)—DL role component, which can be seen intuitively as a projection over the role—in role R (subscripts i and j are omitted if it is clear from the context). For F, which concerns the DL role components in an attribute, we thus have $F : C$, with F denoting the role component From that relates to class C, and $[F]A$ denoting the role component F of A, where if To is used, it is the DL role component that associates with the datatype of the attribute, and if From is used, it is the DL role component that associates with the class of the attribute. Thus, for each $A \in \mathcal{A}$ and denoting with Literal the top for data types (i.e., for the domain of values $\Delta_D^{\mathcal{I}}$; see below), the $\mathcal{DLR_{US}}$ axiom $A \sqsubseteq \texttt{From} : \top \sqcap \texttt{To} : \texttt{Literal}$ holds. Finally, $\mathcal{U}$ntil and $\mathcal{S}$ince together with $\bot$ and $\top$ suffice to define the temporal operators: $\diamond^+$ (some time in the future) as $\diamond^+ C \equiv \top \mathcal{U} C$, $\oplus$ (at the next moment) as $\oplus C \equiv \bot \mathcal{U} C$, and likewise for their past counterparts; $\Box^+$ (always in the future) and $\Box^-$ (always in the past) are the duals of $\diamond^+$ and $\diamond^-$.

The model theoretic semantics of $\mathcal{DLR_{US}}$ assumes a flow of time $\mathcal{T} = \langle \mathcal{T}_p, < \rangle$, where $\mathcal{T}_p$ is a set of countably infinite time points also referred to as chronons and $<$ is isomorphic to the usual ordering on the integers. The language of $\mathcal{DLR_{US}}$ is interpreted in temporal models over $\mathcal{T}$, which are triples in the form $\mathcal{I} = \langle \mathcal{T}, \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}(t)} \rangle$, where $\Delta^{\mathcal{I}}$ is the union of two non empty disjoint sets, the *domain of objects*, $\Delta_O^{\mathcal{I}}$, and *domain of values*, $\Delta_D^{\mathcal{I}}$, and $\cdot^{\mathcal{I}(t)}$ the interpretation function such that, for every $t \in \mathcal{T}$ ($t \in \mathcal{T}$ will be used as a shortcut for $t \in \mathcal{T}_p$), every class C, and every n-ary relation R, we have $C^{\mathcal{I}(t)} \subseteq \Delta_O^{\mathcal{I}}$ and $R^{\mathcal{I}(t)} \subseteq (\Delta_O^{\mathcal{I}})^n$; also, $(u, v) = \{w \in \mathcal{T} \mid u < w < v\}$. Note that in [1, 3], $\Delta_D^{\mathcal{I}}$ was already used in the $\mathcal{DLR_{US}}$-based logic reconstruction of ER_{VT}, and $\Delta^{\mathcal{I}} = \Delta_O^{\mathcal{I}} \cup \Delta_D^{\mathcal{I}}$, but this was not explicitly stated in the $\mathcal{DLR_{US}}$ syntax and semantics; it is now in Fig. 2.

A *knowledge base* is a finite set Σ of $\mathcal{DLR_{US}}$ axioms of the form $C_1 \sqsubseteq C_2$ and $R_1 \sqsubseteq R_2$, and with R_1 and R_2 being relations of the same arity. An interpretation $\mathcal{I}$ satisfies $C_1 \sqsubseteq C_2$ ($R_1 \sqsubseteq R_2$) if and only if the interpretation of C_1 (R_1) is included in the interpretation of C_2 (R_2) at all time, i.e. $C_1^{\mathcal{I}(t)} \subseteq C_2^{\mathcal{I}(t)}$ ($R_1^{\mathcal{I}(t)} \subseteq R_2^{\mathcal{I}(t)}$), for all $t \in \mathcal{T}$.

4 Formalising Transition of Attributes

Status attributes are introduced in [15], which reuse the idea of status classes from [3], and its hierarchy is shown in Fig. 1. It has a rigorous formalisation that specifies the operational semantics of temporal attributes, to control the movement and permissible transitions of attributes, though we commit to the intention of suspension, such that suspended classes cannot participate in a transition. Previous formalisation of evolution constraints DEX and DEV in [3] asserted that both suspended classes and active classes can participate in a transition. This violates its ISA2 proposition (objects suspended in a subclass must also be suspended or active in the superclass), which, if permitted, would allow other operations on suspended entities and to participate in transitions, e.g., one should not permit an extension of an object from a Suspended-Employee to an Active-Manager if Manager $\sqsubseteq$ Employee. The class must first be active before any transition.

In addition, we add the notion of *arbitrary* quantitative evolution going beyond one chronon (the next point $(t + 1)$). The essence here is to have a way to represent fixed

$$
\begin{aligned}
C \rightarrow\ & \top \mid \bot \mid \mathit{CN} \mid \neg C \mid C_1 \sqcap C_2 \mid \exists^{\leqslant k}[U_j]R \mid \exists[\mathbf{F}]\mathbf{A} \mid \\
& \Diamond^+ C \mid \Diamond^- C \mid \Box^+ C \mid \Box^- C \mid \oplus C \mid \ominus C \mid C_1\ \mathcal{U}\, C_2 \mid C_1\ \mathcal{S}\, C_2 \\
R \rightarrow\ & \top_n \mid \mathit{RN} \mid \neg R \mid R_1 \sqcap R_2 \mid U_i/n : C \mid \\
& \Diamond^+ R \mid \Diamond^- R \mid \Box^+ R \mid \Box^- R \mid \oplus R \mid \ominus R \mid R_1\ \mathcal{U}\, R_2 \mid R_1\ \mathcal{S}\, R_2 \\
\mathbf{A} \rightarrow\ & \top_{\mathbf{A}} \mid \mathbf{AN} \mid \neg\mathbf{A} \mid \mathbf{F} : \mathbf{C} \mid \\
& \Diamond^+ \mathbf{A} \mid \Diamond^- \mathbf{A} \mid \Box^+ \mathbf{A} \mid \Box^- \mathbf{A} \mid \oplus \mathbf{A} \mid \ominus \mathbf{A} \mid \mathbf{A_1}\ \mathcal{U}\, \mathbf{A_2} \mid \mathbf{A_1}\, \mathcal{S}\, \mathbf{A_2}
\end{aligned}
$$

$$
\begin{aligned}
\top^{\mathcal{I}(t)} &= \Delta^{\mathcal{I}}_O \\
\bot^{\mathcal{I}(t)} &= \emptyset \\
\mathit{CN}^{\mathcal{I}(t)} &\subseteq \top^{\mathcal{I}(t)} \\
(\neg C)^{\mathcal{I}(t)} &= \top^{\mathcal{I}(t)} \setminus C^{\mathcal{I}(t)} \\
(C_1 \sqcap C_2)^{\mathcal{I}(t)} &= C_1^{\mathcal{I}(t)} \cap C_2^{\mathcal{I}(t)} \\
(\exists^{\leqslant k}[U_j]R)^{\mathcal{I}(t)} &= \{\ o \in \top^{\mathcal{I}(t)} \mid \sharp\{\langle o_1,\ldots,o_n\rangle \in R^{\mathcal{I}(t)} \mid o_j = o\} \leqslant k\} \\
(\exists\,[\mathbf{F}]\mathbf{A}^{\mathcal{I}(\mathbf{t})} &= \{\ \mathbf{o} \in \top^{\mathcal{I}(\mathbf{t})} \mid \sharp\{\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A}^{\mathcal{I}(\mathbf{t})} \geq \mathbf{1}\}\} \\
(C_1\, \mathcal{U}\, C_2)^{\mathcal{I}(t)} &= \{\ o \in \top^{\mathcal{I}(t)} \mid \exists v > t\,.(o \in C_2^{\mathcal{I}(v)} \wedge \forall w \in (t,v)\,.\,o \in C_1^{\mathcal{I}(w)})\} \\
(C_1\, \mathcal{S}\, C_2)^{\mathcal{I}(t)} &= \{\ o \in \top^{\mathcal{I}(t)} \mid \exists v < t\,.(o \in C_2^{\mathcal{I}(v)} \wedge \forall w \in (v,t)\,.\,o \in C_1^{\mathcal{I}(w)})\} \\
(\top_n)^{\mathcal{I}(t)} &= (\Delta^{\mathcal{I}}_O)^n \\
\mathit{RN}^{\mathcal{I}(t)} &\subseteq (\top_n)^{\mathcal{I}(t)} \\
(\neg R)^{\mathcal{I}(t)} &= (\top_n)^{\mathcal{I}(t)} \setminus R^{\mathcal{I}(t)} \\
(R_1 \sqcap R_2)^{\mathcal{I}(t)} &= R_1^{\mathcal{I}(t)} \cap R_2^{\mathcal{I}(t)} \\
(U_i/n : C)^{\mathcal{I}(t)} &= \{\ \langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid o_i \in C^{\mathcal{I}(t)}\} \\
(R_1\, \mathcal{U}\, R_2)^{\mathcal{I}(t)} &= \{\ \langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v > t\,.(\langle o_1,\ldots,o_n\rangle \in R_2^{\mathcal{I}(v)} \wedge \\
&\qquad \forall w \in (t,v)\,.\ \langle o_1,\ldots,o_n\rangle \in R_1^{\mathcal{I}(w)})\} \\
(R_1\, \mathcal{S}\, R_2)^{\mathcal{I}(t)} &= \{\ \langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v < t\,.(\langle o_1,\ldots,o_n\rangle \in R_2^{\mathcal{I}(v)} \wedge \\
&\qquad \forall w \in (v,t)\,.\ \langle o_1,\ldots,o_n\rangle \in R_1^{\mathcal{I}(w)})\} \\
(\Diamond^+ R)^{\mathcal{I}(t)} &= \{\langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v > t\,.\ \langle o_1,\ldots,o_n\rangle \in R^{\mathcal{I}(v)}\} \\
(\oplus R)^{\mathcal{I}(t)} &= \{\langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle o_1,\ldots,o_n\rangle \in R^{\mathcal{I}(t+1)}\} \\
(\Diamond^- R)^{\mathcal{I}(t)} &= \{\langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v < t\,.\ \langle o_1,\ldots,o_n\rangle \in R^{\mathcal{I}(v)}\} \\
(\ominus R)^{\mathcal{I}(t)} &= \{\langle o_1,\ldots,o_n\rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle o_1,\ldots,o_n\rangle \in R^{\mathcal{I}(t-1)}\} \\
(\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} &= \Delta^{\mathcal{I}}_O \times \Delta^{\mathcal{I}}_D \\
\mathbf{AN}^{\mathcal{I}(\mathbf{t})} &\subseteq (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \\
(\mathbf{F} : \mathbf{C})^{\mathcal{I}(t)} &= \{\ \langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \mathbf{o} \in \mathbf{C}^{\mathcal{I}(\mathbf{t})}\} \\
(\mathbf{A_1}\, \mathcal{U}\, \mathbf{A_2})^{\mathcal{I}(t)} &= \{\ \langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \exists \mathbf{v} > \mathbf{t}\,.(\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A_2}^{\mathcal{I}(\mathbf{v})} \wedge \forall \mathbf{w} \in (\mathbf{t},\mathbf{v})\,.\,\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A_1}^{\mathcal{I}(\mathbf{w})})\} \\
(\mathbf{A_1}\, \mathcal{S}\, \mathbf{A_2})^{\mathcal{I}(t)} &= \{\ \langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \exists \mathbf{v} < \mathbf{t}\,.(\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A_2}^{\mathcal{I}(\mathbf{v})} \wedge \forall \mathbf{w} \in (\mathbf{v},\mathbf{t})\,.\,\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A_1}^{\mathcal{I}(\mathbf{w})})\} \\
(\Diamond^+ \mathbf{A})^{\mathcal{I}(t)} &= \{\langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \exists \mathbf{v} > \mathbf{t}\,.\,\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A}^{\mathcal{I}(\mathbf{v})}\} \\
(\oplus \mathbf{A})^{\mathcal{I}(t)} &= \{\langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A}^{\mathcal{I}(\mathbf{t+1})}\} \\
(\Diamond^- \mathbf{A})^{\mathcal{I}(t)} &= \{\langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \exists \mathbf{v} < \mathbf{t}\,.\,\langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A}^{\mathcal{I}(\mathbf{v})}\} \\
(\ominus \mathbf{A})^{\mathcal{I}(t)} &= \{\langle \mathbf{o},\mathbf{d}\rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(\mathbf{t})} \mid \langle \mathbf{o},\mathbf{d}\rangle \in \mathbf{A}^{\mathcal{I}(\mathbf{t-1})}\}
\end{aligned}
$$

Fig. 2. Syntax and semantics of $\mathcal{DLR}_{\mathcal{US}}$, modified to include attributes (in bold face); o denote objects, d domain values, $v, w, t \in \mathcal{T}$, F is a role component in an attribute.

time changes e.g. transition after 3 years. We provide a generalisation of the next time as $(t + x)$. We use the subsumption notion that, if we have a class $A \sqsubseteq \oplus B$ and $B \sqsubseteq \oplus C$, we have $A \sqsubseteq \oplus \oplus C$, which can also be written in shorthand notation (i.e. syntactic sugar that does not affect the computational complexity) as $A \sqsubseteq \oplus_2 C$. This enables us to set the number of chronons before a class evolution occurs.

An integrated picture of status classes and status attributes is shown in Fig. 1, the dashed lines showing transition of objects and attributes.

4.1 Attribute Migration

Attribute migration occurs when an attribute in the same object object migrates to another attribute, for example the attributes *has_degree* and *has_postgrad*, is a transition from one attribute to another. Attribute migration is more complex than object migration because it is bidirectional, i.e., it can cause migration of objects, triggering reclassification of objects as well as participate in an object migration. For example, an HIV patient becoming an AIDS patient after his CD4 count (attribute) falls below 180 or a bank account (class) being frozen, due to expiry of a work permit.

Representation of attribute hierarchies is uncommon, mainly because it is not properly defined and formalised, but is relevant for modelling temporal data. New results on attribute hierarchies with subsumption in [15] allow us to capture the interaction between the permissible statuses of classes and status attributes in temporal transitions. A proper representation of transition in the conceptual model will enable a modeller to know how to design a temporal database.

We introduce attribute migration with their axioms in the extended $\mathcal{DLR}_{\mathcal{US}}$, building up work done on relation migration [14] and object migration [3, 5]. In attribute migration we look at the values which may change with time. ER diagrams do not have the capability of representing values on the diagrams but this can be modified by adding the value alongside the attribute. ORM [13] has "attribute free" diagrams, but each value type has a mapped_to attribute to a datatype, while UML [8] displays the attribute, datatype, and optionally permitted values inside the class in the diagram.

We discuss attribute transition with their examples and give their description logic semantics below, ordered along dynamic constraints, quantitative constraints, and persistence constraints. Our assumption is that all the attributes exist in the class and are active until they are disabled.

1. Dynamic constraints - these constraints model how attribute migration occurs in an object, from the source attribute to the target attribute. We have three types of dynamic transitions:

   ADEX: Attribute dynamic extension, which occurs when the attributes are still part of the source attribute as it migrates to the target attribute, for example the number of degrees you have can only increase over time, from an undergraduate degree to masters to phd. This can be modelled as the attributes has_degree for BSc, and has_postgrad for MSc and PhD

   (ADEX) *Dynamic extension of an attribute*

   $$\text{ADEX}_{\mathtt{A}_1,\mathtt{A}_2} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \Diamond^+(\mathtt{A}_1 \sqcap \mathtt{A}_2)$$

   ADEV: Attribute dynamic evolution occurs when an attribute migrates from the source attribute to a target attribute, but ceases to be a member of the source attribute. The attribute changes from the previous one, for example when changing the name of a product, an attribute new_name is added, or for a particular copyright (a has_copyright attribute) on creative work to evolve to public domain (pub_domain, e.g., as boolean).

   (ADEV) *Dynamic evolution of an attribute*

   $$\text{ADEV}_{\mathtt{A}_1,\mathtt{A}_2} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \Diamond^+(\neg\mathtt{A}_1 \sqcap \mathtt{A}_2)$$

   SADEV: Strong attribute dynamic evolution is a subclass of dynamic attribute evolution, which occurs when the source attribute can never go back to the source attribute, for example a scheduled attribute can never become scheduled after it has become active.

   (SADEV) *Strong Dynamic evolution of an attribute*

   $$\text{SADEV}_{\mathtt{A}_1,\mathtt{A}_2} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \Diamond^+\mathtt{A}_2 \sqcap \Box^+\neg\mathtt{A}_1$$

2. Quantitative constraints - these constraints specify the exact amount of time an attribute transition occurs from the source attribute to the target attribute. We use the generalisation of the next time, $(t + x)$, for quantitative constraints.

   AQEX: Attribute quantitative extension occurs when an attribute is set to migrate after a specified amount of time and the attribute still remains a member of the source attribute, for example, the modeller may specify that after the probationary period, say 6 months, a manager starts earning a full salary, whereas before he earned, say, 90% of his stipulated salary.

   (AQEX) *Quantitative extension of an attribute*
   $\text{AQEX}_{\mathtt{A_1},\mathtt{A_2}} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \oplus_{\mathtt{n}}(\mathtt{A}_1 \sqcap \mathtt{A}_2)$

   AQEV occurs when attributes migrate after a given time period but are no longer members of the source attribute, for example, an employee receiving a different bonus every 2 years.

   (AQEV) *Quantitative evolution of an attribute*
   $\text{AQEV}_{\mathtt{A_1},\mathtt{A_2}} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \oplus_{\mathtt{n}}(\neg\mathtt{A}_1 \sqcap \mathtt{A}_2)$

   SAQEV occurs when attributes migrate after a set period of time and do not go back to their previous state.

   (SAQEV) *Strong Quantitative evolution of an attribute*
   $\text{SAQEV}_{\mathtt{A_1},\mathtt{A_2}} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \oplus_{\mathtt{n}}(\neg\mathtt{A}_1 \sqcap \mathtt{A}_2)$

3. Persistence constraints - these constraints specify the persistent, non changing, state of an attribute after it has migrated from the source attribute to the target attribute. Once an attribute migrates, its value never changes.

   APEX: Persistent attribute extension occurs when the attribute migrates, but will never change it value in the future, and is still a member in the source attribute.

   (APEX) *Persistent extension of an attribute*
   $\text{APEX}_{\mathtt{A_1},\mathtt{A_2}} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \Box^+(\mathtt{A}_1 \sqcap \mathtt{A}_2)$

   APEV: Persistent attribute evolution occurs when the attribute migrates, and ceases to be a member of the source attribute and its value remains constant from that point on.

   (APEV) *Persistent evolution of an attribute*
   $\text{APEV}_{\mathtt{A_1},\mathtt{A_2}} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \Box^+(\neg\mathtt{A}_1 \sqcap \mathtt{A}_2)$

   SAPEV occurs when the attribute migrates but it never changes its value after the migration. For example when an attribute is disabled, it will never be active, it will always persist in the disabled state.

   (SAPEV) *Strong Persistent evolution of an attribute*
   $\text{SAPEV}_{\mathtt{A_1},\mathtt{A_2}} \sqsubseteq \mathtt{A}_1 \sqcap \neg\mathtt{A}_2 \sqcap \Box^+(\neg\mathtt{A}_1 \sqcap \mathtt{A}_2)$

4.2 Interaction between Transition of Classes and Attributes

Object migration affects only temporal classes and as a result temporal attributes are affected. The interaction between migrating temporal classes and temporal attributes brings forth two cases. *CASE A* in which object migration induces an attribute migration, and vice versa, *CASE B*. Although some aspects cannot be formalised in $\mathcal{DLR}_{\mathcal{US}}$, it is useful to at least consider the scenarios.

CASE A: The object migration causes an attribute migration such that the attributes move from the source to the target class. An example for *CASE A* is illustrated in Fig. 3, where dashed lines show examples of transitions on the integrated status classes and status attributes. Whenever a class migrates, the attribute undergoes the same kind of migration, for instance, when a class ceases to exist, it is disabled, undergoes a dynamic evolution DEV, which triggers a parallel transition for its attributes, ADEV. For example when a student (member of a class Student) graduates and becomes an alumni, his address (represented with an attribute) changes and the scheduled attribute of occupation becomes active. *CASE A* is not just a byproduct of inheritance because the values of the attributes change as object migration takes place.

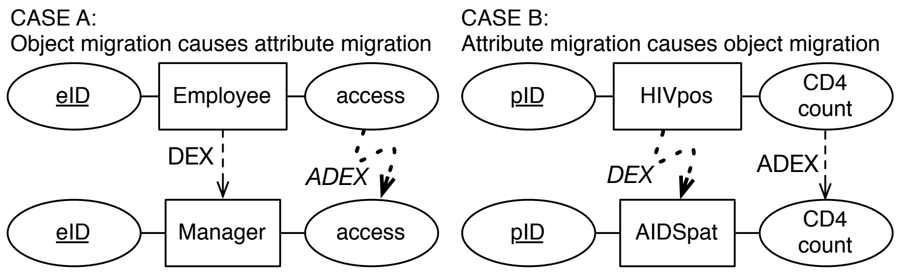

Fig. 3. EER diagram extended with migration constraints showing the interaction between objects and attributes. The dashed arrows illustrate class migration and attribute migration.

CASE B: This type of transition occurs when the value of an attribute changes through an attribute migration and that forces the evolution or extension of an object from the source to the target class. This case can be very useful in medical information systems, when monitoring the value of attributes of a patient: if it falls below a certain threshold, the patient is moved to a different class, which is illustrated in Fig. 3 for HIV positive to transition to AIDS patient upon passing the threshold of CD4 count of 180. It can also be used in administration, to group individuals according to a set criteria, for example when an employee's annual income increases to above a certain value, he is moved to a higher class of tax remittance.

4.3 Logical Implications

Logical implications are important to derive new constraints from a set of defined axioms. Given the set of axioms for attribute migration and the set of axioms for status attributes [15], several logical implications can be derived. We only need a few axioms for status attributes for that, being the straight-forward disjointness and completeness constraints for the four status classes and attributes as indicated in Fig. 1, and using names C denoting the active C, etc., we have:

(ADISAB5) Disabled persists, with a semantics as $a \in Disabled\text{-}A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \notin A^{\mathcal{I}(t')}$ and in DL-notation $\texttt{Disabled-A} \sqsubseteq \Box^{+}\neg\texttt{A}$;

(ADISAB4) Disabled will never be Active, $\texttt{Disabled-A} \sqsubseteq \neg(\texttt{From} : \texttt{Scheduled-C} \sqcup \texttt{From} : \texttt{Suspended-C})$;

(CSUSP3) Freezing attributes of suspended classes, $o \in \mathit{Suspended}\text{-}C^{\mathcal{I}(t)} \rightarrow \langle o, d\rangle \in \mathit{Suspended}\text{-}A^{\mathcal{I}(t)}$, and in DL-notation: $\texttt{Suspended-C} \sqsubseteq \forall[\texttt{From}]\texttt{Suspended-A}$;

(ASCH1) Persists until active, with a semantics $a \in \mathit{Scheduled}\text{-}A^{\mathcal{I}(t)} \rightarrow \exists t'{>}t.a \in A^{\mathcal{I}(t')}$ and in DL notation $\texttt{Scheduled-A} \sqsubseteq \Diamond^{+}\texttt{A}$.

Then we can prove the following.

1. Only temporal attributes can participate in the migration
   We know that only temporal classes are involved in transition, proven in [3], and they have both snapshot and temporal attributes. Temporal attributes hold at single time points only, we prove by contradiction that the attributes cannot be snapshot. Suppose the attribute is snapshot, $\text{ADEX}_{A_1,A_2} \sqsubseteq \oplus\ \text{ADEX}_{A_1,A_2} \sqcap \ominus \text{ADEX}_{A_1,A_2}$, which contradicts because temporal attributes hold only at single time points. Therefore, we can only have temporal attributes.

2. Only active attributes participate in attribute migration
   A temporal active class can have any of the four status attributes, we prove by contradiction that only active attributes can participate in the transition. To prove, we use the status attribute axioms from [15], with A defined as $\Delta_O^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$. Suppose that
   - $a \in \mathit{Disabled}\text{-}A^{\mathcal{I}(t)}$ by (ADISAB5), but disabled is an irreversible state by (ADISAB4). Thus, it can never be modified again, this contradicts, because in a transition the value changes.
   - $a \in \mathit{Suspended}\text{-}A^{\mathcal{I}(t)}$, by (CSUSP3) the attributes are frozen, meaning, no modification can be done on it. This contradicts because the transition warrants the change of the value
   - $a \in \mathit{Scheduled}\text{-}A^{\mathcal{I}(t)}$ By (ASCH1), and remains unchanged until it is active, which contradicts.
   - $a \in A^{\mathcal{I}(t)}$, there is no constraint preventing the change if its value.

5 Discussion

This research provides an avenue for discussion of several issues concerning temporal data in many application areas and how temporal attributes can contribute to temporal reasoning. Temporal reasoning can use temporal constraints to check consistency (no conflicting temporal information) and satisfiability of some constraint, by giving an accurate representation of reality thus eliminate and filter disallowed (impossible) scenarios. Transition constraints are an upcoming research area in OBDA for dynamic objects [7] time points, so that we can query on the past on historical databases and from that look at how to predict the future.

Representing transition constraints for attributes over evolving attributes has not been researched and we present description logic axioms to describe transition behaviour using a minor extended version of the description logic $\mathcal{DLR}_{\mathcal{US}}$ for attributes, as well as some logical implications resulting from the attribute transition. Although it was possible to formalise several transitions, not all attribute transitions can be represented, notably attribute transitions that are dependent on one another, and it was difficult to find relevant use cases for some of them, but not others.

$\mathcal{DLR}_{\mathcal{US}}$ does not have a value comparison operator for attributes, needed in order to formalise *CASE B*. Due to this limitation, we cannot give the full formalisation, but it is important to note *CASE B* as a proposal for future investigation into a suitable description logic language, and to ensure that we cover all the areas that pertain to attributes. It may be possible to elaborate on the datatypes as used in OWL 2 [10] or DL-Lite [4] for more comprehensive logic-based reconstructions of conceptual data models. We can use transition constraints of attributes to control the behaviour and movement of objects as they evolve, thus enabling us to verify and validate our data, using the rules before any evolution occurs, which is a suggested usage also by [19].

$\mathcal{DLR}_{\mathcal{US}}$ is undecidable, which might be seen as a drawback. However, it is useful first to have a language with high expressivity to model what may be needed, rather than complexity and *a priori* restricting oneself, which enables a more comprehensive understanding of temporal attributes and their role in temporal conceptual data models. In the future, we will look at decidable languages together with a prioritisation of language constructs needed for the more relevant types of temporal attributes and their transitions. Based upon that one can choose which of the constraints can be represented and modelled and used to determine the optimal trade-off between a subset of temporal constructs and the complexity costs. This research will also permit us to find out the best way to link up conceptual modelling desires vis-a-vis OBDA [6] to find out what can be implemented in temporal OBDA.

Research is ongoing in security, which considers fine grained access to data, and having temporal databases will be needed to ensure correctness of implementation before there is a build up of possibly dirty data that does not meet the business rules. Temporal attributes can be used to model time changing data in real world applications by enabling the creation of trigger rules to manage the integrity of databases. Hopefully, this paper contributes also to bring awareness on this subject because we need to find an efficient way to represent temporal attributes in databases that will enable users to monitor their evolution.

6 Conclusions

Transition constraints are important to record database evolution, which is useful in several application domains. In this paper we gave a formalisation of evolving temporal attributes and its interaction with evolving objects using $\mathcal{DLR}_{\mathcal{US}}$.

These results have the potential to assist database designers on how to model temporal data. We are currently looking into designing a graphical modelling tool to test these results, and future works pertain to the investigation of the best trade-off between modelling temporal attributes and the complexity of the language.

References

1. Artale, A., Franconi, E., Wolter, F., Zakharyaschev, M.: A temporal description logic for reasoning about conceptual schemas and queries. In: Flesca, S., Greco, S., Leone, N., Ianni, G.

(eds.) Proceedings of the 8th Joint European Conference on Logics in Artificial Intelligence (JELIA'02). LNAI, vol. 2424, pp. 98–110. Springer Verlag (2002)

2. Artale, A., Kontchakov, R., Wolter, F., Zakharyaschev, M.: Temporal description logic for ontology-based data access. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI'13). pp. 711–717 (2013)
3. Artale, A., Parent, C., Spaccapietra, S.: Evolving objects in temporal information systems. Ann. Math. Artif. Intell. 50(1-2), 5–38 (2007)
4. Artale, A., Ryzhikov, V., Kontchakov, R.: Dl-lite with attributes and datatypes. In: Proceedings of the European Conference on Artificial Intelligence (ECAI'12). pp. 61–66 (2012)
5. Artale, A., Kontchakov, R., Ryzhikov, V., Zakharyaschev, M.: Complexity of reasoning over temporal data models. In: Parsons, J., Saeki, M., Shoval, P., Woo, C.C., Wand, Y. (eds.) ER. Lecture Notes in Computer Science, vol. 6412, pp. 174–187. Springer (2010)
6. Baader, F., Borgwardt, S., Lippmann, M.: Temporalizing ontology-based data access. In: Bonacina, M. (ed.) Automated Deduction – CADE-24, Lecture Notes in Computer Science, vol. 7898, pp. 330–344. Springer Berlin Heidelberg (2013)
7. Baader, F.: Ontology-based monitoring of dynamic systems (2014)
8. Booch, G., Rumbaugh, J.E., Jacobson, I.: The unified modeling language user guide - the ultimate tutorial to the UML from the original designers. Addison-Wesley object technology series, Addison-Wesley-Longman (1999)
9. Calvanese, D., De Giacomo, G.: The DL Handbook: Theory, Implementation and Applications, chap. Expressive description logics, pp. 178–218. Cambridge University Press (2003)
10. Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U.: OWL 2: The next step for OWL. Journal of Web Semantics: Science, Services and Agents on the World Wide Web 6(4), 309–322 (2008)
11. Hall, G., Gupta, R.: Modeling transition. In: ICDE. pp. 540–549. IEEE Computer Society (1991)
12. Halpin, T.: Temporal modeling and ORM. In: Meersman, R., Tari, Z., Herrero., P. (eds.) OTM 2008 Workshops. LNCS, vol. 5333, pp. 688–698. Springer (2008)
13. Halpin, T.A.: Object-role modeling: Principles and benefits. IJISMD 1(1), 33–57 (2010)
14. Keet, C.M., Artale, A.: A basic characterization of relation migration. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) OTM Workshops. LNCS, vol. 5231, pp. 484–493. Springer (2010)
15. Keet, C.M., Ongoma, E.A.N.: Temporal attributes: their status and subsumption (2014), (in preparation)
16. Lutz, C., Wolter, F., Zakharyaschev, M.: Temporal description logics: A survey. In: Demri, S., Jensen, C.S. (eds.) TIME. pp. 3–14. IEEE Computer Society (2008)
17. Object Management Group: Superstructure specification. Standard 2.4.1, Object Management Group (2012), http://www.omg.org/spec/UML/2.4.1/
18. Parent, C., Spaccapietra, S., Zimányi, E.: Conceptual modeling for traditional and spatio-temporal applications—the MADS approach. Berlin Heidelberg: Springer Verlag (2006)
19. Theodoulidis, B., Alexakis, P., Loucopoulos, P.: Verification and validation of temporal business rules. In: DAISD. pp. 1–15 (1992)