

# FORMALIZING TEMPORAL ATTRIBUTES IN TEMPORAL CONCEPTUAL DATA MODELS

By  
ESTHER AOKO NASUBO ONGOMA

Supervised By  
PROFESSOR THOMAS MEYER  
and  
DR. C. MARIA KEET

*A thesis submitted in fulfilment of the requirements  
for the degree of Masters in Computer Science  
in the  
School of Mathematics, Statistics and Computer Science,  
University of KwaZulu-Natal - Westville*

2014





## DECLARATION OF AUTHORSHIP

I, E. A. NASUBO ONGOMA, declare that this thesis titled, ‘Formalising Temporal Attributes in Temporal Conceptual Data Models’ and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

---

Date:

---

# LIST OF PUBLICATIONS

## Publication 1

Ongoma, E. A. N., Keet, C. M., and Meyer, T. Transition constraints for temporal attributes. In Informal Proceedings of the 27th International Workshop on Description Logics, Vienna, Austria, July 17-20, 2014. (2014), M. Bienvenu, M. Ortiz, R. Rosati, and M. Simkus, Eds., vol. 1193 of CEUR Workshop Proceedings, CEUR-WS.org, pp. 684-695.

## Publication 2

Keet, C.M., Ongoma, E.A.N. Temporal Attributes: their Status and Subsumption. Asia-Pacific Conference on Conceptual Modelling (APCCM'15). Koehler, H., Saeki, M. (Eds.), Conferences in Research and Practice in Information Technology (CRPIT), Vol. 165. January, 27-30, 2015, Sydney, Australia.

# ABSTRACT

Traditional conceptual data languages (e.g. ER, UML and ORM) are not capable of representing evolving data. Representing data that changes over time in conceptual data models and ontologies is required by various application domains such as medical information systems, financial applications and information security systems. To represent time effectively in data models, a language expressive enough to capture the operational semantics of time-varying information is required. Description logics provide formal semantics that characterise temporal conceptual modelling constructs by specifying the structure of conceptual data models. This turn permits reasoning on implicit knowledge. Previous research focused on representing and reasoning on temporal classes and relationships, but had scant support for representing and reasoning over temporal attributes. Partial formalisation of temporal attributes prevents the full utilisation of temporal conceptual data models and tracking the interaction between temporal attributes and temporal classes. Improper representation of temporal attributes in conceptual models leads to inconsistency in databases with respect to the constraints they hold. This research is centred on logic-based representation of temporal data at the conceptual level which provides a link between temporal conceptual data models and a temporal description logic. Our aim is to properly formalise temporal attributes to create a fully temporised conceptual model. We extend the very expressive temporal description logic language  $\mathcal{DLR}_{US}$  with a precise syntax and semantics for temporal attributes, then use it to formalise temporal attributes in a temporal conceptual data model. An expressive logic is chosen because it allows us to express many temporal constructs for classes, relations and now attributes. The results from this research can be summarised as follows. Firstly, the result introduces the notion of status attributes which captures the evolution of a temporal attribute as it moves along a temporal object. Secondly, the results provide a mechanism for attribute transition concerning the change of attribute properties over time as the attribute migrates from one temporal class to the next. Thirdly, the effect of class inheritance on a temporal attributes have been show clearly. We provide the formalisation of status and transition for temporal attributes as well as their logical implications which permit the correct behaviour in subsumption and transition.  $EER_{VT}^{++}$  is very expressive and allows full temporalisation of attributes but at the same time it is undecidable. Future work will look at a more decidable language to permit the creation of a tool with these results.

# ACKNOWLEDGEMENTS

This dissertation would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my utmost gratitude to my supervisor Dr. Maria Keet, for introducing me to the wonders and frustrations of research. Her selfless and dependable support, guidance and advice made all this bearable.

I thank Professor Thomas Meyer for the seamless transfer as my new supervisor and for his advice on the DL workshop paper.

My sincere gratitude to the University of Cape Town for allowing me to visit their university to enhance my knowledge and for the warm lab.

The SA-AR project for allowing me to travel for the LPCM conference in Stellenbosch. I am deeply grateful to the Centre for Artificial Intelligence Research, jointly funded by the University of KwaZulu-Natal and the CSIR Meraka Institute for the scholarship that enabled me to travel to Vienna for the DL Workshop and for the APCCM conference in Sydney, Australia.

To my friends, at the ICT4D lab for their constant cheer and encouragement, those in Durban at the CAIR lab and from the Chemistry department. This thesis would not be complete without your constant cheer.

Last but not least, my gratitude to my family for your prayers and unfailing love and to God, who makes all things possible.

# TABLE OF CONTENTS

DECLARATION OF AUTHORSHIP	i
LIST OF PUBLICATIONS	ii
ABSTRACT	iii
ACKNOWLEDGEMENTS	iv
CONTENTS	v
LIST OF FIGURES	vii
LIST OF TABLES	ix
ABBREVIATIONS	xi
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 General Introduction . . . . .	1
1.2 Motivation . . . . .	1
1.3 Conceptual Data Models . . . . .	2
1.4 Temporal Conceptual Data Models . . . . .	2
1.5 Logic for Conceptual Data Models . . . . .	3
1.6 Research Questions . . . . .	6
1.7 Organisation of dissertation . . . . .	6
<b>2 RELATED WORK</b>	<b>9</b>
2.1 Time Concepts . . . . .	9
2.1.1 Models of time . . . . .	9
2.1.2 Time Dimension . . . . .	10
2.1.3 Notions of time . . . . .	10
2.2 Temporal Research Areas . . . . .	11
2.3 Modified Criteria for Modelling Temporal Models . . . . .	12
2.4 Study of Temporal Conceptual Models . . . . .	14
2.4.1 The Temporal Conceptual Model $\mathcal{ER}_{\mathcal{VT}}$ . . . . .	16
2.5 $\mathcal{DLR}_{US}$ . . . . .	19
2.6 Summary . . . . .	21

<b>3</b>	<b>TEMPORAL ATTRIBUTES</b>	<b>23</b>
3.1	$\mathcal{DLR}_{US}$ with Attributes	23
3.2	Temporal constraints	25
3.2.1	Timestamping	25
3.2.2	Evolution constraints	26
3.2.2.1	Lifecycle	26
3.3	Inheritance and Temporal Attributes	32
3.4	Transition Constraints	37
3.4.1	Transition Constraints for Classes	39
3.4.2	Transition Constraints for Attributes	41
3.5	Interaction between transition classes and attributes	44
3.6	Summary	46
<b>4</b>	<b><math>EER_{VT}^{++}</math> AND <math>\mathcal{DLR}_{us}</math></b>	<b>47</b>
4.1	Definition of $EER_{VT}^{++}$	47
4.1.1	$EER_{VT}^{++}$ Conceptual data model	47
4.1.2	$EER_{VT}^{++}$ Semantics	49
4.1.3	Mapping $EER_{VT}^{++}$ into the extended $\mathcal{DLR}_{us}$	49
4.1.4	Graphical Representation of $EER_{VT}^{++}$	50
4.2	Differences between $\mathcal{ER}_{VT}$ & $EER_{VT}^{++}$	53
4.3	Summary	55
<b>5</b>	<b>APPLICATIONS OF TEMPORAL ATTRIBUTES</b>	<b>57</b>
5.1	Application areas	57
5.1.1	Administration	58
5.1.2	Information Security Systems	60
5.1.3	Medical Information Systems	61
5.1.4	Financial Institutions	62
5.2	Summary	63
<b>6</b>	<b>DISCUSSION</b>	<b>65</b>
6.1	Answering research questions	65
6.2	Discussion of the results	66
6.3	Transferability to other modelling languages	68
6.4	Challenges	69
<b>7</b>	<b>CONCLUSION</b>	<b>71</b>
7.1	Dissertation Achievement	71
7.2	Significance of the Results	72
7.3	Future work	72
<b>A</b>	<b><math>\mathcal{ER}_{VT}</math> CLASSES AND RELATIONSHIPS</b>	<b>75</b>
A.1	Those involving Classes	75
A.2	Those involving Relations	77
	<b>REFERENCES</b>	<b>79</b>



# LIST OF FIGURES

2.1	Syntax and semantics of $\mathcal{DLR}_{US}$ .	19
3.1	Syntax and semantics of $\mathcal{DLR}_{US}$ , modified to include attributes	24
3.2	EER diagram with the attribute hierarchy of status attributes.	27
3.3	EER diagram with class and attribute hierarchy	28
3.4	Example ER diagram showing attribute migration	41
3.5	$EER_{VT}^{++}$ diagram extended with migration constraints	44
4.1	$EER_{VT}^{++}$ Graphical syntax.	51
4.2	$EER_{VT}^{++}$ diagram showing an application example	52
4.3	$EER_{VT}^{++}$ verses $\mathcal{ER}_{VT}$ graphical representation	54
5.1	$\mathcal{ER}_{VT}$ model	59
5.2	$\mathcal{ER}_{VT}$ diagram showing quantitative migration	60
5.3	$EER_{VT}^{++}$ diagram extended with migration constraints	61
5.4	$EER_{VT}^{++}$ diagram with Financial institution example	63
6.1	Example of evolution due to change in data	68
6.2	ORM diagram showing the transition constraints	68
6.3	UML diagram showing the transition constraints	69



# LIST OF TABLES

2.1	Modified criteria for temporal conceptual models . . . . .	16
3.1	Illustrations of logical implication of status attributes . . . . .	37
6.1	Anaysis of $EER_{VT}^{++}$ against $\mathcal{ER}_{VT}$ , TIMERplus and MADS . . . . .	67



# ABBREVIATIONS

<b>ABAC</b>	<b>Attribute Based Access Control</b>
<b>ADEX</b>	<b>Attribute Dynamic Extension</b>
<b>ADEV</b>	<b>Attribute Dynamic Evolution</b>
<b>AIDS</b>	<b>Acquired Immune Deficiency Syndrom</b>
<b>APEX</b>	<b>Attribute Persistent Extension</b>
<b>APEV</b>	<b>Attribute Persistent Evolution</b>
<b>AQEX</b>	<b>Attribute Quantitative Extension</b>
<b>AQEV</b>	<b>Attribute Quantitative Evolution</b>
<b>DEX</b>	<b>Dynamic Extension</b>
<b>DEV</b>	<b>Dynamic Evolution</b>
<b>DL</b>	<b>Description Logics</b>
<b><math>\mathcal{DLR}_{US}</math></b>	<b>DLR with Until and Since operators</b>
<b>ER</b>	<b>Entity Relationship</b>
<b>EER</b>	<b>Extended Entity Relationship</b>
<b><math>\mathcal{ER}_{VT}</math></b>	<b>EER with Valid Time (with Temporal Classes)</b>
<b><math>EER_{VT}^{++}</math></b>	<b>EER with Valid Time (with Temporal Classes, Temporal Relationships and Temporal Attributes)</b>
<b>HIV</b>	<b>Human Immunodeficiency Virus</b>
<b>HR</b>	<b>Human Resource</b>
<b>OBDA</b>	<b>Ontology-Based Data Access</b>
<b>ORM</b>	<b>Object Role Model</b>
<b>PEX</b>	<b>Persistent Extension</b>
<b>PEV</b>	<b>Persistent Evolution</b>
<b>QEX</b>	<b>Quantitative Extension</b>
<b>QEV</b>	<b>Quantitative Evolution</b>

---

<b>RDEX</b>	<b>Dynamic Extension of a Relation</b>
<b>RDEV</b>	<b>Dynamic Evolution of a Relation</b>
<b>RPEX</b>	<b>Persistent Extension of a Relation</b>
<b>RPEV</b>	<b>Persistent Evolution of a Relation</b>
<b>RQEX</b>	<b>Quantitative Extension of a Relation</b>
<b>RQEV</b>	<b>Quantitative Evolution of a Relation</b>
<b>SADEV</b>	<b>Strong Attribute Dynamic Evolution</b>
<b>SDEV</b>	<b>Strong Dynamic Evolution</b>
<b>SPEV</b>	<b>Strong Persistent Evolution</b>
<b>SQEV</b>	<b>Strong Quantitative Evolution</b>
<b>SRDEX</b>	<b>Strong Relationship Dynamic Evxtension</b>
<b>SRDEV</b>	<b>Strong Relationship Dynamic Evolution</b>
<b>SRPEV</b>	<b>Strong Relationship Persistent Evolution</b>
<b>SRQEV</b>	<b>Strong Relationship Quantitative Evolution</b>
<b>SQL</b>	<b>Structured Query Language</b>
<b>tSQL</b>	<b>temporal Structured Query Language</b>
<b>UML</b>	<b>Unified Modelling Language</b>

*Dedicated to my family . . .*





# 1

# INTRODUCTION

## 1.1 General Introduction

Databases and knowledge representation systems represent some aspect of the real world known as “mini world”; they store and maintain information about the world. With the growth of databases and the dynamic nature of data, there is need for recording and keeping track of past, current and possible future changes of data. As current systems can only record data, the big question becomes, how do we expand the capabilities of these systems to keep track of the evolution of the recorded data? As an illustration, consider a student who after his graduation becomes a postgraduate or an employee who later on becomes a manager after a few years. The evolution of the student from an undergraduate to a postgraduate, or an employee to a manager should be shown. The challenge is enabling this kind of evolution and formulating constraints which govern the evolution. One approach to solving this problem is to have a conceptual data model which stores the different states of evolution and the required parameters for proper data storage. The approach to this research is to extend an existing temporal conceptual model and link it with a temporal description logic language to permit reasoning over temporal models. To this end, we first give the motivation for this research, then introduce non temporal and temporal conceptual data models in [Section 1.3](#) and [Section 1.4](#) respectively. Thereafter introduce logics for conceptual data languages and the research questions and finally give the structure of the dissertation in [Section 1.7](#).

## 1.2 Motivation

We need a temporal conceptual data model that ensures that all user requirements are met and that can check the satisfiability of the model. This is facilitated by having a formalised temporal model, with formalised classes, relations and attributes. Previous research addressed formalisation of temporal classes and relations but had little formalisation of temporal attributes. Lack of a proper formalisation of temporal attributes will

lead to database integration problems that will build up in temporal models if not addressed. A fully formalised temporal model on the other hand ensures correct behaviour of a program and allows only permitted actions.

### 1.3 Conceptual Data Models

Conceptual data models are the first step towards creating a database, it is the point at which data requirements are captured by the database designer. They specify user requirements with detailed descriptions of entities, relationships, their properties and constraints. User requirements are done at the conceptual level to ensure correctness, clarity and cost-effectiveness of quality database schemas [64]. Conceptual data models have many advantages that make it an ideal tool for data modellers. Firstly, it uses graphical syntax which makes it easier for non-technical users to understand and interpret concepts, relations and their constraints. Secondly, they allow users to concentrate more on data specification and less about storage and implementation details, thus ensuring that all the requirements are captured and reflected in the database before implementation. Thirdly, they are used for future reference when one needs to update the database. Examples of conceptual data modelling languages are Entity Relationship model (ER) [25], Object Role Modelling (ORM) [42] and Unified Modelling Language (UML) [57].

The ER model was developed by Chen [25] which arose from the need of organisations to have a unified methodology for file structure and database design [26]. Despite its success, it was incapable of representing hierarchical relationships i.e., the superclass and the subclass relationships [56], that define generalisation and specialisation [54]. This gap was addressed by adding extensions to the ER model, now known as the Extended or Enhanced Entity Relationship (EER) model. There are many notational variants of the enhanced ER model in practice (see [63] for a list of older models), the most prominently used EER is by Elmasri and Navathe [32].

### 1.4 Temporal Conceptual Data Models

Time plays a major role in today's databases in modelling histories, planning for future and tracking entities across time as they migrate between roles [40]. Traditional databases store data that is true at that current time; any alteration in the database will result into addition of new values and deletion of the previous values. Therefore, they do not maintain the history of the database which shows how data has evolved. Adequately addressing the issue of time in databases is challenging because databases use relational databases which have a flat structure, are two-dimensional in nature and lack time support [28]. Traditional databases have two shortcomings regarding temporal data. First,

they have poor support for storing complex temporal information e.g., merging temporal overlapping data. Second, relational database use SQL which has limited support for expressing temporal queries. Very few tools exist which support temporal conceptual data modelling. This means that temporal modellers have to define their own temporal models as well as their own query systems [66]. Temporal databases differ from traditional databases by having the ability to define valid and/or transactional time and have a temporal primary key that has non-overlapping period constraints. In the past 20 years, the research community has proposed several ways to incorporate time into relational databases. Snodgrass and Ahn [62] proposed adding time as a third dimension either as valid time or transaction time. Snodgrass [61] proposed the use of a temporal SQL (tSQL) although it was never fully implemented. Steiner [65] proposed adding a software layer which supports temporal data model and its temporal query language on top of the RDBMS. Availability of algorithms that can automatically map conceptual models to relational database e.g. [38] appeals to most database developers because now, developers can now concentrate more on temporal conceptual data models and less on its mapping.

Temporal research encompasses several areas, temporal reasoning by using explicit knowledge to get implicit consequences and temporal data maintenance, which is how to store temporal data for future use and to preserve history so that databases are queried on past data and predict the future from the past. Temporal research has been published in diverse fields, in logic-based representation and reasoning, temporal conceptual data models, informal business rules about time among others. Our research focuses on the linking of a temporal description logic to conceptual data models to permit reasoning over temporal models.

## 1.5 Logic for Conceptual Data Models

Description Logics (DL) are used as a formal language for representing and reasoning over knowledge, which provides a way to model relationships and entities in a domain of interest [16]. They provide formal semantics that characterise temporal conceptual modelling constructs by specifying the structure of conceptual data models which in turn permits reasoning on implicit knowledge [22]. Over the years, DL is being used more and more because of its high expressivity and decidability which enable it to deduce correct answers in tractable time [1, 16]. Having temporal conceptual models is just a partial solution to the real problem, we also need to reason over the models before we can comfortably create temporal databases, which is the first theoretical step towards temporality. Examples of conceptual data models that have been formalised using description logics include the EER model using  $\mathcal{DLR}_{ifd}$  in [50] to formalise classes

and relationships, UML in [18], first using  $\mathcal{DLR}_{ifd}$ , then using the description logic language  $\mathcal{ALCQI}$ .  $\mathcal{ER}_{VT}$  has also been formalised using the description logic languages  $\mathcal{DLR}_{US}$  in [2, 8] and  $\mathcal{ALCQI}_{US}$  in [6]. ORM formalisation using the description logic language  $\mathcal{DLR}_{ifd}$  was presented in [46, 49], whereas the formalisation of ORM2 using  $\mathcal{ALCQI}$  DL with its [30, 35]

Using DL's to formalise conceptual data models is advantageous in that complete logical reasoning can be achieved in conceptual model by an underlying DL inference engine [3]. The formalisation is based on model-theoretic semantics, which clarify the meaning of language constructors and give definition to the relevant modelling notions i.e., schema satisfiability, logical implication and subsumption. DL are used in relational databases to automatically deduce implicit knowledge from the explicitly represented knowledge and yield a correct answer in finite time [17]. This is done through expressing the conceptual domain of the data source, integrating multiple data sources, expressing, and evaluating queries. We can informally say it is done to in a way to imitate the human reasoning or conceptualisation.

The correspondence between conceptual data models and description logics is seen by the implementation of a CASE tool (iCOM) developed by Fillottrani et. al. [34]. The iCOM tool has a graphical interface using a chimera of UML and EER diagrams and uses a description logic reasoner that acts as an inference engine to verify correctness and manifest inconsistencies on the model. This enables it to have the capability to achieve complete logical reasoning by automatically inferring implicit facts, deriving stricter constraints and detecting any inconsistencies in the conceptual model. We also see the mapping of ORM in the DogmaModeler tool using the DL  $\mathcal{SHOIN}$  to achieve reasoning on ORM [45].

Description logics have been extended with temporal logic for representing and reasoning over temporal conceptual models. They allow for logical reconstruction and extension of conceptual data models for dynamic and evolving information e.g.,  $TDL-Lite$  [9],  $\mathcal{ALCQIT}$  [2],  $\mathcal{ALCQI}_{US}$  [4]. The Temporal description logic proposed in [6] is  $\mathcal{DLR}_{US}$ , a combination of the expressive and decidable  $\mathcal{DLR}$  [23] with the linear temporal logic with temporal operators *Since* and *Until*.  $\mathcal{DLR}_{US}$  captures the temporal and atemporal constructs of a temporal ER schema, with both timestamping and evolution constraints.  $\mathcal{ER}_{VT}$  [2], uses the temporal description logic language  $\mathcal{DLR}_{US}$  [5] which covers mainly the temporal behaviour of classes [13] and relationships [7]. Most of the previous work had little or no formalisation of temporal attributes, yet, for a temporal conceptual data model to be fully utilised in information system development that deals with evolving data, a proper treatment of temporal attributes is also necessary. Temporal attributes are used in many application areas and it is difficult to find an application area that does not require temporal data. We look at the following application areas, administration, financial applications, medical information systems and information security

systems, to monitor, verify and ensure that processes run as planned. Such information has to be captured also at the conceptual model layer during design instead of being ignored or only encoded in an implementation. Most times, users of temporal conceptual data models ignore adding temporal attributes at the conceptual stage only to add them at implementation. As a result, we do not have a temporal conceptual data model that can fully capture temporal information and reason over it for correctness. However, we know that attributes are important because they associate values to objects and give the characteristics of the entities and relationships.

An ontology is a formal specification of a subject domain, which is similar to conceptual modelling, conceptual models have a specific application while ontologies are application independent. It can be argued that conceptual data models formalised with description logics can be regarded as ‘application ontology’. On going temporal research on ontology based data access (OBDA) [12] has omitted temporal attributes, despite that it is essential in OBDA, as the “ontology” in OBDA is quite like a formalised conceptual data model. OBDA adds a semantic layer to multiple sources to query over the databases by rewriting the queries and ontologies into the vocabulary of data sources. A recent comprehensive treatment of attributes in DL was only for the atemporal case [14]. To the best of our knowledge, no logic-based characterisation, including evolution constraints and its implications, exists for temporal attributes, which prevents its potential usability for temporal information system development. A “temporary attribute” was formalised in [13] but it had no effect on modelling in  $\mathcal{ER}_{\mathcal{VT}}$ , and on transition constraints and their interaction with temporal classes. To capture temporal information in temporal conceptual data models, the conceptual modelling language must have the expressiveness to represent evolving data, and ideally, a modelling tool. In order to address this gap in understanding and representing temporal attributes, we approach it from a modelling expressiveness viewpoint, instead of a priori, limiting ourselves to a computationally well-behaved logic of low expressiveness.  $\mathcal{DLR}_{\mathcal{US}}$  is chosen as a base logic, with a minor extension to its syntax and semantics to represent attributes and extending the conceptual data modelling language  $\mathcal{ER}_{\mathcal{VT}}$  (with its reconstruction in  $\mathcal{DLR}_{\mathcal{US}}$ ), that we obtain by putting all the piecemeal refinements together, now known as  $EER_{\mathcal{VT}}^{++}$ . This very expressive temporal conceptual modelling language includes not only the temporalisation of relationships (a by-product from [7]) but also the new formalisation for temporal attributes that has not only the assertion of temporary attribute and freezing, but, more importantly, a full set of constraints including status attributes, temporal attribute inheritance (subsumption), attribute transition constraints and their logical implications. While  $\mathcal{DLR}_{\mathcal{US}}$  is undecidable, it is expressive enough to obtain insight into the language features required for a full temporalisation and a comprehensive understanding of temporal attributes. The solution proposed will bring strictness on

information representation, hence detects inconsistencies and consequently have better databases.

## 1.6 Research Questions

The aim of this study is to create a better quality data model through the formalisation of temporal attributes in temporal models, that can automatically detect inconsistencies and derive new constraints. This research is guided by the following research questions:

1. Do conceptual models have temporal attributes? If so, how are they represented?
2. Is it possible to have a fully temporal model with respect to the current EER model?  
If so,
  - (a) how are temporal attributes added graphically on the temporal model?
  - (b) does the formalisation of temporal attributes add to reasoning properties in a temporal model?
3. Do temporal attributes affect temporal classes and vice versa? If so, how?

## 1.7 Organisation of dissertation

This dissertation is organised as follows

- Chapter 1 outlines the introduction on time in conceptual models, as well as the terminology used throughout the dissertation.
- Chapter 2 gives the background to temporal modelling research as well as the syntax and semantics of the description logic language  $\mathcal{DLR}_{US}$ .
- Chapter 3 presents the main work of this dissertation and provides a rigorous formalization of temporal conceptual data model using the extended DL language  $\mathcal{DLR}_{US}$ , status, transition constraints, as well as the interaction between temporal classes and temporal attributes.
- Chapter 4 presents the textual syntax of  $EER_{VT}^{++}$  as well as its mapping to  $\mathcal{DLR}_{US}$  and finally outlines the difference between  $\mathcal{ER}_{VT}$  and  $EER_{VT}^{++}$ .
- Chapter 5 gives different application areas of the temporal model  $EER_{VT}^{++}$ , by entailing why we need temporal attributes.
- Chapter 6 gives a summary of the work, as well as the challenges met.
- Chapter 7 concludes the work done in this study and provides direction for future work

- Appendix [A](#) has the formalisation of status classes and status relations as well as transition.





# 2

## RELATED WORK

This literature review gives a brief overview of temporal conceptual data models with the key concepts in temporal research. Key concepts of time used in various temporal data models are introduced, with an overview of existing temporal models, as well as the introduction of the description logic language  $\mathcal{DLR}_{US}$ .

### 2.1 Time Concepts

Time is used in many aspects of human life - to monitor and record changes and duration of events and to show the timeline of activities. To model time correctly, there is a need to specify the beginning of time in our model to dispel any ambiguity in the data. Since there is no known beginning or end of time, a reference point is needed to measure time points. This standardisation is done with the use of calendar systems, (e.g. Jewish, Chinese, Islamic) [61]. Although this system is not very accurate to measure the ‘true day’, it is the most acceptable (accurate) way of recording time. The de facto international calendar used to record extensive periods of time in the database is the Gregorian calendar. To count the ongoing passage of time, we use the clock, which orders periods that are less than one day [61]. Time in databases is an ordered sequence of time points on a linear time axis, with the size of a data item termed as the temporal granularity. The smallest time instant is known as the chronon, and is defined as the smallest, discrete, non-decomposable unit of time in a temporal data model [31]. Consecutive chronons may be grouped into larger intervals or segments, termed as granules. Temporal granularities are a sequence of time granules consisting of time instants, usually with a formal definition, and specifies the temporal qualification of a set of data, which includes, year, month, week, day, hour, minute and second [19]. Semantics of granularities need to be captured and represented to achieve temporal reasoning.

#### 2.1.1 Models of time

A temporal database stores time for which facts are true. This time is stored as either an instant or an interval. An instant is the time a fact occurred in real life and the fact has the same start and end time [47]. For example when a person is employed, that event

occurred once and is recorded once. An interval is a time slice (period), specifying the period within which the fact occurred. It is the consecutive chronons between two time instants, with the start time point and end time point, being different [69]. Interval can be looked at in two ways: Continuous time point, for example when a person starts working for a company for 3 years. Non continuous time point, as known as temporal element termed as a finite union of intervals, which may have several start times and end times [47]. For example when a person starts working for a company on a one year contract, later stops after the contract expires signs a new contract and resumes work. Non continuous time are closed within set theoretic operations of union and complement, continuous time points are not. An interval that includes Now (current time) expands as time advances.

### 2.1.2 Time Dimension

There are several time dimensions used by data modellers. These include valid time, transaction time, bi-temporal time and user defined time [62]. The database developer can determine which time dimension to use, depending on the database constraints set by a company.

1. Valid time (VT) is the actual time when the fact is said to occur in the world. This can be updated at any time in the database.
2. Transaction time (TT) is the time when the fact is recorded in the database, it can be termed as a history of data changes, it cannot be time later than current time and does not allow for updates on the data.
3. Bi-temporal data (BT) is a combination of both valid and transaction time, time is independent with respect to one another.
4. User defined time (UDT) is additional information that has data time values, needed in a database whose temporal semantics are not either valid time or transaction time. The user interprets the time, for example birthday.

### 2.1.3 Notions of time

Temporal constructs can be added on the ER model in two ways:

1. Implicit: The temporal ER model is made temporal, meaning that the temporal dimension is hidden in the interpretation structure and temporal information is implied throughout the diagram, when new semantics are added, the old ones are just declared temporal.
2. Explicit: The temporal ER model retains the original EER semantics and introduces new temporal constructs to represent temporal entities and relationships. This

change may vary from a minor to a total change and will cause the designers to learn about the additional semantics.

## 2.2 Temporal Research Areas

Ongoing temporal research in conceptual modelling include:

1. Extension of the conceptual models e.g. UML, ER or ORM to represent temporal data.
  - (a) These extensions may be hidden or informal. For instance, UML has a “freeze” attribute [57] which states that the value of the attribute will not change over time in the class. This cannot be fully utilised for temporal conceptual data models because it represents only one aspect of temporality.
  - (b) Informal business rules about time in ORM2 [40], uses constraints and derivation rules to model time in ORM. The business rules may either be static or dynamic rules to model histories and track changes in the entities as they migrate from one role to another.
  - (c) Older temporal models extend existing conceptual models with time constructs [13, 36, 37, 60, 69]. For a survey of older temporal models, see, Gregersen and Jensen [37]. UML has been extended to include temporal information in [21] by introducing new terms that differentiate between temporal and non temporal. These include constant, permanent to symbolise snapshot, while instant and interval specify temporal time.
  - (d) Adding temporal constraints to a non-temporal model. McBrien in [55] uses a non-temporal UML to present temporal constraints to model objects using linear temporal logic *Until* and *Since*.
  - (e) Annotation-based ad hoc formal constraints without a specification of the model [52]. Khatri et. al [52] used telic (goal related semantics) and atelic (devoid of culmination), to differentiate semantics by looking at the ‘what’, seeing what is important in the data model and ‘when’ to capture temporal semantics.
  - (f) Having an object-oriented specification language that describe information systems. TROLL [48] uses declarative specification of conceptual models to describe the behaviour of an object using temporal logic for dynamic constraints on attribute evolution and first order logic on object assertions.
2. Spatio-temporal conceptual data models (e.g. MADS [59]) used for modelling spatial temporal data. MADS is an extension of ER model with support for data

types and operators for spatial temporal data. GeoFrame-T in [29] uses temporal UML to model spatial temporal data.

3. Using temporal logic on temporal conceptual models to formalise and reason over temporal models. Logic-based representation and reasoning at the concept-level uses description logics to define classes and relations of a conceptual schema [6, 54] as well as check complexity results and algorithms for satisfiability. This formalisation is used to define clearly snapshot and temporal concepts in a conceptual model.
4. Representing temporal ontology based data access (t-OBDA) in the presence of data [12, 15]. OBDA has formalised classes and relations to give a unified view of data. Adding temporality to OBDA allows one to track changes over time.

Our research narrows down to logic-based representation and reasoning of the temporal ER model with temporal description logics. The advantage of using a logic-based temporal representation is that reasoning can be achieved, therefore we can derive hidden constraints not specified in the model and that temporal enables us to have a consistent ER model. The other reason is that both implicit and explicit approaches to modelling temporal ER can be addressed [2], therefore we eliminate the problem of specifying which model one can support. Before we formalise the temporal model, we carried a survey of three existing temporal ER models, these are, MADS [58, 60], TIMEERplus [36] and  $\mathcal{ER}_{VT}$  [13], using the now modified criteria for modelling temporal models, introduced in the next section.

## 2.3 Modified Criteria for Modelling Temporal Models

The criteria for modelling temporal conceptual data models by Gregersen and Jensen [37] was modified for logic-based temporal conceptual modelling. This criteria is not limited to our work, but it can also be used as a reference or guide for designing or choosing temporal conceptual data models in general.

1. Time dimensions with built-in support  
Time support given by temporal model, in terms of how temporal aspects are represented in the models, for example, valid time, transaction time, and user defined time, all or combinations of the above.
2. Implicit verses Explicit Constructs  
Shows how temporal support is achieved using temporal constructs on the model either, implicitly, by hiding timestamps in temporal semantics or explicitly by adding temporal constructs to already existing ER constructs, thereby changing the aesthetics of the diagram.

### 3. Mandatory and Optional use of Temporal constructs

The extent of changes made on the temporal ER model, which determine if additional temporal constructs are mandatory or optional. If all the original temporal constructs are simply made temporal, without adding more constructs, it means that it is mandatory. Mandatory means that the user can no longer use the older constructs in the diagram. Optional use provides the designer with the choice of mixing temporal and non-temporal constructs.

### 4. Time Data types supported

Different time data types may be used for capturing temporal aspects of database objects, to indicate either valid or transaction time. The time data types include instant, interval and temporal elements.

### 5. Support for Granularities

Data to be modelled from the mini world may be captured using different granularities. It is important to allow variability of temporal data and not restrict it to just one granularity.

### 6. Timestamping

It is a temporal marking mechanism that positions data relevance on a timescale, depending on the criterion used, e.g. Valid Time (VT) or Transaction Time (TT). It is optional for the users and enriches the static view of data, over a period of time [13] to discriminate between objects that change with time and those that are time-invariant.

### 7. Evolution constraints

These are rules that govern transitions in order to model temporal behaviour of an object. They control the mechanism that rules dynamic aspects, permissible transitions from one state to the next one [13].

#### (a) Status

They model lifecycle of an object, attribute or relation and are also known as status constraints [3]. They rule permissible evolution of an instance a membership in a class/relationship along its lifespan [13]. Lifecycle of an object is grouped into scheduled, active, suspended and disabled.

#### (b) Transition

These are constraints that restrict movement of data from one state to another, they are also known as transition constraints. It is a way of enforcing data to ensure that it does not enter an impossible state because of a previous state; i.e. they rule evolution of object from being a member of one class to another member of another class (from source to target class) [13].

## 8. Upward Compatibility

The capability of preserving the non-temporal semantics of conventional conceptual schemas when embedded into temporal schemas, with respect to the ER model it extended. Upward compatibility checks if the original ER diagram will be the same, when temporal semantics are removed.

## 9. Snapshot Reducibility

To check if the snapshots of the temporal model are the same as the conventional conceptual model. This applies to the various constraints that may be defined on attribute and relationship types.

## 10. Design Model or Implementation Model

Either an algorithm that maps directly to a relational model or a query language, map to the temporal ER then to a conventional ER diagram.

## 11. Graphical notation provided

To most people, it is easier to learn and define with graphical notations. We can look at the mapping cardinalities in the over the lifespans of the entities.

## 12. Graphical Editor

This would enable potential users to try it hence increase its usability.

## 2.4 Study of Temporal Conceptual Models

### I. MADS model [58, 60]

MADS is a spatial temporal ER model developed from the ERC+ [69] model, using valid time. The model uses two types of granularity, the spatial granularity, i.e. scales and temporal granularity, (second, minute and hour), with functions to allow conversion from one granularity to another. MADS allows for timestamping of attributes and objects types by allowing lifecycle, i.e. objects can be able to move through several states, as data evolves, these states include not-yet-existing, active, suspended and disabled. For complex data structures, its attributes are decomposed into other attributes. Relationship types allow for timestamping by generalisation, timing, transition and time based aggregation. The MADS model also provides a logic based formalisation of MADS semantics. Temporal constructs are added explicitly to be able to cater for spatial and temporal objects, relationships and attributes. Nothing was added for attributes so far, we have the timestamping and the lifecycle for attributes. It is used for land management and utility networks and it has been tested in various projects in oil management in Columbia.

### II. $\mathcal{ER}_{\mathcal{VT}}$ [13]

$\mathcal{ER}_{\mathcal{VT}}$  is a temporal extension of the EER model [32], using valid time. The

model's temporal constructs are added explicitly on the model, for instance time-stamping by having a marking T for Temporal and S for Snapshot. This marking helps in both separating temporal and non temporal constructs and also preserves upward compatibility. The model also has evolution constructs that allow and permit evolution of classes and relationships along its lifecycle, which records the state of membership of an object/ relationship.  $\mathcal{ER}_{VT}$  uses the description logic language  $\mathcal{DLR}_{US}$  to formalise both classes and relationships. The model has a textual syntax with model theoretic semantics for temporal extension of the ER semantics.

### III. TIMEERplus [36]

The TimeERplus model was developed from the EER model by [32] and uses both valid time and transaction time. The model uses implicit temporal support, with the option to use the temporal constructs or not, thus ensuring upward compatibility with the previous model. The model has support for multiple granularities, including hour, day, week, month and year. To differentiate between temporal and atemporal data, the notation for specification of time sequences attributes, observation and update pattern relationship is provided. These values are LS (Lifespan), TT (Transaction Time), BT (Bi-Temporal Time), LT (combination of Lifespan and Transaction Time) and VT (Valid Time). Lifespan participation constraint is added to the entity to restrict the participation of the entity over time. No logic based semantics or model theoretic semantics have been provided for this model.

Table 2.1 uses the modified design criteria for temporal ER models against three temporal models, MADS, TIMEERplus and  $\mathcal{ER}_{VT}$ . The possible outcomes are Yes, to symbolise that the model has the said characteristic, while No means that it does not meet the criteria. Two of the temporal conceptual models used in this literature use explicit method with valid time; with the exception of the TIMEERplus, which uses the implicit method, with both valid and transaction time. The TIMEERplus model and the  $\mathcal{ER}_{VT}$  model have extended the EER model [32], while MADS uses the ERC+ model. Both MADS and  $\mathcal{ER}_{VT}$  give the model theoretic semantics that show a formal clarification of temporal constructs and to reason over them. The major difference in the models discussed is that the MADS model is for the spatio-temporal data, while the other conceptual data models are for temporal data. The MADS model has a data manipulation language while the TIMEERplus and  $\mathcal{ER}_{VT}$ ; have no known associated program that can manipulate data. Despite all these benefits, there is no compete ER model, that can reason over temporal classes, relationships and attributes and their interrelationships. We narrowed down to the  $\mathcal{ER}_{VT}$  model due to the fact that it met most of the requirements. In  $\mathcal{ER}_{VT}$  provided model theoretic semantics which make it possible to map it with the description logics language  $\mathcal{DLR}_{US}$ .

TABLE 2.1: Modified criteria for modeling temporal conceptual models, overview of  $\mathcal{ER}_{\mathcal{VT}}$ , TIMEERplus and MADS

Modified criteria for modeling temporal models		$\mathcal{ER}_{\mathcal{VT}}$	TIMEERplus	MADS
1. Time dimension support		VT & UDT	BT & UDT	VT & UDT
2. Implicit verses Explicit Constructs		Explicit	Implicit	Explicit
3. Mandatory and Optional		Optional	Optional	Optional
4. Time Datatypes support		Instant, Inter- val & TE	Instant & TE	Instant, Inter- val & TE
5. Support for Granularities		No	Yes	No
6. Timestamping		Yes	Yes	Yes
7. Status	Classes	Yes	No	Yes
	Relations	Yes <sup>1</sup>	No	Yes
	Attributes	No	No	No
8. Transition	Classes	Yes	No	Yes
	Relations	Yes <sup>2</sup>	No	Yes
	Attributes	No	No	No
9. ISA	Classes	Yes	No	Yes
	Relations	Yes	No	Yes
	Attributes	No	No	No
10. Upward Compatibility		Yes	Yes	Yes
11. Snapshot Reducibility		Yes	Yes	Yes
12. Graphical notation provided		Yes	Yes	Yes
13. Graphical Editor		No	No	Yes

### 2.4.1 The Temporal Conceptual Model $\mathcal{ER}_{\mathcal{VT}}$

This Section introduces the semantics of the temporal EER model  $\mathcal{ER}_{\mathcal{VT}}$ .  $\mathcal{ER}_{\mathcal{VT}}$  supports timestamping for classes, attributes, and relationships.  $\mathcal{ER}_{\mathcal{VT}}$  is equipped with a textual and a graphical syntax along with model-theoretic semantics as a temporal extension of the EER semantics [24].

**Definition 2.1** ( $\mathcal{ER}_{\mathcal{VT}}$  Conceptual Data Model). An  $\mathcal{ER}_{\mathcal{VT}}$  conceptual data model is a tuple:  $\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}, \text{ISA}, \text{DISJ}, \text{COVER}, \text{S}, \text{T}, \text{KEY})$ , such that:  $\mathcal{L}$  is a finite alphabet partitioned into the sets:  $\mathcal{C}$  (class symbols),  $\mathcal{A}$  (attribute symbols),  $\mathcal{R}$  (relationship symbols),  $\mathcal{U}$  (role symbols), and  $\mathcal{D}$  (domain symbols) and the set  $\mathcal{C}$  of class symbols is partitioned into a set  $\mathcal{C}^S$  of *Snapshot classes* (marked with an S), a set  $\mathcal{C}^M$  of *Mixed classes* (unmarked classes), and a set  $\mathcal{C}^T$  of *Temporary classes* (marked with a T). A similar partition applies to the set  $\mathcal{R}$ .

<sup>1</sup>Status Relations was only formalised in [8], it was not available in [13]

<sup>2</sup>Transition for relations was formalised in [51], it was not available in [13]



1.  $\text{ATT}$  is a function that maps a class symbol in  $\mathcal{C}$  to an  $\mathcal{A}$ -labeled tuple over  $\mathcal{D}$ ,  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle$ .
2.  $\text{REL}$  is a function that maps a relationship symbol in  $\mathcal{R}$  to an  $\mathcal{U}$ -labeled tuple over  $\mathcal{C}$ ,  $\text{REL}(R) = \langle U_1 : C_1, \dots, U_k : C_k \rangle$ , and  $k$  is the *arity* of  $R$ .
3.  $\text{CARD}$  is a function  $\mathcal{C} \times \mathcal{R} \times \mathcal{U} \mapsto \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$  denoting cardinality constraints. We denote with  $\text{CMIN}(C, R, U)$  and  $\text{CMAX}(C, R, U)$  the first and second component of  $\text{CARD}$ .
4.  $\text{ISA}$  is a binary relationship  $\text{ISA} \subseteq (\mathcal{C} \times \mathcal{C}) \cup (\mathcal{R} \times \mathcal{R})$ .  $\text{ISA}$  between relationships is restricted to relationships with the same arity.  $\text{ISA}$  is visualized with a directed arrow.
5.  $\text{DISJ}$ ,  $\text{COVER}$  are binary relations over  $(2^{\mathcal{C}} \times \mathcal{C}) \times (2^{\mathcal{R}} \times \mathcal{R})$ , describing disjointness and covering partitions, respectively, over a group of  $\text{ISA}$  that share the same superclass/super-relation.  $\text{DISJ}$  is visualized with a circled “d” and  $\text{COVER}$  with a double directed arrow.
6.  $\text{S}$ ,  $\text{T}$  are binary relations over  $\mathcal{C} \times \mathcal{A}$  containing, respectively, the snapshot and temporary attributes of a class;
7.  $\text{KEY}$  is a function,  $\text{KEY} : \mathcal{C} \rightarrow \mathcal{A}$ , that maps a class symbol in  $\mathcal{C}$  to its key attribute. Keys are visualized as underlined attributes.

The model-theoretic semantics associated with the  $\mathcal{ER}_{\mathcal{VT}}$  modelling language adopts the snapshot representation of temporal conceptual data models [27]<sup>3</sup>.

**Definition 2.2** ( $\mathcal{ER}_{\mathcal{VT}}$  Semantics). Let  $\Sigma$  be an  $\mathcal{ER}_{\mathcal{VT}}$  schema. A *temporal database state* for the schema  $\Sigma$  is a tuple  $\mathcal{B} = (\mathcal{T}, \Delta^{\mathcal{I}} \cup \Delta_D^{\mathcal{I}}, \mathcal{I}^{(t)})$ , such that:  $\Delta^{\mathcal{I}}$  is a nonempty set of abstract objects disjoint from  $\Delta_D^{\mathcal{I}}$ ;  $\Delta_D^{\mathcal{I}} = \bigcup_{D_i \in \mathcal{D}} \Delta_{D_i}^{\mathcal{I}}$  is the set of basic domain values used in the schema  $\Sigma$ ; and  $\mathcal{I}^{(t)}$  is a function that for each  $t \in \mathcal{T}$  maps:

- Every basic domain symbol  $D_i$  into a set  $D_i^{\mathcal{I}(t)} = \Delta_{D_i}^{\mathcal{I}}$ .
- Every class  $C$  to a set  $C^{\mathcal{I}(t)} \subseteq \Delta^{\mathcal{I}}$ —thus *objects* are instances of classes.
- Every relationship  $R$  to a set  $R^{\mathcal{I}(t)}$  of  $\mathcal{U}$ -labeled tuples over  $\Delta^{\mathcal{I}}$ —i.e. let  $R$  be an  $n$ -ary relationship connecting the classes  $C_1, \dots, C_n$ ,  $\text{REL}(R) = \langle U_1 : C_1, \dots, U_n : C_n \rangle$ , then,  $r \in R^{\mathcal{I}(t)} \rightarrow (r = \langle U_1 : o_1, \dots, U_n : o_n \rangle \wedge \forall i \in \{1, \dots, n\}. o_i \in C_i^{\mathcal{I}(t)})$ . We adopt the convention:  $\langle U_1 : o_1, \dots, U_n : o_n \rangle \equiv \langle o_1, \dots, o_n \rangle$ , when  $\mathcal{U}$ -labels are clear from the context.
- Every attribute  $A$  to a set  $A^{\mathcal{I}(t)} \subseteq \Delta^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$ , such that, for each  $C \in \mathcal{C}$ , if  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle$ , then,  $o \in C^{\mathcal{I}(t)} \rightarrow (\forall i \in \{1, \dots, h\}, \exists a_i. \langle o, a_i \rangle \in A_i^{\mathcal{I}(t)} \wedge \forall a_i. \langle o, a_i \rangle \in A_i^{\mathcal{I}(t)} \rightarrow a_i \in \Delta_{D_i}^{\mathcal{I}})$ .

<sup>3</sup>Following the snapshot paradigm,  $\mathcal{T}_p$  is a set of time points (or chronons) and  $<$  is a binary precedence relation on  $\mathcal{T}_p$ , the flow of time  $\mathcal{T} = \langle \mathcal{T}_p, < \rangle$  is assumed to be isomorphic to either  $\langle \mathbb{Z}, < \rangle$  or  $\langle \mathbb{N}, < \rangle$ . Thus, standard relational databases can be regarded as the result of mapping a temporal database from time points in  $\mathcal{T}$  to atemporal constructors, with the same interpretation of constants and the same domain.

$\mathcal{B}$  is said a *legal temporal database state* if it satisfies all of the constraints expressed in the schema, i.e. for each  $t \in \mathcal{T}$ :

- For each  $C_1, C_2 \in \mathcal{C}$ , if  $C_1 \text{ ISA } C_2$ , then,  $C_1^{\mathcal{I}(t)} \subseteq C_2^{\mathcal{I}(t)}$ .
- For each  $R_1, R_2 \in \mathcal{R}$ , if  $R_1 \text{ ISA } R_2$ , then,  $R_1^{\mathcal{I}(t)} \subseteq R_2^{\mathcal{I}(t)}$ .
- For each cardinality constraint  $\text{CARD}(C, R, U)$ , then:  
 $o \in C^{\mathcal{I}(t)} \rightarrow \text{CMIN}(C, R, U) \leq \#\{r \in R^{\mathcal{I}(t)} \mid r[U] = o\} \leq \text{CMAX}(C, R, U)$ .
- For  $C, C_1, \dots, C_n \in \mathcal{C}$ , if  $\{C_1, \dots, C_n\} \text{ DISJ } C$ , then,  
 $\forall i \in \{1, \dots, n\}. C_i \text{ ISA } C \wedge \forall j \in \{1, \dots, n\}, j \neq i. C_i^{\mathcal{I}(t)} \cap C_j^{\mathcal{I}(t)} = \emptyset$ .  
(Similar for  $\{R_1, \dots, R_n\} \text{ DISJ } R$ )
- For  $C, C_1, \dots, C_n \in \mathcal{C}$ , if  $\{C_1, \dots, C_n\} \text{ COVER } C$ , then,  
 $\forall i \in \{1, \dots, n\}. C_i \text{ ISA } C \wedge C^{\mathcal{I}(t)} = \bigcup_{i=1}^n C_i^{\mathcal{I}(t)}$ .  
(Similar for  $\{R_1, \dots, R_n\} \text{ COVER } R$ )
- For each snapshot class  $C \in \mathcal{C}^S$ , then,  $o \in C^{\mathcal{I}(t)} \rightarrow \forall t' \in \mathcal{T}. o \in C^{\mathcal{I}(t')}$ .
- For each temporary class  $C \in \mathcal{C}^T$ , then,  $o \in C^{\mathcal{I}(t)} \rightarrow \exists t' \neq t. o \notin C^{\mathcal{I}(t')}$ .
- For each snapshot relationship  $R \in \mathcal{R}^S$ , then,  $r \in R^{\mathcal{I}(t)} \rightarrow \forall t' \in \mathcal{T}. r \in R^{\mathcal{I}(t')}$ .
- For each temporary relationship  $R \in \mathcal{R}^T$ , then,  $r \in R^{\mathcal{I}(t)} \rightarrow \exists t' \neq t. r \notin R^{\mathcal{I}(t')}$ .
- For each class  $C \in \mathcal{C}$ , if  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle$ , and  $\langle C, A_i \rangle \in \mathbf{s}$ , then,  
 $(o \in C^{\mathcal{I}(t)} \wedge \langle o, a_i \rangle \in A_i^{\mathcal{I}(t)}) \rightarrow \forall t' \in \mathcal{T}. \langle o, a_i \rangle \in A_i^{\mathcal{I}(t')}$ .
- For each class  $C \in \mathcal{C}$ , if  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle$ , and  $\langle C, A_i \rangle \in \mathbf{T}$ , then,  
 $(o \in C^{\mathcal{I}(t)} \wedge \langle o, a_i \rangle \in A_i^{\mathcal{I}(t)}) \rightarrow \exists t' \neq t. \langle o, a_i \rangle \notin A_i^{\mathcal{I}(t')}$ .
- For each  $C \in \mathcal{C}, A \in \mathcal{A}$  such that  $\text{KEY}(C) = A$ , then,  $A$  is a snapshot attribute—i.e.  
 $\langle C, A_i \rangle \in \mathbf{s}$ — and  $\forall a \in \Delta_D^{\mathcal{I}}. \#\{o \in C^{\mathcal{I}(t)} \mid \langle o, a \rangle \in A^{\mathcal{I}(t)}\} \leq 1$ .

Given such a set-theoretic semantics for the temporal EER (or, for that matter, UML class diagrams or ORM), some relevant modelling notions such as satisfiability, subsumption, and derivation of new constraints by means of logical implication have been defined rigorously [13].

**Definition 2.3** (Reasoning Services). Let  $\Sigma$  be a schema,  $C \in \mathcal{C}$  a class, and  $R \in \mathcal{R}$  a relationship. The following modelling notions can be defined:

1.  $C (R)$  is *satisfiable* if there exists a legal temporal database state  $\mathcal{B}$  for  $\Sigma$  such that  $C^{\mathcal{I}(t)} \neq \emptyset$  ( $R^{\mathcal{I}(t)} \neq \emptyset$ ), for some  $t \in \mathcal{T}$ ;
2.  $\Sigma$  is *satisfiable* if there exists a legal temporal database state  $\mathcal{B}$  for  $\Sigma$  ( $\mathcal{B}$  is also said a *model* for  $\Sigma$ );

$$\begin{aligned}
C &\rightarrow \top \mid \perp \mid CN \mid \neg C \mid C_1 \sqcap C_2 \mid \exists^{\leq k}[U_j]R \mid \\
&\quad \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 RN \mid \neg R \mid R_1 \sqcap R_2 \mid U_i/n : C \mid \\
&\quad \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 \\
\\
\top^{\mathcal{I}(t)} &= \Delta^{\mathcal{I}} \\
\perp^{\mathcal{I}(t)} &= \emptyset \\
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^{\leq k}[U_j]R)^{\mathcal{I}(t)} &= \{d \in \top^{\mathcal{I}(t)} \mid \#\{\langle d_1, \dots, d_n \rangle \in R^{\mathcal{I}(t)} \mid d_j = d\} \leq k\} \\
(C_1 \mathcal{U} C_2)^{\mathcal{I}(t)} &= \{d \in \top^{\mathcal{I}(t)} \mid \exists v > t. (d \in C_2^{\mathcal{I}(v)} \wedge \forall w \in (t, v). d \in C_1^{\mathcal{I}(w)})\} \\
(C_1 \mathcal{S} C_2)^{\mathcal{I}(t)} &= \{d \in \top^{\mathcal{I}(t)} \mid \exists v < t. (d \in C_2^{\mathcal{I}(v)} \wedge \forall w \in (v, t). d \in C_1^{\mathcal{I}(w)})\} \\
(\top_n)^{\mathcal{I}(t)} &\subseteq (\Delta^{\mathcal{I}})^n \\
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 d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid d_i \in C^{\mathcal{I}(t)}\} \\
(R_1 \mathcal{U} R_2)^{\mathcal{I}(t)} &= \{\langle d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \\
&\quad \exists v > t. (\langle d_1, \dots, d_n \rangle \in R_2^{\mathcal{I}(v)} \wedge \forall w \in (t, v). \langle d_1, \dots, d_n \rangle \in R_1^{\mathcal{I}(w)})\} \\
(R_1 \mathcal{S} R_2)^{\mathcal{I}(t)} &= \{\langle d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \\
&\quad \exists v < t. (\langle d_1, \dots, d_n \rangle \in R_2^{\mathcal{I}(v)} \wedge \forall w \in (v, t). \langle d_1, \dots, d_n \rangle \in R_1^{\mathcal{I}(w)})\} \\
(\diamond^+ R)^{\mathcal{I}(t)} &= \{\langle d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v > t. \langle d_1, \dots, d_n \rangle \in R^{\mathcal{I}(v)}\} \\
(\oplus R)^{\mathcal{I}(t)} &= \{\langle d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle d_1, \dots, d_n \rangle \in R^{\mathcal{I}(t+1)}\} \\
(\diamond^- R)^{\mathcal{I}(t)} &= \{\langle d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v < t. \langle d_1, \dots, d_n \rangle \in R^{\mathcal{I}(v)}\} \\
(\ominus R)^{\mathcal{I}(t)} &= \{\langle d_1, \dots, d_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle d_1, \dots, d_n \rangle \in R^{\mathcal{I}(t-1)}\}
\end{aligned}$$

FIGURE 2.1: Syntax and semantics of  $\mathcal{DLR}_{US}$ .

3.  $C_1$  ( $R_1$ ) is *subsumed* by  $C_2$  ( $R_2$ ) in  $\Sigma$  if every legal temporal database state for  $\Sigma$  is also a legal temporal database state for  $C_1$  ISA  $C_2$  ( $R_1$  ISA  $R_2$ );
4. A schema  $\Sigma'$  is *logically implied* by a schema  $\Sigma$  over the same signature if every legal temporal database state for  $\Sigma$  is also a legal temporal database state for  $\Sigma'$ .

## 2.5 $\mathcal{DLR}_{US}$

The temporal description logic  $\mathcal{DLR}_{US}$  [6] combines the propositional temporal logic with the *Since* and *Until* operators and the (non-temporal) description logic  $\mathcal{DLR}$  [16, 23] that serves as common foundational language for various conceptual data modeling languages. The formal foundations of  $\mathcal{ER}_{VT}$  allowed also to prove a correct encoding of  $\mathcal{ER}_{VT}$  schemas as knowledge base in  $\mathcal{DLR}_{US}$  [4, 6].  $\mathcal{DLR}_{US}$  can be regarded as an expressive fragment of the first-order temporal logic  $L^{\{\text{since, until}\}}$  [27, 43]. The basic syntactical types of  $\mathcal{DLR}_{US}$  are *classes* (also known as *entity types* or *object types*) and *n-ary relations* (associations) of arity  $\geq 2$ . Starting from a set of *atomic classes* (denoted by  $CN$ ), a set of *atomic relations* (denoted by  $RN$ ), and a set of *role symbols* (denoted by

$U$ , comparable to an ORM-role or component of a UML association) we can define inductively (complex) class and relation expressions (see upper part of Figure 2.1), where the binary constructors  $(\sqcap, \sqcup, \mathcal{U}, \mathcal{S})$  are applied to relations of the same arity,  $i, j, k, n$  are natural numbers,  $i \leq n$ , and  $j$  does not exceed the arity of  $R$ . Observe that for both class and relation expressions all the Boolean constructors are available. The selection expression  $U_i/n : C$  denotes an  $n$ -ary relation whose  $i$ -th argument ( $i \leq n$ ), named  $U_i$ , is of type  $C$ . (In ORM terminology,  $U_i/n : C$  refers to the role  $U_i$  played by  $C$  in the fact type.) If it is clear from the context, we omit  $n$  and simply write  $(U_i : C)$ . The projection expression  $\exists^{\leq k}[U_j]R$  is a generalisation with cardinalities of the projection operator over argument  $U_j$  of relation  $R$ ; the plain classical projection is  $\exists^{\geq 1}[U_j]R$ . It is also possible to use the pure argument position version of the language by replacing role symbols  $U_i$  with their corresponding position numbers  $i$ .

The model-theoretic semantics of  $\mathcal{DLR}_{\mathcal{US}}$  assumes a flow of time  $\mathcal{T} = \langle \mathcal{T}_p, < \rangle$ , where  $\mathcal{T}_p$  is a set of time points (also called chronons) and  $<$  a binary precedence relation on  $\mathcal{T}_p$ , which is assumed to be isomorphic to  $\langle \mathbb{Z}, < \rangle$ . The language of  $\mathcal{DLR}_{\mathcal{US}}$  is interpreted in *temporal models* over  $\mathcal{T}$ , which are triples of the form  $\mathcal{I} \doteq \langle \mathcal{T}, \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}(t)} \rangle$ , where  $\Delta^{\mathcal{I}}$  is non-empty set of objects (the *domain* of  $\mathcal{I}$ ) and  $\cdot^{\mathcal{I}(t)}$  an *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  $C^{\mathcal{I}(t)} \subseteq \Delta^{\mathcal{I}}$  and  $R^{\mathcal{I}(t)} \subseteq (\Delta^{\mathcal{I}})^n$ . The semantics of class and relation expressions is defined in the lower part of Figure 2.1, where  $(u, v) = \{w \in \mathcal{T} \mid u < w < v\}$ . We will use the following equivalent abbreviations:  $C_1 \sqcup C_2 \equiv \neg(\neg C_1 \sqcap \neg C_2)$ ;  $C_1 \rightarrow C_2 \equiv \neg C_1 \sqcup C_2$ ;  $\exists[U]R \equiv \exists^{\geq 1}[U]R$ ;  $\forall[U]R \equiv \neg \exists[U]\neg R$ ;  $R_1 \sqcup R_2 \equiv \neg(\neg R_1 \sqcap \neg R_2)$ . Furthermore, the operators  $\diamond^*$  (at some moment) and its dual  $\square^*$  (at all moments) can be defined for both classes and relations as  $\diamond^*C \equiv C \sqcup \diamond^+C \sqcup \diamond^-C$  and  $\square^*C \equiv C \sqcap \square^+C \sqcap \square^-C$ ,  $\diamond^*R \equiv R \sqcup \diamond^+R \sqcup \diamond^-R$  and  $\square^*R \equiv R \sqcap \square^+R \sqcap \square^-R$  respectively.

Until and Since together with  $\perp$  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  $\oplus C \equiv \perp \mathcal{U} C$ , and likewise for their past counterparts;  $\square^+$  (always in the future) and  $\square^-$  (always in the past) are the duals of  $\diamond^+$  and  $\diamond^-$  respectively i.e.  $\square^+C \equiv \neg \diamond^+ \neg C$  and  $\square^-C \equiv \neg \diamond^- \neg C$  for both classes and relations.

To show the mapping of  $\mathcal{DLR}_{\mathcal{US}}$  to  $\mathcal{ER}_{\mathcal{VT}}$ , we use the example of an undergraduate student who evolves into a postgraduate. A student can either be an undergraduate or postgraduate student.

$EEER_{\mathcal{VT}}^{++}$  textual syntax as UnderGraduate ISA Student, PostGraduate ISA Student and its identifier as  $\text{ID}(\text{Student}) = \text{StdID}$ , course = Mathematics, age = 26, with its  $\mathcal{DLR}_{\mathcal{US}}$  notation as  $\text{UnderGraduate} \sqsubseteq \text{Student}$ ,  $\text{PostGraduate} \sqsubseteq \text{Student}$ , and  $\text{UnderGraduate} \sqsubseteq \exists^1[\text{From}]\square^*\text{StdID}$ ,  $\text{UnderGraduate} \sqsubseteq \diamond^+\text{Student}$ ,

with  $\top \sqsubseteq \exists^{\leq 1}[\text{To}](\text{StdID} \sqcap \text{From} : \text{Student})$ , respectively. Disjoint classes, UnderGraduate and PostGraduate are written as  $\text{UnderGraduate} \sqcap \text{PostGraduate} = \perp$ .

## 2.6 Summary

The literature review gave an overview of temporal modelling research, the definitions used as well as an introduction of description logic  $\mathcal{DLR}_{\mathcal{US}}$ . The existing criteria for modelling temporal conceptual models was modified for modelling logic based temporal conceptual models. Then discussed and compared three temporal conceptual data models, against the criteria developed to establish the model that had most work done on it. We chose  $\mathcal{ER}_{\mathcal{VT}}$  and gave its definition and its semantics. The next chapter introduces the extended  $\mathcal{ER}_{\mathcal{VT}}$  model as well as the formalisation of temporal attributes.



# 3

## TEMPORAL ATTRIBUTES

This chapter gives a rigorous formalisation of temporal attributes in temporal conceptual data models. The extended  $\mathcal{DLR}_{US}$  is introduced to enable formalisation of temporal attributes, which are both functional and mandatory, in terms of timestamping and evolution constraints. This entails the definition of status attributes and transition constraints as well as the interaction between temporal classes and temporal attributes, which leads to logical implications and refinement from the previous formalisation.

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

The Description Logic  $\mathcal{DLR}_{US}$  [6] is an expressive fragment of first order logic (FOL) that combines the propositional temporal logic with the temporal operators *Since* and *Until* and the (non-temporal) description logic  $\mathcal{DLR}$  [22].  $\mathcal{DLR}_{US}$  is extended to include a precise syntax and semantics for temporal attributes, although attributes are binary relations, the semantics are added for a better formalisation of temporal attributes. Artale et al. [13] defined a temporal attribute as a binary relation for each attribute,  $A \in \mathcal{A}$ , for  $\langle A, C \rangle \in \mathcal{T}$ , with its  $\mathcal{DLR}_{US}$  axiom as  $C \sqsubseteq \neg \exists [\text{From}] (\Box^* A)$ , without explicitly defining it in the model. The use of temporal constructors without being included in the  $\mathcal{DLR}_{US}$  syntax and semantics created a gap in this area and we seek to address it. This now allows us to properly formalise temporal attributes in a temporal conceptual model.

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  $C^{\mathcal{I}(t)} \subseteq \Delta_O^{\mathcal{I}}$ ,  $R^{\mathcal{I}(t)} \subseteq (\Delta_O^{\mathcal{I}})^n$  and  $A^{\mathcal{I}(t)} \subseteq (\Delta_O^{\mathcal{I}} \cup \Delta_D^{\mathcal{I}})$ ; also,  $(u, v) = \{w \in \mathcal{T} \mid u < w < v\}$ . Note that in [6, 13],  $\Delta_D^{\mathcal{I}}$  was already used in the  $\mathcal{DLR}_{US}$ -based logic reconstruction of  $\mathcal{ER}_{VT}$ , and

$\Delta^{\mathcal{I}} = \Delta_O^{\mathcal{I}} \cup \Delta_D^{\mathcal{I}}$ , but was not explicitly stated in the  $\mathcal{DLR}_{US}$  syntax and semantics; it is now in Fig. 3.1.

$$\begin{aligned}
C &\rightarrow \top \mid \perp \mid CN \mid \neg C \mid C_1 \sqcap C_2 \mid \exists^{\leq k}[U_j]R \mid \exists[\mathbf{F}]\mathbf{A} \mid \\
&\quad \diamond^+ C \mid \diamond^- C \mid \square^+ C \mid \square^- 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 RN \mid \neg R \mid R_1 \sqcap R_2 \mid U_i/n : C \mid \\
&\quad \diamond^+ R \mid \diamond^- R \mid \square^+ R \mid \square^- 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 \\
&\quad \diamond^+ \mathbf{A} \mid \diamond^- \mathbf{A} \mid \square^+ \mathbf{A} \mid \square^- \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 \\
\mathbf{D} &\rightarrow \top_{\mathbf{D}} \mid \perp_{\mathbf{D}} \mid \{d_1, \dots, d_n\} \\
\\ 
\top^{\mathcal{I}(t)} &= \Delta_O^{\mathcal{I}} \\
\perp^{\mathcal{I}(t)} &= \emptyset \\
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^{\leq k}[U_j]R)^{\mathcal{I}(t)} &= \{ o \in \top^{\mathcal{I}(t)} \mid \#\{\langle o_1, \dots, o_n \rangle \in R^{\mathcal{I}(t)} \mid o_j = o\} \leq k \} \\
(\exists[\mathbf{F}]\mathbf{A})^{\mathcal{I}(t)} &= \{ \mathbf{o} \in \top^{\mathcal{I}(t)} \mid \#\{\langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}^{\mathcal{I}(\mathbf{t})} \mid \mathbf{d} \geq 1\} \} \\
(C_1 \mathcal{U} C_2)^{\mathcal{I}(t)} &= \{ o \in \top^{\mathcal{I}(t)} \mid \exists v > t. (o \in C_1^{\mathcal{I}(v)} \wedge \forall w \in (t, v). o \in C_2^{\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_O^{\mathcal{I}})^n \\
(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, \dots, 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, \dots, o_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v > t. (\langle o_1, \dots, o_n \rangle \in R_2^{\mathcal{I}(v)} \wedge \\
&\quad \forall w \in (t, v). \langle o_1, \dots, o_n \rangle \in R_1^{\mathcal{I}(w)}) \} \\
(R_1 \mathcal{S} R_2)^{\mathcal{I}(t)} &= \{ \langle o_1, \dots, o_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v < t. (\langle o_1, \dots, o_n \rangle \in R_2^{\mathcal{I}(v)} \wedge \\
&\quad \forall w \in (v, t). \langle o_1, \dots, o_n \rangle \in R_1^{\mathcal{I}(w)}) \} \\
(\diamond^+ R)^{\mathcal{I}(t)} &= \{ \langle o_1, \dots, o_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v > t. \langle o_1, \dots, o_n \rangle \in R^{\mathcal{I}(v)} \} \\
(\oplus R)^{\mathcal{I}(t)} &= \{ \langle o_1, \dots, o_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle o_1, \dots, o_n \rangle \in R^{\mathcal{I}(t+1)} \} \\
(\diamond^- R)^{\mathcal{I}(t)} &= \{ \langle o_1, \dots, o_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \exists v < t. \langle o_1, \dots, o_n \rangle \in R^{\mathcal{I}(v)} \} \\
(\ominus R)^{\mathcal{I}(t)} &= \{ \langle o_1, \dots, o_n \rangle \in (\top_n)^{\mathcal{I}(t)} \mid \langle o_1, \dots, o_n \rangle \in R^{\mathcal{I}(t-1)} \} \\
(\top_{\mathbf{D}})^{\mathcal{I}(t)} &= \Delta_D^{\mathcal{I}} \\
(\perp_{\mathbf{D}})^{\mathcal{I}(t)} &= \emptyset \\
(\top_{\mathbf{A}})^{\mathcal{I}(t)} &= \Delta_O^{\mathcal{I}} \times \Delta_D^{\mathcal{I}} \\
(\mathbf{AN})^{\mathcal{I}(t)} &\subseteq (\top_{\mathbf{A}})^{\mathcal{I}(t)} \\
(\mathbf{F} : \mathbf{C})^{\mathcal{I}(t)} &= \{ \langle \mathbf{o}, \mathbf{d} \rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(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}(t)} \mid \exists v > t. (\langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}_2^{\mathcal{I}(v)} \wedge \forall w \in (t, v). \langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}_1^{\mathcal{I}(w)}) \} \\
(\mathbf{A}_1 \mathcal{S} \mathbf{A}_2)^{\mathcal{I}(t)} &= \{ \langle \mathbf{o}, \mathbf{d} \rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(t)} \mid \exists v < t. (\langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}_2^{\mathcal{I}(v)} \wedge \forall w \in (v, t). \langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}_1^{\mathcal{I}(w)}) \} \\
(\diamond^+ \mathbf{A})^{\mathcal{I}(t)} &= \{ \langle \mathbf{o}, \mathbf{d} \rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(t)} \mid \exists v > t. \langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}^{\mathcal{I}(v)} \} \\
(\oplus \mathbf{A})^{\mathcal{I}(t)} &= \{ \langle \mathbf{o}, \mathbf{d} \rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(t)} \mid \langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}^{\mathcal{I}(t+1)} \} \\
(\diamond^- \mathbf{A})^{\mathcal{I}(t)} &= \{ \langle \mathbf{o}, \mathbf{d} \rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(t)} \mid \exists v < t. \langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}^{\mathcal{I}(v)} \} \\
(\ominus \mathbf{A})^{\mathcal{I}(t)} &= \{ \langle \mathbf{o}, \mathbf{d} \rangle \in (\top_{\mathbf{A}})^{\mathcal{I}(t)} \mid \langle \mathbf{o}, \mathbf{d} \rangle \in \mathbf{A}^{\mathcal{I}(t-1)} \}
\end{aligned}$$

FIGURE 3.1: Syntax and semantics of  $\mathcal{DLR}_{US}$ , 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.

The basic syntactic types of the extended  $\mathcal{DLR}_{US}$  are 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, binary constructs ( $\sqcup, \sqcap, \mathcal{U}, \mathcal{S}$ ) supplied to relations of the same arity,  $i, j, k, n$ , which are natural numbers  $i < n$  and  $j$  does not exceed arity of  $R$ . DL role components ( $U$ , for relations and  $F$  for attributes),  $F$  denotes a role component in an attribute,  $F \subseteq U$ , and  $\{\text{From}, \text{To}\} \subseteq F$ . The selection expression  $U_i/n : C$  denotes an  $n$ -ary relation whose  $i$ -th argument ( $i \leq n$ ) is of type  $C$ , and  $F : C$  denotes is



the role component  $\{\text{From}\}$  in the attribute, to the datatype, e.g.  $A \sqsubseteq \text{From} : C \sqcap [To]D$ , is an attribute  $A$  in class  $C$  which has the datatype  $D$ . A datatype  $\perp_D$  denotes the empty set, while  $\top_D$  is the set of all possible values. The range of datatypes is defined as  $\{d_1, \dots, d_n\}$ . In the semantics of attributes,  $a$  represents  $\langle o, d \rangle \in A$ , where  $o$  is the object and  $d$  the datatype. *Until* and *Since* together with  $\perp$  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  $\oplus C \equiv \perp \mathcal{U} C$ , and likewise for their past counterparts;  $\square^+$  (always in the future) and  $\square^-$  (always in the past) are the duals of  $\diamond^+$  and  $\diamond^-$  respectively i.e.  $\square^+ C \equiv \neg \diamond^+ \neg C$  and  $\square^- C \equiv \neg \diamond^- \neg C$  for both classes and relations. We will use the following equivalent abbreviations:  $C_1 \sqcup C_2 \equiv \neg(\neg C_1 \sqcap \neg C_2)$ ;  $C_1 \rightarrow C_2 \equiv \neg C_1 \sqcup C_2$ ;  $\exists[U]R \equiv \exists_{\geq 1}[U]R$ ;  $\forall[U]R \equiv \neg \exists[U]\neg R$ ;  $R_1 \sqcup R_2 \equiv (\neg R_1 \sqcap \neg R_2)$ . Furthermore, the operators  $\diamond^*$  (at some moment) and its dual  $\square^*$  (at all moments) can be defined for both classes and relations as  $\diamond^* C \equiv C \sqcup \diamond^+ C \sqcup \diamond^- C$  and  $\square^* C \equiv C \sqcap \square^+ C \sqcap \square^- C$  respectively. A *knowledge base* is a finite set  $\Sigma$  of  $\mathcal{DLR}_{\mathcal{US}}$  axioms of the form  $C_1 \sqsubseteq C_2$  and  $R_1 \sqsubseteq R_2$ , and  $A_1 \sqsubseteq A_2$ , 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  $t \in \mathcal{T}$ . The next section uses the extension of  $\mathcal{DLR}_{\mathcal{US}}$  to give the full temporal constraints in temporal models.

## 3.2 Temporal constraints

$\mathcal{ER}_{\mathcal{VT}}$  defined attributes as a binary relation,  $\mathcal{C} \times \mathcal{A}$ , which is inaccurate, because attributes  $\mathcal{A}$  are the set of binary relations from the set of classes and the domain symbols  $\mathcal{D}$ , which gives us the definition  $\mathcal{C} \times \mathcal{D}$ . As a consequence of this, we change the semantics of attributes from  $\langle o, a \rangle$ , where  $o$  is the object and  $a$  is the attribute to  $\langle o, d \rangle \in A$ , where  $d$  is the datatype. We use this definition to define temporal constraints.

### 3.2.1 Timestamping

Timestamping puts a temporal mechanism that positions data on a time-scale [3, 13] to keep track on which elements ought to be recorded. Timestamping allows one to distinguish between temporal and atemporal attributes to keep track of how an attribute changes over time. On the  $\mathcal{ER}_{\mathcal{VT}}$  model, the symbols S and T are inserted in the temporal ER model to represent Snapshot and Temporal. With the correction as stated above, we give definition of attribute timestamping as:

#### *Snapshot attribute*

These are attributes that are time invariant, they do not change over time.

$$o \in C^{\mathcal{I}(t)} \wedge \langle o, d \rangle \in A^{\mathcal{I}(t)} \rightarrow \forall t' \in \mathcal{T}. \langle o, d \rangle \in A^{\mathcal{I}(t')}$$

$$C \sqsubseteq \neg \exists [\text{From}] : (A \sqcap \Diamond^* \neg A)$$

### Temporal attribute

These are attributes that do not have the same value at all times.

$$o \in C^{\mathcal{I}(t)} \wedge \langle o, d \rangle \in A^{\mathcal{I}(t)} \rightarrow \exists t' \neq t. \langle o, d \rangle \notin A^{\mathcal{I}(t')}$$

$$C \sqsubseteq \neg \exists [\text{From}] : (\Box^* A)$$

## 3.2.2 Evolution constraints

Evolution constraints are rules that govern transitions and valid states of the knowledge base or logic-based temporal conceptual data models. These rules control the mechanism that rules dynamic aspects, permissible transitions from one state to the next [13], as well as the lifespan. According to Hall and Gupta [39] temporal modelling addresses the question *how does the world change?* One way is lifecycle, the evolving state of membership of an object from its creation to its end, covered in [13, 51] for objects and relations respectively. This is defined by extending the notion of status classes and status relations to *status attributes* to govern the evolution of temporal attributes, there after we present its formalisation.

### 3.2.2.1 Lifecycle

Lifecycle can be viewed in terms of *status* which records the evolving membership of an object, relation and attributes. Status models the normal behaviour in the real world; items either exist or do not (disabled). If they exist, they are *scheduled*, to become *active* in the future, as time passes, they can be *suspended* and reactivated and finally may become *disabled* when they expire. These four statuses can be used to model evolution of data, and are represented in an EER diagram as shown in Fig. 3.2. Status classes were introduced in [13] and status relations in [7], but it fell short of status attributes to constrain the permissible states of affairs. To represent temporal attributes to the level of detail required and to have them interact with temporalised classes, we extend the notion of status classes and status relations to *status attributes* to specify the operational semantics of temporal attributes. Status attributes also has four different statuses: exist, scheduled, active, or suspended, and disabled. We describe each one of them informally and illustrate that they are relevant for conceptual modelling and knowledge representation, and subsequently introduce the formalisation.

- *Scheduled*: an attribute is scheduled if it belongs to an active class or a scheduled class. For instance, a bonus payout to an employee occurs only after passing the probationary period successfully.
- *Active*: the status of an attribute is active if it fully instantiates the type-level attribute, thus, these are the normal temporal attributes and they only belong to

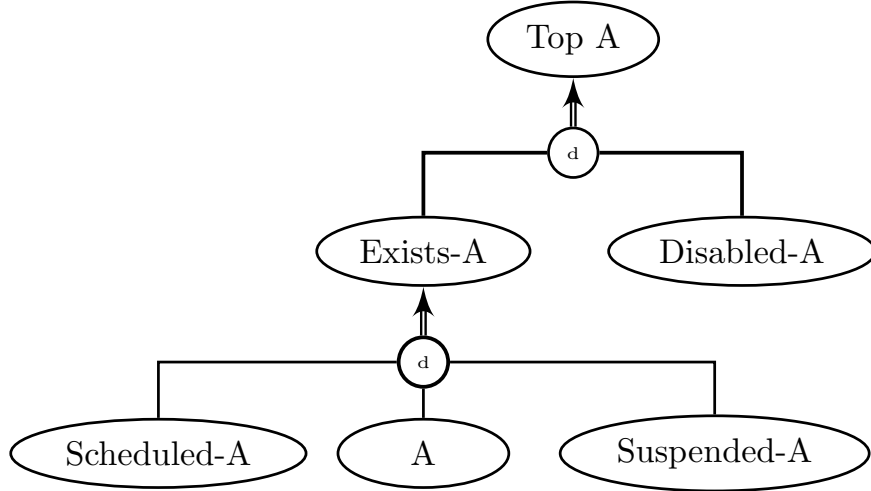


FIGURE 3.2: EER diagram with the attribute hierarchy of status attributes.

an active class. They can change at any time in the class; e.g., an access level to certain information, date of birth, an employee's salary attribute.

- *Suspended*: These are attributes that belong to either a suspended class or an active class. They are temporarily inactive and will become active after a given period of time or until the suspended class becomes active. For instance, an employee is suspended due to an ongoing fraud case, so its attributes are suspended.
- *Disabled*: These belong to either a disabled class or an active class. When the membership of the class has expired, its attributes are also disabled, and it can also be true for an active class that no longer requires the use of that attribute. Disabled attributes is an irreversible form of suspended, meaning that once the attribute is disabled it can never be activated again, it is in a permanent freeze state. For instance, an employee quits, so the profile is disabled, or a software company decides to change the application from for-payment to free and open source, so that the price attribute becomes disabled.

Concerning the formalization, the subsumption hierarchy and disjointness depicted in Fig. 3.3 are straightforward and omitted from the formalisation, status classes and status attributes are combined to show the interaction and constraints of status attributes in status classes with the disjointness and subsumption hierarchy. The rectangular boxes are status classes 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 and disabled respectively. We give both the model-theoretic semantics and the  $\mathcal{DLR}_{\mathcal{US}}$  axiom;  $a$  is an element in  $A$ , which can also be written as  $\langle o, d \rangle$ , where  $o$  represents an object in the class and  $d$  is the domain of the class.

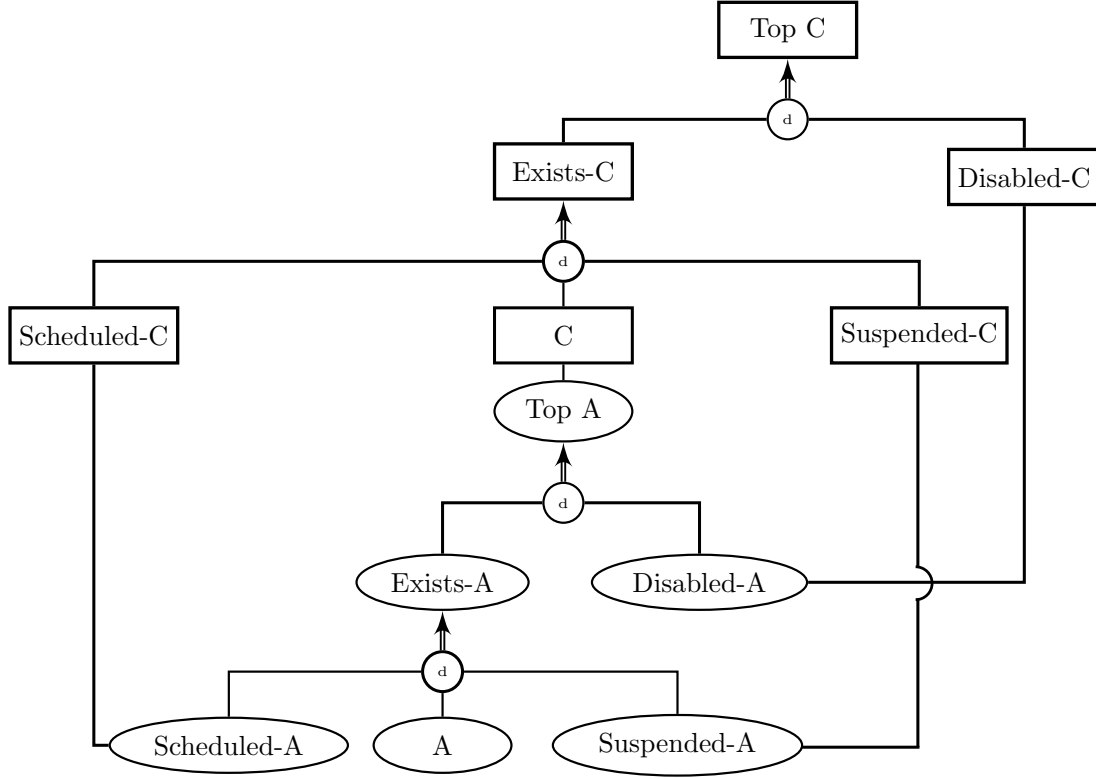


FIGURE 3.3: EER diagram with the class hierarchy of status classes (rectangles) integrated with the attribute hierarchy of status attributes (ovals).

Some of the axioms from status classes [13], and status relations [8] are reused in this formalisation, (see Appendix A). The axioms in the previous work stated that “Existence persists until Disabled”. We change the framing of the sentence to now read as “Class Existence persists unless disabled”. We use the axiom “Attribute Existence persists unless Disabled”, for both classes and attributes, which will be proven in Proposition 3.2.

### Status Classes: Axioms

(EXISTS) *Class Existence persists unless disabled*

$$o \in \text{Exists-}C^{\mathcal{I}(t)} \rightarrow \forall t' > t. o \in (\text{Exists-}C^{\mathcal{I}(t')} \vee \text{Disabled-}C^{\mathcal{I}(t')})$$

$$\text{Exists-C} \sqsubseteq \Box^+(\text{Exists-C} \sqcup \text{Disabled-C})$$

### Status attributes: Axioms

(AEXISTS1) *Attribute existence persists unless disabled*

$$a \in \text{Exists-}A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \in (\text{Exists-}A^{\mathcal{I}(t')} \vee \text{Disabled-}A^{\mathcal{I}(t')})$$

$$\text{Exists-A} \sqsubseteq \Box^+(\text{Exists-A} \sqcup \text{Disabled-A})$$

(AEXISTS2) *Exist attribute involves scheduled, suspended or active.*

$$a \in \text{Exists-}A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \in (\text{Scheduled-}A^{\mathcal{I}(t')} \vee A^{\mathcal{I}(t')} \vee \text{Suspended-}A^{\mathcal{I}(t')})$$

$$\text{Exists-A} \sqsubseteq \Box^+(\text{Scheduled-A} \sqcup A \sqcup \text{Suspended-A})$$

(AEXISTS3) *Existing attributes belong to an existing class.*

$$\langle o, d \rangle \in \text{Exists-}A^{\mathcal{I}(t)} \rightarrow o \in \text{Exists-}C^{\mathcal{I}(t)}$$

$$\text{Exists-A} \sqsubseteq \text{From} : \text{Exists-C}$$

(AACT1) *Active attributes belong to an active class only.*

$$\langle o, d \rangle \in A^{\mathcal{I}(t)} \rightarrow o \in C^{\mathcal{I}(t)}$$

$$A \sqsubseteq \text{From} : C$$

(ASCH1) *Scheduled attribute will eventually become active.*

$$a \in \text{Scheduled-}A^{\mathcal{I}(t)} \rightarrow \exists t' > t. a \in A^{\mathcal{I}(t')}$$

$$\text{Scheduled-A} \sqsubseteq \Diamond^+ A$$

(ASCH2) *Scheduled attribute can never follow active.*

$$a \in A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \notin \text{Scheduled-}A^{\mathcal{I}(t')}$$

$$A \sqsubseteq \Box^+ \neg \text{Scheduled-A}$$

(ASUSP1) *Suspended attribute was active in the past.*

$$a \in \text{Suspended-}A^{\mathcal{I}(t)} \rightarrow \exists t' < t. a \in A^{\mathcal{I}(t')}$$

$$\text{Suspended-A} \sqsubseteq \Diamond^- A$$

(ASUSP2) *Suspended attributes belong to active or suspended class.*

$$\langle o, d \rangle \in \text{Suspended-}A^{\mathcal{I}(t)} \rightarrow o \in (\text{Suspended-}C^{\mathcal{I}(t)} \vee C^{\mathcal{I}(t)})$$

$$\text{Suspended-A} \sqsubseteq \text{From} : (\text{Suspended-C} \sqcup C)$$

(ADISAB1) *Disabled persists.*

$$a \in \text{Disabled-}A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \in \text{Disabled-}A^{\mathcal{I}(t')}$$

$$\text{Disabled-A} \sqsubseteq \Box^+ \text{Disabled-A}$$

(ADISAB2) *Disabled attribute was active in the past.*

$$a \in \text{Disabled-}A^{\mathcal{I}(t)} \rightarrow \exists t' < t. a \in A^{\mathcal{I}(t')}$$

$$\text{Disabled-A} \sqsubseteq \Diamond^- A$$

(ADISAB3) *Disabled attributes belong to a disabled or active class.*

$$\langle o, d \rangle \in \text{Disabled-}A^{\mathcal{I}(t)} \rightarrow o \in (\text{Disabled-}C^{\mathcal{I}(t)} \vee C^{\mathcal{I}(t)})$$

$$\text{Disabled-A} \sqsubseteq \text{From} : (\text{Disabled-C} \sqcup C)$$

(CSUSP3) *Freezing attributes of suspended classes / Suspended class has suspended attributes only.*

$$o \in \text{Suspended-}C^{\mathcal{I}(t)} \rightarrow \langle o, d \rangle \in \text{Suspended-}A^{\mathcal{I}(t)}$$

$$\text{Suspended-C} \sqsubseteq \forall [\text{From}] \text{Suspended-A}$$

(CACTIVE) *An active class contains only exists or disabled attributes.*

$$o \in C^{\mathcal{I}(t)} \rightarrow \langle o, d \rangle \in (\text{Exists-}A^{\mathcal{I}(t)} \vee \text{Disabled-}A^{\mathcal{I}(t)})$$

$$C \sqsubseteq \forall [\text{From}] (\text{Exists-A} \sqcup \text{Disabled-A})$$

Henceforth, we denote with  $\Sigma_{sa}$  the above set of  $\mathcal{DLR}_{US}$  axioms that formalise status attributes, status classes and status relations. Some axioms are quite similar to those for relationships introduced in [7], others are specific to attributes. Similar ones are: AACT1 corresponds to Artale et. al's [7] ACT, ADISAB1 to RDISAB1, ADISAB2 to RDISAB2, ASUSP1 to RSUSP1, ASUSP2 to RSUSP2, ASCH1 to RSCH1, and ASCH2 to RSCH2. The new ones specific to attributes are: AEXISTS1, AEXISTS2, AEXISTS3 and ADISAB3.

CSUSP3 is the same axiom as Artale et. al. [13] FREEZ which intends to capture "Freezing attributes of suspended classes" so as "to make attributes of suspended objects unchangeable". The unchangeability starts immediately the object becomes "suspended" whose idea originated from [33], with a semantics  $o \in \text{Suspended-}C^{\mathcal{I}(t)} \wedge \langle o, a \rangle \in A^{\mathcal{I}(t)} \rightarrow \langle o, a \rangle \in A^{\mathcal{I}(t-1)}$  and in  $\mathcal{DLR}_{US}$  notation  $\text{Suspended-C} \sqsubseteq \neg \exists [\text{From}] (A \sqcap \ominus A)$  [13]. However, suspended requires that it was active in the past, which applies to classes, relationships and attributes (ASUSP1), and it is not just that the attribute may not be 'not-active', but, it has to be suspended, too. Hence, CSUSP3 captures the 'freezing' more precisely. Observe that, as with ACT for relations (Active relations involve only active classes) [7], AACT1 cannot be proven and therefore had to be added to the set of basic constraints: while by AEXISTS3, we know an attribute has to be either scheduled, active, or suspended, and one can exclude suspended thanks to CSUSP3, one cannot contradict  $a \in \text{Scheduled-}A^{\mathcal{I}(t)}$  due to ASCH3, as scheduled attributes may belong to either scheduled or active classes.

### Status Attributes: Logical Implications

Logical implications are important to derive new constraints. From the  $\mathcal{DLR}_{US}$  axioms above ( $\Sigma_{sa}$ ), the following logical implications are obtained.

**Proposition 3.1 (Status Classes: Logical Implications).** *Proof.*

(EXIST1) *Class Existence persists unless disabled.*

$$\Sigma_{sa} \models \text{Exists-C} \sqsubseteq \text{Exists-C} \sqcup \text{Disabled-C}$$

Let  $o \in \text{Exists-C}^{\mathcal{I}(t)}$  and by

$$(\text{EXISTS}) \forall t' > t. o \in (\text{Exists-C}^{\mathcal{I}(t')} \vee \text{Disabled-C}^{\mathcal{I}(t')})$$

once it is disabled, by (DISAB1, see Appendix A), the disabled state persists.  $\square$

**Proposition 3.2 (Status Attributes: Logical Implications).** *Given the set of axioms  $\Sigma_{sa}$  and an attribute,  $A \sqsubseteq \text{From} : C \sqcap \text{To} : D$ , with  $D$  a data type, the following logical implications hold:*

*Proof.*

(AEXIST4) *Attribute Existence persists until Disabled.*

$$\Sigma_{sa} \models \text{Exists-A} \sqsubseteq \text{Exists-A} \sqcup \text{Disabled-A}$$

Let  $a \in \text{Exists-A}^{\mathcal{I}(t)}$  and by

(AEXIST1)  $\forall t' > t. a \in (Exists-A^{I(t')} \vee Disabled-A^{I(t')})$

once it is disabled, by (ADISAB1), the disabled state persists.

(ADISAB5) Disabled will never become active again.  $a \in Disabled-A^{I(t)} \rightarrow \exists t' > t. a \notin A^{I(t')}$

$Disabled-A \sqsubseteq \Box^+ \neg A$

(CSCH) *Scheduled class has scheduled attributes only.*

$\Sigma_{sa} \models Scheduled-C \sqsubseteq \forall[From]Scheduled-A$

$o \in Scheduled-C^{I(t)} \rightarrow \langle o, d \rangle \in Scheduled-A^{I(t)}$

(CSCH). Let  $o \in Scheduled-C^{I(t)}$ , then if  $\langle o, d \rangle \in A^{I(t)}$ , then  $o \in C^{I(t)}$  by AACT1, which contradicts the premise; if  $\langle o, d \rangle \in Suspended-A^{I(t)}$ , then  $o \in C^{I(t)}$  or  $o \in Suspended-C^{I(t)}$  (by ASUSP2), which also contradicts; if  $\langle o, d \rangle \in Disabled-A^{I(t)}$ , it contradicts likewise due to ADISAB3; therefore  $\langle o, d \rangle \in Scheduled-A^{I(t)}$  (which does not contradict ASCH3).

(ASCH3) *Scheduled attribute belongs to an active or scheduled class.*

$\Sigma_{sa} \models Scheduled-A \sqsubseteq From : (Scheduled-C \sqcup C)$

$\langle o, d \rangle \in Scheduled-A^{I(t)} \rightarrow o \in (Scheduled-C^{I(t)} \vee C^{I(t)}).$

(ASCH3). Let  $\langle o, d \rangle \in Scheduled-A^{I(t)}$ , then  $o \notin Disabled-C^{I(t)}$  because by AEXISTS3,  $o \in Exists-C^{I(t)}$ , and  $Exists-C^{I(t)}$  is disjoint from  $Disabled-C^{I(t)}$ ; if  $o \in Suspended-C^{I(t)}$  it contradicts CSUSP3, for it would force  $\langle o, d \rangle \in Suspended-A^{I(t)}$ , which contradicts the premise; hence  $\langle o, d \rangle \in Scheduled-A^{I(t)} \rightarrow o \in (Scheduled-C^{I(t)} \vee C^{I(t)}).$

(CDISAB4) *Disabled class has only disabled attributes.*

$o \in Scheduled-C^{I(t)} \rightarrow \langle o, d \rangle \in Scheduled-A^{I(t)}$

$\Sigma_{sa} \models Disabled-C \sqsubseteq \forall[From]Disabled-A$

(CDISAB4). Let  $o \in Disabled-C^{I(t)}$ , then if  $\langle o, d \rangle \in A^{I(t)}$ ,  $o$ 's status contradicts because of AACT1; if  $\langle o, d \rangle \in Suspended-A^{I(t)}$ , it leads to a contradiction because of ASUSP2; if  $\langle o, d \rangle \in Scheduled-A^{I(t)}$ , likewise a contradiction because of ASCH3; therefore  $\langle o, d \rangle \in Disabled-A^{I(t)}$  (which does not contradict ADISAB3).

(ASCH4) *Scheduled attribute persists until active.*

$\Sigma_{sa} \models Scheduled-A \sqsubseteq Scheduled-A \cup A$

The proof for ASCH4 is similar to the proof of SCH3 in [13]

Let  $a \in Scheduled-A^{I(t_0)}$ , then  $a \in Exists-A^{I(t_0)}$  and, by (ASCH1),  $\exists t_1 > t_0. a \in A^{I(t_1)}$ . Let's assume that  $t_1 = \min\{t \in \mathcal{T} \mid t > t_0 \text{ and } a \in A^{I(t)}\}$ . Now, by (AEXISTS1),  $\forall t'. t_0 < t' < t_1. a \in (Exists-A \sqcup Disabled-A)^{I(t')}$ . On the other hand, by (ADISAB5),  $a \notin Disabled-A^{I(t')}$ . By the 'min' choice of  $t_1$ ,  $a \in A^{I(t')}$  and also  $a \in Suspended-A^{I(t')}$ . Thus,  $\forall t'. t_0 < t' < t_1. a \in Scheduled-A^{I(t')}$ . Together with axiom (ASCH2), we can also conclude that  $Scheduled-A^{I(t')}$  is true just on a single interval.

(ASCH5) *Scheduled attribute cannot evolve directly to disabled.*

$\Sigma_{sa} \models Scheduled-A \sqsubseteq \oplus \neg Disabled-A$

The proof for ASCH5 is similar to SCH4

Let  $a \in \text{Scheduled-A}(t_0)$ , then by (ASCH1),  $\exists t_1 > t_0. a \in C^{\mathcal{I}(t_1)}$ . Thus by, (ADISAB5),  $a \notin \text{Disabled-A}^{\mathcal{I}(t_0+1)}$

(AACT2) *Active attribute will possibly evolve into suspended or disabled.*

$$\Sigma_{sa} \models A \sqsubseteq \Box^+(A \sqcup \text{Suspended-A} \sqcup \text{Disabled-A})$$

(AACT2). An attribute remaining active, becoming suspended, or disabled is the same as saying it cannot become scheduled anymore. By ASCH2, if  $a \in A^{\mathcal{I}(t)}$  then  $\forall t' > t. a \notin \text{Scheduled-A}^{\mathcal{I}(t')}$ , therefore  $\forall t' > t. a \in (A^{\mathcal{I}(t')} \vee \text{Suspended-A}^{\mathcal{I}(t')} \vee \text{Disabled-A}^{\mathcal{I}(t')})$ .

(ASUSP3) *Suspended attributes can never be followed by scheduled or disabled.*

$$\Sigma_{sa} \models \text{Suspended-A} \sqsubseteq \oplus(\neg \text{Scheduled-A} \sqcap \neg \text{Disabled-A})$$

(ASUSP3). Let  $a \in \text{Suspended-A}^{\mathcal{I}(t)}$ , then by ASUSP1,  $\exists t' < t. a \in A^{\mathcal{I}(t')}$ , so  $a$  was active in the past; by ASCH1  $\exists t'' > t'. a \in A^{\mathcal{I}(t'')}$  (with  $a \in \text{Scheduled-A}^{\mathcal{I}(t'')}$ ), but there cannot be a  $t'''$  such that  $t'' < t''' < t'$  due to ASCH4, i.e.,  $a$  must first become active before any possible migration to another status, and by ADISAB1 disabled persists, so  $a \notin \text{Disabled-A}^{\mathcal{I}(t+1)}$ .

(ADISAB4) *Disabled attribute cannot belong to a scheduled or suspended class.*

$$\Sigma_{sa} \models \text{Disabled-A} \sqsubseteq \neg(\text{From : Scheduled-C} \sqcup \text{From : Suspended-C})$$

(ADISAB4). Let  $\langle o, d \rangle \in \text{Disabled-A}^{\mathcal{I}(t)}$ , then if  $o \in \text{Scheduled-C}^{\mathcal{I}(t)}$ , it contradicts ADISAB3 as it says that either  $o \in \text{Disabled-C}^{\mathcal{I}(t)}$  or  $o \in C^{\mathcal{I}(t)}$ , thus also if  $o \in \text{Suspended-C}^{\mathcal{I}(t)}$ , it contradicts ADISAB3 too, therefore,  $a \notin \text{Scheduled-A}^{\mathcal{I}(t)}$  and  $a \notin \text{Suspended-A}^{\mathcal{I}(t)}$ .

□

The analogues of the logical implications for attributes to those for temporal relations are that ASCH4 corresponds to RSCH3 and ASCH5 to RSCH4 in [7]. The new ones specific to attributes not represented in either objects[13] or relations [51] are: CSCH, CDISAB4, ASCH3, AACT2, ASUSP3, and ADISAB4.

### 3.3 Inheritance and Temporal Attributes

Given that inheritance for temporalised classes was proven in [13] with five ISA implications for temporalised class subsumption with respect to status classes proven.

(ISA1) *Objects active in B must be active in A*, i.e.  $B \sqsubseteq A$ ,

(ISA2) *Objects suspended in B must be either suspended or active in A*,  
i.e.,  $\text{Suspended-B} \sqsubseteq (\text{Suspended-A} \sqcup A)$ ,

(ISA3) *Objects disabled in B must be either disabled, suspended or active in A*,  
i.e.,  $\text{Disabled-B} \sqsubseteq (\text{Disabled-A} \sqcup A \sqcup \text{Suspended-A})$ ,

(ISA4) *Objects scheduled in B must exist in A*,  
i.e.,  $\text{Scheduled-B} \sqsubseteq \text{Exists-A}$ ,



(ISA5) *Objects disabled in A, and active in B in the past, must be disabled in B,*

$$\text{i.e., Disabled-A} \sqcap \Diamond \neg B \sqsubseteq \text{Disabled-B.}$$

We also wanted to find out inheritance of temporal attributes in temporal classes, but for attributes, one cannot simply replace ‘object’ with ‘attribute of object’, because the superclass may not have that attribute. Moreover, the representation of attribute hierarchies is uncommon, but attribute inheritance is important, and therewith the interaction between the permissible statuses in the temporal class subsumption with the temporal attributes. Such consequences for subsumption and their interaction with classes in a hierarchy are not specified for status relations [7, 51]. To prove the logical implications for subsumption with respect to temporal attributes, first recall the very notion of subsumption. Let C and D be classes, and  $D \sqsubseteq C$ , which means that all instances of D are also instances of C. This is achieved in a logical theory iff D has either one of the following:

1. D has same set or superset of attributes of C
2. D has same set or superset of relations of C
3. Attribute and Relations active in C must also be active in D, otherwise it would deduce  $C \sqsubseteq D$
4. If D does not have an attribute or relationship, neither does C

Class hierarchies are relevant for conceptual modelling, and even more so for ontologies. We obtain the 8 logical implications as included in Proposition 3.3. Attributes of a class—e.g., where  $a \in A^{\mathcal{I}(t)}$  and  $a$  denotes the tuple  $\langle o, d \rangle \in A^{\mathcal{I}(t)}$  and  $o \in C^{\mathcal{I}(t)}$ —are written in shorthand notation, like  $A_C^{\mathcal{I}(t)}$  or in DL notation  $A_C$ , and to prevent wieldy subscripts, we use the following abbreviations: Active C, Scheduled Sch-C, Suspended Sus-C and Disabled Dis-C; so, e.g.,  $a \in \text{Disabled-A}_{\text{Dis-C}}^{\mathcal{I}(t)}$  represents a disabled attribute in a disabled class C.

**Proposition 3.3 (Status Attributes and Class ISA: Logical Implications).** *Given the set of axioms  $\Sigma_{sa}$ , attribute A and  $D \sqsubseteq C$ , where,  $D \sqsubseteq [\text{From}]A_D$  and, if present,  $C \sqsubseteq [\text{From}]A_C$ , at time  $t$ ,  $o$  is the object and  $a = \langle o, d \rangle$  is the attribute of the object in the class, the following logical implications hold:*

(ISA1<sub>A</sub>) *Attributes of objects disabled in C, and active in D in the past, are disabled in D.*

$$\text{Disabled-A}_{\text{Dis-C}} \sqcap \Diamond \neg A_D \sqsubseteq \text{Disabled-A}_{\text{Dis-D}} \sqcup \neg \text{Top-A}$$

(ISA2<sub>A</sub>) *Attributes active in D must be either active in C or is not present in C.*

$$A_D \sqsubseteq A_C \sqcup \neg \text{Top-A}$$

(ISA3<sub>A</sub>) *Attributes suspended in D must be either suspended in C, or is not present in C.*

$$\text{Suspended-A}_D \sqsubseteq \text{Suspended-A}_C \sqcup \neg \text{Top-A}$$

(ISA4<sub>A</sub>) *Attributes of objects suspended in D must be either suspended in C, or is not present in C.  $\text{Top-A}_{\text{Sus-D}} \sqsubseteq \text{Suspended-A}_{\text{Sus-C}} \sqcup \neg \text{Top-A}$*

(ISA5<sub>A</sub>) Attributes disabled in  $D$  must be either disabled in  $C$ , or is not present in  $C$ .

$$\text{Disabled-}A_D \sqsubseteq \text{Disabled-}A_C \sqcup \neg\text{Top-}A$$

(ISA6<sub>A</sub>) Attributes of objects disabled in  $D$  must be either disabled in  $C$  or is not present

$$\text{in } C. \text{Top-}A_{\text{Dis-}D} \sqsubseteq \text{Disabled-}A_{\text{Dis-}C} \sqcup \neg\text{Top-}A$$

(ISA7<sub>A</sub>) Attributes scheduled in  $D$  must be either scheduled in  $C$  or is not present in  $C$ .

$$\text{Scheduled-}A_D \sqsubseteq \text{Scheduled-}A_C \sqcup \neg\text{Top-}A$$

(ISA8<sub>A</sub>) Attributes of objects scheduled in  $D$  must be either scheduled in  $C$ , or is not present

$$\text{in } C. \text{Top-}A_{\text{Sch-}D} \sqsubseteq \text{Scheduled-}A_{\text{Sch-}C} \sqcup \neg\text{Top-}A$$

**Proof**

(ISA1<sub>A</sub>) Attributes of objects disabled in  $C$ , and active in  $D$  in the past, are disabled in  $D$ .

*Proof.* Given:  $o \in \text{Disabled-}C^{\mathcal{I}(t_0)}$ , by CDISAB4, then  $a \in \text{Disabled-}A_{\text{Dis-}C}^{\mathcal{I}(t_0)}$  and  $a \in A_D^{\mathcal{I}(t_1)}$ , where  $t_1 < t_0$ . By Subsumption premise,  $A$  is also an attribute of  $D$  and by AACT1, for any  $a \in A^{\mathcal{I}(t_1)}$ , then  $o \in D^{\mathcal{I}(t_1)}$ , i.e.  $\Diamond^-D$ . By ISA5,  $\text{Disabled-}C \sqcap \Diamond^-D \sqsubseteq \text{Disabled-}D$ ; thus, because  $o \in \text{Disabled-}C^{\mathcal{I}(t_0)}$ , then also  $o \in \text{Disabled-}D^{\mathcal{I}(t_0)}$ . Therefore by CDISAB4,  $a \in \text{Disabled-}A_{\text{Dis-}D}^{\mathcal{I}(t_0)}$ .

(ISA2<sub>A</sub>) Attributes active in  $D$  must be either active in  $C$  or is not present in  $C$ .

Given:  $o \in D^{\mathcal{I}(t)}$ , and its active attribute,  $a \in A_D^{\mathcal{I}(t)}$ .

**Part 1. part 1.** From ISA1, we get  $D \sqsubseteq C$ , and thus by CACTIVE,

$C \sqsubseteq \text{From} : (\text{Exists-}A \sqcup \text{Disabled-}A)$ , and thus

$(\text{Scheduled-}A \sqcup A \sqcup \text{Suspended-}A) \text{ or } \text{Disabled-}A$  (AEXISTS2).

**Part 2. part 2.** We prove by contradiction that if  $a \in A_D^{\mathcal{I}(t)}$ , then  $a \in A_C^{\mathcal{I}(t)}$ . If attributes in  $C$  are:

- $a \in \text{Disabled-}A_C^{\mathcal{I}(t)}$ , it contradicts: by ADISAB5,  $a \notin A^{\mathcal{I}(t')}$  where  $t' > t$ , together with ISA1<sub>A</sub>, if  $a \in \text{Disabled-}A_C^{\mathcal{I}(t)}$ , forces  $a \in \text{Disabled-}A_D^{\mathcal{I}(t)}$ , therefore  $a \notin \text{Disabled-}A_D^{\mathcal{I}(t)}$ .
- $a \in \text{Scheduled-}A_C^{\mathcal{I}(t)}$ , it contradicts: by ASCH1,  $a \in A^{\mathcal{I}(t')}$ , where  $t' > t$ , but we have  $a \in A_D^{\mathcal{I}(t)}$ , which means that it must have been active in  $C$ , but by ASCH2,  $a \notin A^{\mathcal{I}(t')}$ , therefore  $a \notin \text{Scheduled-}A_C^{\mathcal{I}(t)}$ .
- $a \in \text{Suspended-}A_C^{\mathcal{I}(t)}$ , contradicts:  $a \in A_C^{\mathcal{I}(t')}$ , where  $t' > t$ , due to ASUSP1, and the subsumption premise would deduce  $A_C \sqsubseteq A_D$ .
- $a \in A_C^{\mathcal{I}(t)}$  does not contradict, by AACT1,  $A \sqsubseteq \text{From} : C$ , since  $D \sqsubseteq C$  by ISA1.

Therefore, if  $a \in A_D^{\mathcal{I}(t)}$ , then  $a \in A_C^{\mathcal{I}(t)}$ , or it is not present in  $C$ .

(ISA3<sub>A</sub>) Attributes suspended in  $D$  must be either suspended in  $C$ , or is not present in  $C$ .

Given:  $o \in D^{\mathcal{I}(t)}$  with an attribute that is suspended:  $a \in \text{Suspended-}A_D^{\mathcal{I}(t)}$ , then, if the attribute is present also in  $C$ , it is existing or disabled (refer to ISA2<sub>A</sub>, part 1). We prove that if  $a \in \text{Suspended-}A_D^{\mathcal{I}(t)}$ , then  $a \in \text{Suspended-}A_C^{\mathcal{I}(t)}$ . Suppose the attribute in  $C$  is:

- $a \in Disabled-A_C^{\mathcal{I}(t)}$ . By ADISAB1,  $a \in Disabled-A^{\mathcal{I}(t')}$ , where  $t' > t$ , then from ISA1<sub>A</sub>, if  $a \in Disabled-A_C^{\mathcal{I}(t)}$ , it forces  $a \in Disabled-A_D^{\mathcal{I}(t)}$ , but by premise,  $a \in Suspended-A_D^{\mathcal{I}(t)}$ , therefore  $a \notin Disabled-A_C^{\mathcal{I}(t)}$ .
- $a \in Scheduled-A_C^{\mathcal{I}(t)}$ . By ASCH1,  $a \in A^{\mathcal{I}(t')}$ , where  $t' > t$ , but we have  $a \in A_D^{\mathcal{I}(t)}$ , which means that it must have been active in  $C$ . But by ASCH2,  $a \notin A^{\mathcal{I}(t')}$  therefore  $a \notin Scheduled-A_C^{\mathcal{I}(t)}$ .
- $a \in A^{\mathcal{I}(t)}$ . By ASUSP1,  $\exists t' < t. a \in A^{\mathcal{I}(t')}$ , but by subsumption premise if  $a \in Suspended-A_D^{\mathcal{I}(t)}$  and  $a \in A_C^{\mathcal{I}(t)}$ , then  $C$  would have more constrained attributes than  $D$ , forcing the deduction  $C \sqsubseteq D$ , which contradicts the premise. Therefore  $a \notin A^{\mathcal{I}(t)}$ .
- $a \in Suspended-A^{\mathcal{I}(t)}$  does not lead to a contradiction: by ASUSP2,  
 $Suspended-A \sqsubseteq From : (Suspended-C \sqcup C)$ .

Therefore, if  $a \in Suspended-A_D^{\mathcal{I}(t)}$ , it must be  $a \in Suspended-A_C^{\mathcal{I}(t)}$  or it does not exist in  $C$ , or: if the attribute exists in  $C$ , then  $Sus-A_D \sqsubseteq Sus-A_C$ .

(ISA4<sub>A</sub>) Attributes of objects suspended in  $D$  must be either suspended in  $C$ , or is not present in  $C$ .

Given:  $o \in Suspended-D^{\mathcal{I}(t)}$ , then from CSUSP3 we get  $a \in Suspended-A_{Susp-D}^{\mathcal{I}(t)}$  and from ISA2,  $o \in Suspended-C^{\mathcal{I}(t)}$  or  $o \in C^{\mathcal{I}(t)}$ . We prove by contradiction that if  $a \in Suspended-A_{Susp-D}^{\mathcal{I}(t)}$ , then  $a \in Suspended-A_{Susp-C}^{\mathcal{I}(t)}$ .

- CSUSP3, i.e.,  $Suspended-C \sqsubseteq From : Suspended-A$ , forces  $a \in Suspended-A_{Susp-C}^{\mathcal{I}(t)}$ .
- If  $o \in C^{\mathcal{I}(t)}$ , then by ISA3<sub>A</sub>,  $a \in Suspended-A^{\mathcal{I}(t)}$  if the attribute is exists in  $D$ .

Therefore attributes in  $C$  can only be suspended else they is not present in  $C$ .

(ISA5<sub>A</sub>) Attributes disabled in  $D$  must be either disabled in  $C$ , or is not present in  $C$ .

Given:  $o \in D^{\mathcal{I}(t)}$  and its disabled attribute  $a \in Disabled-A_D^{\mathcal{I}(t)}$ , then  $C$  has existing or disabled attributes (refer to ISA2<sub>A</sub>, part 1). We prove by contradiction that if  $a \in Disabled-A_D^{\mathcal{I}(t)}$ , then  $a \in Disabled-A_C^{\mathcal{I}(t)}$ . Now suppose the attribute in  $C$  is:

- $a \in Scheduled-A^{\mathcal{I}(t)}$ . ASCH1 implies  $a \in A^{\mathcal{I}(t')}$ , where  $t < t'$ . But if  $a \in Disabled-A_D^{\mathcal{I}(t)}$  then  $a \in A_D^{\mathcal{I}(t_0)}$  with  $t_0 < t$  (from ADISAB2), i.e., it must have been active before, and therefore  $a \in A_C^{\mathcal{I}(t_0)}$  (thanks to ISA2<sub>A</sub>), but by ASCH2 this cannot happen, therefore  $a \notin Scheduled-A_C^{\mathcal{I}(t)}$ .
- $a \in Suspended-A^{\mathcal{I}(t)}$  or  $a \in A^{\mathcal{I}(t)}$ , contradicts because of the subsumption premise (if  $D$  does not have some attribute  $A$  or relationship  $R$ , then neither does  $C$ ).

Therefore, if  $a \in Disabled-A_D^{\mathcal{I}(t)}$ , then  $a \in Disabled-A_C^{\mathcal{I}(t)}$ , or it is not present in  $C$ .

(ISA6<sub>A</sub>) Attributes of objects disabled in  $D$  must be either disabled in  $C$  or is not present in  $C$ .

Given:  $o \in Disabled-D^{\mathcal{I}(t)}$  and its attributes are disabled by CDISAB4,  $a \in Disabled-A_{Dis-D}^{\mathcal{I}(t)}$ , From ISA3, we have  $(Disabled-C \sqcup C \sqcup Suspended-C)$ . We prove

by contradiction that if  $a \in Disabled-A_{Dis-D}^{\mathcal{I}(t)}$ , the attribute can only be  $a \in Disabled-A_{Dis-C}^{\mathcal{I}(t)}$

- We start with  $o \in Suspended-C^{\mathcal{I}(t)}$ , by CSUSP3, it must be that  $a \in Suspended-A^{\mathcal{I}(t)}$ , but this contradicts ISA5<sub>A</sub>, because if  $a \in Disabled-A_D^{\mathcal{I}(t)}$ , then  $a \in Disabled-A_C^{\mathcal{I}(t)}$ .
- Next,  $o \in C^{\mathcal{I}(t)}$ . By CACTIVE,  $C \sqsubseteq From : (Exists-A \sqcup Disabled-A)$ , then if  $a \in Disabled-A_{Dis-D}^{\mathcal{I}(t)}$ , the attribute in  $D$  can only be disabled by ISA5<sub>A</sub>, because if Exists-A then it violates the subsumption premise.
- Last,  $o \in Disabled-C^{\mathcal{I}(t)}$ . CDISAB4 forces  $a \in Disabled-A^{\mathcal{I}(t)}$ , which holds.

Therefore if  $a \in Disabled-A_{Dis-D}^{\mathcal{I}(t)}$ , then it must be  $a \in Disabled-A_{Dis-C}^{\mathcal{I}(t)}$  else, it does not exist.

(ISA7<sub>A</sub>) Attributes scheduled in  $D$  must be either scheduled in  $C$  or is not present in  $C$ .

Given:  $o \in D^{\mathcal{I}(t)}$  and its attribute scheduled,  $a \in Scheduled-A_D^{\mathcal{I}(t)}$ .  $C$  has existing or disabled attributes (see ISA2<sub>A</sub>, part 1). We prove by contradiction that if  $a \in Scheduled-A_D$ , we must have  $a \in Scheduled-A_C$ . Now suppose:

- $a \in A_C^{\mathcal{I}(t)}$ , then, by the subsumption premise,  $a \in A_D^{\mathcal{I}(t)}$ , but by ASCH2,  $a \in A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \notin Scheduled-A^{\mathcal{I}(t)}$ , hence, a contradiction;
- $a \in Suspended-A_C^{\mathcal{I}(t)}$ , then because of ASUSP1,  $\exists t' < t. a \in A^{\mathcal{I}(t')}$ , hence, the same argument as in the previous item applies.
- $a \in Disabled-A_C^{\mathcal{I}(t)}$ , which means there must be a time  $t' < t$ , s.t.  $a \in A^{\mathcal{I}(t')}$  (from ADISAB2) that contradicts (see first item);

Therefore, either  $a \in Scheduled-A_C^{\mathcal{I}(t)}$  or the attribute is not present in  $C$ .

(ISA8<sub>A</sub>) Attributes of objects scheduled in  $D$  must be either scheduled in  $C$ , or is not present in  $C$ .

Given:  $o \in Scheduled-D^{\mathcal{I}(t)}$ , therefore also  $a \in Scheduled-A_{Sched-D}^{\mathcal{I}(t)}$  (by CSCH) and  $Scheduled-D \sqsubseteq Exists-C$ , i.e.,  $Scheduled-D \sqsubseteq (Scheduled-C \sqcup C \sqcup Suspended-C)$  from ISA4. We prove that if  $a \in Scheduled-A_{Sched-D}^{\mathcal{I}(t)}$ , then  $a \in Scheduled-A_{Sched-C}^{\mathcal{I}(t)}$  must hold.

- If  $o \in Suspended-C^{\mathcal{I}(t)}$ , then by CSUSP3,  $a \in Suspended-A^{\mathcal{I}(t)}$ , which contradicts: by ASUSP1,  $\exists t' < t. a \in A^{\mathcal{I}(t')}$  but by ASCH2,  $a \in A^{\mathcal{I}(t)} \rightarrow \forall t' > t. a \notin Scheduled-A^{\mathcal{I}(t)}$ .
- If  $o \in C^{\mathcal{I}(t)}$ , then by ISA7<sub>A</sub>,  $a \in Scheduled-A^{\mathcal{I}(t)}$  if the attribute exists in  $C$ .
- If  $o \in Scheduled-C^{\mathcal{I}(t)}$ , by CSCH  $Scheduled-C \sqsubseteq [From]Scheduled-A$ , we have  $a \in Scheduled-A^{\mathcal{I}(t)}$ .

Therefore if  $a \in Scheduled-A_{Sched-D}^{\mathcal{I}(t)}$ , then  $a \in Scheduled-A_{Sched-C}^{\mathcal{I}(t)}$  else it is not present in  $C$ .  $\square$

This completes the logical implications with respect to subsumption in the context of status classes and status attributes. The interaction between subsumption for status classes that participate in status relations is expected to yield similar implications.

TABLE 3.1: Illustrations of logical implication of status attributes' interaction with status classes in the subsumption hierarchy, as they may be declared for some conceptual model for an application in an organisation.

Logical Implication	Examples of the intended behaviour it can enforce
(ISA1 <sub>A</sub> ) Attributes of objects disabled in C, and active in D in the past, are disabled in D	When the production of a product is discontinued, the whole object and its attributes are disabled therefore the same attributes in the subclasses are also disabled
(ISA2 <sub>A</sub> ) Attributes active in D must be either active in C or is not present in C	These are normal (temporal) attributes; e.g., salary, price of a product
(ISA3 <sub>A</sub> ) Attributes suspended in D must be either suspended in C, or is not present in C	Going on leave, the manager's access rights can be suspended, and also as employee the access rights are suspended (but not the object itself)
(ISA4 <sub>A</sub> ) Attributes of objects suspended in D must be either suspended in C, or is not present in C	A manager is target of a fraud case, so the object and its attributes are suspended, and as a result its attributes are suspended also as an employee
(ISA5 <sub>A</sub> ) Attributes disabled in D must be either disabled in C, or is not present in C	When a company that produces software applications makes one of its software applications open source, the price attribute is disabled therefore the attribute price in the superclass is also disabled
(ISA6 <sub>A</sub> ) Attributes of objects disabled in D must be either disabled in C or is not present in C	A manager leaves the company, hence, becomes a member of the Disabled-Manager class, and so will its attributes be disabled. Then also the objects' attributes for it's superclass Employee must be disabled.
(ISA7 <sub>A</sub> ) Attributes scheduled in D must be either scheduled in C or is not present in C	Attributes can be scheduled when an employee expects to get his usual salary after probation period, the attribute prob_salary is used during probation, but after probation, the attribute that was scheduled salary is now used
(ISA8 <sub>A</sub> ) Attributes of objects scheduled in D must be either scheduled in C, or is not present in C	A manager is scheduled to start work in a few weeks, the attribute salary will be scheduled in the employee class.

### 3.4 Transition Constraints

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 doesn't 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 and its  $\mathcal{DLR}_{US}$  formalisations. These transitions can be either an evolution or an extension, with 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 for classes were introduced in [13], while quantitative extension TEX or QEX, quantitative evolution TEV or QEV, persistent extension PEX, persistent evolution PEV were introduced in [10]. The parallel transition for relations are relation dynamic 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 introduced in [51]. However, there is no support for attribute migration or on the effect of object migration on attributes, yet every object transition affects attributes.

Attribute migration is more complex than object migration because it is bilateral, 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 permit. Representation of attribute hierarchies is uncommon, mainly because it is not properly defined and formalised, but is relevant for modelling temporal. With the results on attribute hierarchy with subsumption in Proposition 3.3 Status Attributes and Class ISA, allow us to capture the interaction between the permissible statuses of classes and status attributes in temporal transitions. Proper representation of transition in the conceptual model will enable a modeller to know how to design a temporal database. This work relies on previous work on transitions of objects and relations for the  $\mathcal{ER}_{VT}$  model. Building up on the prior work on object migration, we represent attribute migration as well as correcting a few errors on transition of classes, that alluded that suspended classes can participate in a transition and we give a thorough definition of quantitative evolution, (instead on merely stating the next time point), in Section 3.4.1. We represent attribute migration as evolution constraints ADEV and ADEX, persistence constraints APEX and APEV and quantitative constraints AQEV and AQEX.

To properly address transition of attributes, a few corrections on transition of classes are made. First, formalisation of evolution constraints DEX and DEV asserted that both suspended classes and active classes can participate in a transition [13]. 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 other than active to participate in transitions, e.g., one should not permit an extension of an object from a Suspended-Employee to an Active-Manager if  $\text{Manager} \sqsubseteq \text{Employee}$ . The class must

first be active before any transition. Thus, one cannot have suspended classes participating in a transition.

Second, we represent the arbitrary quantitative evolution, that only specified quantitative constraint that can capture movement for only one chronon, the next point  $(t+1)$ . The essence here is to have a way to represent fixed time changes for example transition after 3 years, we provide a generalisation of the next time as  $(t+x)$ , where  $x \in \mathbb{Z}$ . 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.

Lastly, we add strong transition constraints for objects to complete the set of class transitions. We also specify here that the migration can only be an evolution and not an extension, because the object will never belong to the source class. For example a scheduled object will never belong to an active class after migration. Strong transition constraints state that in an evolution, the class can never belong to the source class after migration. These include Strong dynamic evolution SDEV.

To show the transition constraints, we will use the integrated picture of status classes and status attributes is shown in Fig. 3.4, the dashed lines showing transition of objects and attributes and thereafter, axiomatic constraints of an object SDEV. Every object transition yields a similar transition for the attributes, for example, a SDEV object transition will have a SADEV attribute transition.

### 3.4.1 Transition Constraints for Classes

Before we introduce attribute migration, we would like to add to add other transition for classes, i.e. strong dynamic evolution, persistent and quantitative, in addition to dynamic constraints that were added in [13]. Strong dynamic evolution SDEV, persistent PEX and PEV and quantitative constraints QEX and QEV were introduced in [51].

#### 1. SDEV

Strong dynamic evolution, when an object migrates from one class to another and it can never belong to the source class again, it migrates completely, for example, if a peripheral device dies, e.g. spark plugs in a car, it needs to be replaced.

(SDEV) *Strong Dynamic evolution*

$$\begin{aligned} \text{SDEV}_{C_1, C_2} &\sqsubseteq C_1 \sqcap \neg C_2 \sqcap \Box^+ (\neg C_1 \sqcap C_2) \\ o \in \text{SDEV}_{C_1, C_2}^{I(t)} &\rightarrow \exists t' > t. \{o \in C_1^{I(t)} \wedge o \notin C_2^{I(t)} \wedge o \notin C_1^{I(t')} \wedge o \in C_2^{I(t')}\} \end{aligned}$$

#### 2. QEX and QEV

They are quantitative extension and quantitative evolution for classes to specify the amount of time when a migration will occur. We have a new time constant

introduced called  $x$ . We introduced this time constant to cater for the time factor that is to be specified by the modeller. The values can vary from 1 to a value  $z \in \mathbb{Z}$  to signify the maximum number of chronons it is to go through before the quantitative transition occurs.

#### QEX

The object is set to migrate after sometime but will still be a member of the source class. For example the modeller can specify that an employee will be promoted after 3 years.

#### (QEX) Quantitative extension

$$\begin{aligned} \text{QEX}_{C_1, C_2} &\sqsubseteq C_1 \sqcap \neg C_2 \sqcap \oplus_n C_2 \\ o \in \text{QEX}_{C_1, C_2}^{\mathcal{I}(t)} &\rightarrow \exists(t+n) > t. \{o \in C_1^{\mathcal{I}(t)} \wedge o \notin C_2^{\mathcal{I}(t+n)} \wedge o \in C_2^{\mathcal{I}(t+n)}\}, \text{ where} \\ n \in \mathbb{Z} \text{ and } t+n \in \mathcal{T}_p \end{aligned}$$

#### QEV

The object is set to migrate after sometime and will cease being a member of the source class. For example one can be promoted to a middle level manager from a low level manager after 5 successive years of satisfactory service, e.g. a sales manager getting a promotion to middle level management, his access rights change from those of a sales manager to a branch manager.

#### (QEV) Quantitative evolution

$$\begin{aligned} \text{QEV}_{C_1, C_2} &\sqsubseteq C_1 \sqcap \neg C_2 \sqcap \oplus_n (\neg C_1 \sqcap C_2) \\ o \in \text{QEV}_{C_1, C_2}^{\mathcal{I}(t)} &\rightarrow \exists(t+x) > t. \{o \in C_1^{\mathcal{I}(t)} \wedge o \notin C_2^{\mathcal{I}(t)} \wedge o \notin C_1^{\mathcal{I}(t+n)} \wedge o \in C_2^{\mathcal{I}(t+n)}\}, \text{ where } n \in \mathbb{Z} \text{ and } t+n \in \mathcal{T}_p \end{aligned}$$

### 3. PEX and PEV

They specify the non changing state of an object after it has migrated from the source class to the target class. Once the migration has occurred, the object stays in that state. PEV and PEX are persistent evolution and persistent extension, for objects.

#### PEX

This occurs when the object migrates to a different class and it is still a member of the other class, for example an alumnus can decide to be a postgraduate, he retains the alumnus status.

#### (PEX) Persistent extension

$$\begin{aligned} \text{PEX}_{C_1, C_2} &\sqsubseteq C_1 \sqcap \neg C_2 \sqcap \Box^+ C_2 \\ o \in \text{PEX}_{C_1, C_2}^{\mathcal{I}(t)} &\rightarrow \forall t' > t. \{o \in C_1^{\mathcal{I}(t)} \wedge o \notin C_2^{\mathcal{I}(t)} \wedge o \in C_1^{\mathcal{I}(t')} \wedge o \in C_2^{\mathcal{I}(t')}\} \end{aligned}$$





“mapped\_to” attribute to a datatype, while UML [20] displays the attribute, datatype, and optionally permitted values inside the class in the diagram. We introduce attribute transition in two ways, as the attribute migrates and changes its values and during object migration.

Attribute migration can be a consequence of object migration or the value of an attribute can trigger object migration. The former is the more natural way, which we discuss in this subsection and give their axiomatisation. Our assumption is that all the attributes exist in the class and are active until they are disabled. To better explain attribute migration, we give examples alongside the transition. Below are the attribute transition constraints:

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 attribute dynamic extension ADEX, attribute dynamic evolution ADEV, and SADEV, strong attribute dynamic evolution.

#### ADEX

Attribute dynamic extension, occurs when the attributes are still part of the source attribute and they add on values to the target attribute, for example, the number of degrees you have can only increase over time.

(ADEX) *Dynamic extension of an attribute*

$$\begin{aligned} \text{ADEX}_{A_1, A_2} &\sqsubseteq A_1 \sqcap \neg A_2 \sqcap \Diamond^+ A_2 \\ a \in \text{ADEX}_{A_1, A_2}^{I(t)} &\rightarrow \exists t' > t. \{a \in A_1^{I(t)} \wedge a \notin A_2^{I(t)} \wedge a \in A_2^{I(t')}\} \end{aligned}$$

#### 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 value of the attribute changes from the previous one, for example whenever there is a salary increase, the attribute value changes.

(ADEV) *Dynamic evolution of an attribute*

$$\begin{aligned} \text{ADEV}_{A_1, A_2} &\sqsubseteq A_1 \sqcap \neg A_2 \sqcap \Diamond^+ (\neg A_1 \sqcap A_2) \\ a \in \text{ADEV}_{A_1, A_2}^{I(t)} &\rightarrow \exists t' > t. \{a \in A_1^{I(t)} \wedge a \notin A_2^{I(t)} \wedge a \notin A_1^{I(t')} \wedge a \in A_2^{I(t')}\} \end{aligned}$$

#### 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 once the boolean attribute dead is set to true, it can never be changed.

(SADEV) *Strong Dynamic evolution of an attribute*

$$\text{SADEV}_{A_1, A_2} \sqsubseteq A_1 \sqcap \neg A_2 \sqcap \Diamond^+ (A_2 \sqcap \Box^+ \neg A_1)$$

$$a \in \text{SADEV1}_{A_1, A_2}^{\mathcal{I}(t)} \rightarrow \exists t' > t. \{a \in A_1^{\mathcal{I}(t)} \wedge a \notin A_2^{\mathcal{I}(t)} \wedge a \in A_2^{\mathcal{I}(t')}\} \wedge \forall t' > t. \{a \notin A_1^{\mathcal{I}(t')}\}$$

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 time constant introduced for quantitative constraints, QEX and QEV for objects. There are two types of quantitative constraints, AQEX and AQEV.

#### 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*

$$\begin{aligned} \text{AQEX}_{A_1, A_2} &\sqsubseteq A_1 \sqcap \neg A_2 \sqcap \oplus_n A_2 \\ a \in \text{AQEX}_{A_1, A_2}^{\mathcal{I}(t)} &\rightarrow a \in A_1^{\mathcal{I}(t)} \wedge a \notin A_2^{\mathcal{I}(t)} \wedge a \in A_2^{\mathcal{I}(t+n)} \text{ where } n \in \mathbb{Z} \end{aligned}$$

#### AQEV

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

(AQEV) *Quantitative evolution of an attribute*

$$\begin{aligned} \text{AQEV}_{A_1, A_2} &\sqsubseteq A_1 \sqcap \neg A_2 \sqcap \oplus_n (\neg A_1 \sqcap A_2) \\ a \in \text{AQEV}_{A_1, A_2}^{\mathcal{I}(t)} &\rightarrow a \in A_1^{\mathcal{I}(t)} \wedge a \notin A_2^{\mathcal{I}(t)} \wedge a \notin A_1^{\mathcal{I}(t+n)} \wedge a \in A_2^{\mathcal{I}(t+n)} \text{ where } n \in \mathbb{Z} \end{aligned}$$

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. There are two types of attribute persistent constraints, APEX and APEV.

#### APEX

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

(APEX) *Persistent extension of an attribute*

$$\begin{aligned} \text{APEX}_{A_1, A_2} &\sqsubseteq A_1 \sqcap \neg A_2 \sqcap \square^+ A_2 \\ a \in \text{APEX}_{A_1, A_2}^{\mathcal{I}(t)} &\rightarrow \forall t' > t. \{a \in A_1^{\mathcal{I}(t)} \wedge a \notin A_2^{\mathcal{I}(t)} \wedge a \in A_2^{\mathcal{I}(t')}\} \end{aligned}$$

#### APEV

Persistent attribute evolution, occurs when the attribute migrates, and ceases

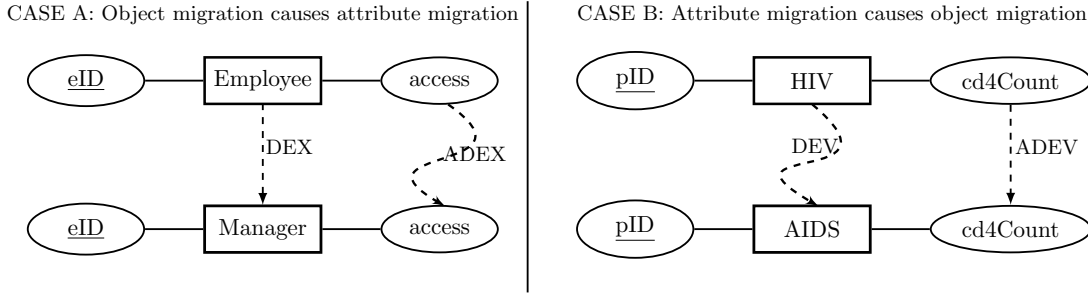


FIGURE 3.5:  $EER_{VT}^{++}$  diagram extended with migration constraints showing the interaction between objects and attributes. The dashed arrows illustrate class migration and attribute migration.

to be a member of the source attribute and its value remains constant from that point on.

(APEV) *Persistent evolution of an attribute*

$$\begin{aligned} \text{APEV}_{A_1, A_2} &\sqsubseteq A_1 \sqcap \neg A_2 \sqcap \square^+(\neg A_1 \sqcap A_2) \\ a \in \text{APEV}_{A_1, A_2}^{\mathcal{I}(t)} &\rightarrow \forall t' > t. \{a \in A_1^{\mathcal{I}(t)} \wedge a \notin A_2^{\mathcal{I}(t)} \wedge a \notin A_1^{\mathcal{I}(t')} \wedge a \in A_2^{\mathcal{I}(t')}\} \end{aligned}$$

### 3.5 Interaction between transition classes and attributes

Object migration affects only temporal classes, which may contain both temporal and non temporal attributes, but only 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  $\mathcal{DCR}_{US}$ , 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. 3.5, where dashed lines show examples of transitions on the integrated status classes and status attributes. Whenever an object migrates, the attribute undergoes the same kind of migration, for instance, when a class ceases to exist, undergoes a dynamic evolution DEV, to disabled, which triggers a parallel transition for its attributes, ADEV. For example when a student (member of a class Student) graduates and becomes an alumnus, 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.

*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. 3.5 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.

The subsequent logical implications for attribute transition constraints are as follows.

**Proposition 3.4 (Attribute migration).** *Given the set of axioms of attribute migration, the following logical implications hold as a consequence:*

1. *Attributes that have undergone persistent migration always have the same value until they are disabled. By Total Target Transition from [13], given*

$A_2 \text{ ISA } \Diamond^- \text{APEX}_{A_1, A_2}$ , *therefore,*

$A_2 \sqsubseteq A_2 \sqcup \text{Disabled-}A_2$

*Let  $a \in A_2^{\mathcal{I}(t_0)}$ , then  $a \in \text{Exists-}A_2^{\mathcal{I}(t_0)}$  and,*

*by  $(\text{APEX}_{A_1, A_2}) \forall t'. t. a \in A_2^{\mathcal{I}(t')}$*

*If we assume that  $t_1 = \min\{t \in \mathcal{T} \mid t > t_0 \text{ and } a \in A_2^{\mathcal{I}(t)}\}$*

*Now by (AEXISTS1)  $\forall t'. t_0 < t' < t_1. o \in (\text{Exists-}A \sqcup \text{Disabled-}A)^{\mathcal{I}(t)}$*

*and (ADISAB5)  $a \notin \text{Disabled-}A^{\mathcal{I}(t)}$*

*By 'min' choice of  $t_1$   $a \notin A_2^{\mathcal{I}(t')}$ . Thus  $\forall t'. t_0 < t' < t_1. \in A_1$*

*Together with APEX we conclude that  $A_2$  is true on just a single interval.*

$\text{APEV}_{A_1, A_2}$  and  $\text{SAPEV}_{A_1, A_2}$ , *can be proven as shown above*

2. *Only temporal attributes can participate in the migration*

*We know that only temporal classes are involved in transition, proven in [13] 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,  $\text{ADEV}_{A_1, A_2} \sqsubseteq \oplus \text{ADEV}_{A_1, A_2} \sqcap \ominus \text{ADEV}_{A_1, A_2}$ , which contradicts because temporal attributes hold only at single time points. Therefore, we can only have temporal attributes,  $\text{ADEV}_{A_1, A_2} \sqsubseteq \oplus \neg \text{ADEV}_{A_1, A_2} \sqcap \neg \ominus \text{ADEV}_{A_1, A_2}$ .*

3. *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 that can participate in the transition. Suppose that*

- $a \in \text{Disabled-}A^{\mathcal{I}(t)}$ , by (ADISAB1) disabled persists, disabled is an irreversible state, by (ADISAB4) Disabled will never be active. It can never be modified again, this contradicts, because in a transition the value changes.

- $a \in \text{Suspended-}A^{\mathcal{I}(t)}$ , 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.
- $a \in \text{Scheduled-}A^{\mathcal{I}(t)}$  By (ASCH1) it persists until active 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.

### 3.6 Summary

This chapter provides an avenue for discussion of several issues concerning temporal data on 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. A refined description logic language  $\mathcal{DLR}_{US}$  was used to properly include temporal constructors for attributes in its syntax and semantics. These constructors enabled the specification of a logic-based semantics for fully temporalised attributes, aided by the notion of status common in temporal conceptual data modelling. In addition, the logical implications for subsumption have been proven, in particular the interaction between status classes and status attributes in a subsumption hierarchy and transition thereby providing rules for effectively modelling and managing temporal data.  $\mathcal{DLR}_{US}$  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 works.

# 4

## $EER_{VT}^{++}$ AND $\mathcal{DLR}_{US}$

The previous chapter gave the formalisation of temporal attributes in  $\mathcal{DLR}_{US}$  and the  $\mathcal{ER}_{VT}$  modelling language using the extended  $\mathcal{DLR}_{US}$  including the definition of status, transition and their logical implications were presented in the previous chapter. In this chapter, the  $EER_{VT}^{++}$  which substantially extends  $\mathcal{ER}_{VT}$ , a very expressive temporal conceptual data modelling language, is defined. The definition of  $EER_{VT}^{++}$  conceptual data model, with its semantics and its mapping into  $\mathcal{DLR}_{US}$  is provided. Thereafter, the graphical representation of  $EER_{VT}^{++}$  and the difference between  $\mathcal{ER}_{VT}$  and  $EER_{VT}^{++}$ . Note that, as we shall see afterwards, it still can be mapped into  $\mathcal{DLR}_{US}$ .  $EER_{VT}^{++}$  supports timestamping, status and inheritance for classes, attributes and relationships.

### 4.1 Definition of $EER_{VT}^{++}$

#### 4.1.1 $EER_{VT}^{++}$ Conceptual data model

**Definition 4.1** ( $EER_{VT}^{++}$  Syntax). An  $EER_{VT}^{++}$  conceptual data model is a tuple:  $\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}_A, \text{CARD}_R, \text{ISA}_C, \text{ISA}_R, \text{ISA}_U, \text{DISJ}_C, \text{DISJ}_R, \text{COVER}, \text{S}, \text{T}, \text{ID}^1, \text{REX}, \text{RDM}, \mathcal{E})$ , such that:  $\mathcal{L}$  is a finite alphabet partitioned into the sets:  $\mathcal{C}$  (class symbols),  $\mathcal{A}$  (attribute symbols),  $\mathcal{R}$  (relationship symbols),  $\mathcal{U}$  (role symbols),  $\mathcal{D}$  (domain symbols), and  $\mathcal{F}$  (attribute role symbols) which symbolises FROM and TO, where the set  $\mathcal{C}$  of class symbols is partitioned into a set  $\mathcal{C}^S$  of *Snapshot classes* (marked with an S), a set  $\mathcal{C}^M$  of *Mixed classes* (unmarked classes), and a set  $\mathcal{C}^T$  of *Temporary classes* (marked with a T), (and likewise for relationships and attributes), and

- ATT is a function that maps a class symbol in  $\mathcal{C}$  to an  $\mathcal{A}$ -labeled tuple over  $\mathcal{D}$ ,  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle$ .
- REL is a function that maps a relationship symbol in  $\mathcal{R}$  to an  $\mathcal{U}$ -labeled tuple over  $\mathcal{C}$ ,  $\text{REL}(R) = \{U_1 : C_1, \dots, U_k : C_k\}$ ,  $k$  is the *arity* of  $R$ , and if  $(U_i : C_i) \in \text{REL}(R)$  then  $\text{PLAYER}(R, U_i) = C_i$  and  $\text{ROLE}(R, C_i) = \{U_i\}$ . The signature of the relationship is  $\sigma_R = \langle \mathcal{U}, \mathcal{C}, \text{PLAYER}, \text{ROLE} \rangle$ , where for all  $U_i \in \mathcal{U}$ ,  $C_i \in \mathcal{C}$ , if  $\sharp U \geq \sharp C$  then for

<sup>1</sup>Previously, it was written as KEY, we change this to ID to differentiate between the key in relational models and the term *identifier*, which is used for conceptual data models.

each  $u_i, c_i, \text{REL}(R)$ , we have  $\text{PLAYER}(R, U_i) = C_i$  and  $\text{ROLE}(R, C_i) = U_i$ , and if  $\sharp U > \sharp C$  then there is at least once  $\text{PLAYER}(R, U_i) = C_i$ ,  $\text{PLAYER}(R, U_{i+1}) = C_i$  and  $\text{ROLE}(R, C_i) = \{U_i, U_{i+1}\}$ .

- $\text{CARD}_R$  is a function  $\mathcal{C} \times \mathcal{R} \times \mathcal{U} \mapsto \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$  denoting cardinality constraints. We denote with  $\text{CMIN}(C, R, U)$  and  $\text{CMAX}(C, R, U)$  the first and second component of  $\text{CARD}_R$ .
- $\text{CARD}_A$  is a function  $\mathcal{C} \times \mathcal{A} \times \mathcal{F} \mapsto \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$  denoting multiplicity for attributes. We denote with  $\text{CMIN}(C, A, F)$  and  $\text{CMAX}(C, A, F)$  the first and second component of  $\text{CARD}_A$ , and  $\text{CARD}_A(C, A, F)$  may be defined only if  $(A, D)$  in  $\text{ATT}(C)$  for some  $D \in \mathcal{D}$ ;
- $\text{ISA}_C$  is a binary relationship  $\text{ISA} \subseteq (\mathcal{C} \times \mathcal{C})$ .
- $\text{ISA}_R$  is a binary relationship  $\text{ISA} \subseteq (\mathcal{R} \times \mathcal{R})$ .
- $\text{ISA}_U$  is a binary relationship  $\text{ISA} \subseteq (\mathcal{U} \times \mathcal{U})$ .  $\text{ISA}$  between roles of a relationships is restricted to relationships with the same signature, i.e., given an  $R_1 \subseteq R_2$  then  $\sigma_{R_1} = \sigma_{R_2}$ , and  $\text{ROLE}(R_1, C_i) \subseteq \text{ROLE}(R_2, C_i)$
- $\text{DISJ}_C, \text{COVER}_C$  are binary relations over  $(2^{\mathcal{C}} \times \mathcal{C})$ , describing disjointness and covering partitions, respectively, over a group of  $\text{ISA}$  that share the same superclass  $\text{DISJ}$
- $\text{DISJ}_R$ , are binary relations over  $(2^{\mathcal{R}} \times \mathcal{R})$ , describing disjointness and covering partitions, respectively, over a group of  $\text{ISA}$  that share the same super-relation.
- $\text{S}, \text{T}$  are binary relations over  $\mathcal{C} \times \mathcal{D}$  containing, respectively, the snapshot and temporary attributes of a class;
- $\text{ID}$  is a function,  $\text{ID} : \mathcal{C} \rightarrow \mathcal{A}$ , that maps a class symbol in  $\mathcal{C}$  to its key attribute and  $A \in \mathcal{A}$  is an attribute defined in  $\text{ATT}(C)$ , i.e.,  $\text{ID}(C)$  may be defined only if  $(A, D) \in \text{ATT}(C)$  for some  $D \in \mathcal{D}$ .
- $\text{REX}, \text{RDM}$  are binary relations over  $2^{\mathcal{U}} \times \mathcal{U}$ , describing disjointness partitions over a group of roles  $\mathcal{U}$  of relations in  $\mathcal{R}$  of the same arity to which  $C$  participates.
- $\mathcal{E}$  is the set of transition constraints, including  $\text{DEV}, \text{DEX}, \text{SDEV}, \text{QEX}, \text{QEV}$  (sometimes written as  $\text{TEX}$  and  $\text{TEV}$  resp.),  $\text{PEX}, \text{PEV}, \text{RDEX}, \text{RDEV}, \text{SRDEV}, \text{RQEX}, \text{RQEV}, \text{SRQEV}, \text{ADEX}, \text{ADEV}, \text{SADEV}, \text{AQEX}, \text{AQEV}, \text{SAQEV}, \text{APEX}, \text{APEV}$  and  $\text{SAPEV}$ .

Basic  $\mathcal{ER}_{VT}$  semantics have been described in [4, 8, 13] and we include here an updated version to cover also the additions for  $EER_{VT}^{++}$ , the main changes are

1. The set  $\mathcal{F}$  in the finite alphabet  $\mathcal{L}$  a function that symbolises the role From and To for an attribute.
2. Addition of the cardinality of the attribute  $\text{CARD}_A$ .
3. The binary relations  $\text{s}$  and  $\text{t}$  between classes and attributes had the binary relation over  $\mathcal{C} \times \mathcal{A}$ , which we now write as  $\mathcal{C} \times \mathcal{D}$ .



4. Additional of transition constraints for temporal attributes, in the set of transition constraints  $\mathcal{E}$ , e.g. ADEX, ADEV.

#### 4.1.2 $EER_{VT}^{++}$ Semantics

Just like  $\mathcal{ER}_{VT}$ , we can define for  $EER_{VT}^{++}$  the satisfiability of a class  $C$ , relationship  $R$ , and  $\Sigma$ , and class and relationship subsumption, and general logical implication; see Definition 2.2.  $EER_{VT}^{++}$  semantics changed the attribute set, now written as  $a$ , which is the object  $o$  and domain  $d$ .  $\mathcal{ER}_{VT}$  had  $a$ , as the mapping between an object  $o$  and an attribute  $a$ .

- Every attribute  $A$  to a set  $A^{\mathcal{I}(t)} \subseteq \Delta^{\mathcal{I}} \times \Delta_D^{\mathcal{I}}$ , such that, for each  $C \in \mathcal{C}$ , if  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle$ , then,  $o \in C^{\mathcal{I}(t)} \rightarrow (\forall i \in \{1, \dots, h\}, \exists d_i. a \in A_i^{\mathcal{I}(t)} \wedge \forall d_i. a \in A_i^{\mathcal{I}(t)} \rightarrow d_i \in \Delta_{D_i}^{\mathcal{I}})$ .

We briefly summarise how the extended  $\mathcal{DLR}_{US}$  captures temporal schemas expressed in  $EER_{VT}^{++}$ . Conceptual data models use DL as a base language to reason over the data, we found out that the language  $\mathcal{DLR}_{US}$  did not clearly represent temporal attributes.

#### 4.1.3 Mapping $EER_{VT}^{++}$ into the extended $\mathcal{DLR}_{US}$

**Definition 4.2** (Mapping  $EER_{VT}^{++}$  into  $\mathcal{DLR}_{US}$ ). Let  $\Sigma = (\mathcal{L}, \text{REL}, \text{ATT}, \text{CARD}_A, \text{CARD}_R, \text{ISAC}, \text{ISAR}, \text{ISAU}, \text{DISJC}, \text{DISJR}, \text{COVER}, \text{S}, \text{T}, \text{KEY}, \text{REX}, \text{RDM}, \mathcal{E})$  be an  $EER_{VT}^{++}$  schema. The  $\mathcal{DLR}_{US}$  knowledge base,  $\mathcal{K}$ , mapping  $\Sigma$  is as follows.

- For each  $A \in \mathcal{A}$ , then,  $A \sqsubseteq \text{From} : \top \sqcap \text{To} : \top \in \mathcal{K}$ ;
- If  $C_1 \text{ ISA } C_2 \in \Sigma$ , then,  $C_1 \sqsubseteq C_2 \in \mathcal{K}$ ;  
(Similarly for relationships and roles)
- If  $\text{REL}(R) = \langle U_1 : C_1, \dots, U_k : C_k \rangle \in \Sigma$ , then  $R \sqsubseteq U_1 : C_1 \sqcap \dots \sqcap U_k : C_k \in \mathcal{K}$ ;
- If  $\text{ATT}(C) = \langle A_1 : D_1, \dots, A_h : D_h \rangle \in \Sigma$ , then,  $C \sqsubseteq \exists[\text{From}]A_1 \sqcap \dots \sqcap \exists[\text{From}]A_h \sqcap \forall[\text{From}](A_1 \rightarrow \text{To} : D_1) \sqcap \dots \sqcap \forall[\text{From}](A_h \rightarrow \text{To} : D_h) \in \mathcal{K}$ ;
- If  $\text{CARD}_R(C, R, U) = (m, n) \in \Sigma$ , then,  $C \sqsubseteq \exists^{\geq m}[U]R \sqcap \exists^{\leq n}[U]R \in \mathcal{K}$ ;
- If  $\text{CARD}_A(C, A, F) = (m, n) \in \Sigma$ , then,  $C \sqsubseteq \exists^{\geq m}A.D \sqcap \exists^{\leq n}A.D \in \mathcal{K}$ ;
- If  $\{C_1, \dots, C_n\} \text{ DISJ } C \in \Sigma$ , then  $\mathcal{K}$  contains:  
 $C_1 \sqsubseteq C \sqcap \neg C_2 \sqcap \dots \sqcap \neg C_n$ ;  
 $C_2 \sqsubseteq C \sqcap \neg C_3 \sqcap \dots \sqcap \neg C_n$ ;  
 $\dots$   
 $C_n \sqsubseteq C$ ;  
(Similarly for relationships)
- If  $\{C_1, \dots, C_n\} \text{ COVER } C \in \Sigma$ , then  $\mathcal{K}$  contains:  
 $C_1 \sqsubseteq C$ ;  $\dots$ ;  $C_n \sqsubseteq C$ ;  $C \sqsubseteq C_1 \sqcup \dots \sqcup C_n$ ;  
(Similarly for relationships)

- If  $ID(C) = A \in \Sigma$ , then,  $\mathcal{K}$  contains:  
 $C \sqsubseteq \exists^{-1}[\text{From}] \Box^* A$ ;  
 $\top \sqsubseteq \exists^{\leq 1}[\text{To}](A \sqcap \text{From} : C)$ ;
- If  $C \in \mathcal{C}^S$ , then,  $C \sqsubseteq (\Box^* C) \in \mathcal{K}$  (similar for  $R \in \mathcal{R}^S$ );
- If  $C \in \mathcal{C}^T$ , then,  $C \sqsubseteq (\Diamond^* \neg C) \in \mathcal{K}$  (similar for  $R \in \mathcal{R}^T$ );
- If  $\langle C, D \rangle \in s$ , then,  $C \sqsubseteq \forall[\text{From}](A \rightarrow \Box^* A) \in \mathcal{K}$ ;
- If  $\langle C, D \rangle \in t$ , then,  $C \sqsubseteq \forall[\text{From}](A \rightarrow \Diamond^* \neg A) \in \mathcal{K}$ .
- If  $\{U_1, U_2, \dots, U_{i-1}\} \text{REX } U_i \in \Sigma$ , then  $\mathcal{K}$  contains:  
 $C \sqsubseteq (\exists^{\geq 1}[U_1]R_1 \sqcup \dots \sqcup \exists^{\geq 1}[U_i]R_i); [U_1]R_1 \sqsubseteq \neg[U_2]R_2 \sqcap \dots \sqcap \neg[U_{i-1}]R_{i-1}, [U_2]R_2 \sqsubseteq \neg[U_3]R_3 \sqcap \dots \sqcap \neg[U_{i-1}]R_{i-1}, \dots, [U_i]R_i$ ;
- If  $\{U_1, U_2, \dots, U_{i-1}\} \text{RDM } U_i \in \Sigma$ , then  $\mathcal{K}$  contains:  
 $C \sqsubseteq (\exists^{\geq 1}[U_1]R_1 \sqcup \dots \sqcup \exists^{\geq 1}[U_i]R_i); [U_1]R_1 \sqsubseteq \neg[U_2]R_2 \sqcap \dots \sqcap \neg[U_{i-1}]R_{i-1}, [U_2]R_2 \sqsubseteq \neg[U_3]R_3 \sqcap \dots \sqcap \neg[U_{i-1}]R_{i-1}, \dots, [U_i]R_i$ ;

#### 4.1.4 Graphical Representation of $EER_{VT}^{++}$

**Definition 4.3** (The graphical representation of the  $EER_{VT}^{++}$  model). Below is the graphical representation of the  $EER_{VT}^{++}$  model, with the additional semantics for temporal attributes.

We give an example with a graphical syntax included in Fig. 4.2

**Example 1.** A University wants to keep time changing data, so that they can track and maintain the progress of both its employees and its students. A person at the university can be employee or a student or both. If he is an employee, he is either employed as an academic staff or non academic staff, with academic staff as tenured or not. Students on the other hand may either be an undergraduate or graduate, with graduate students either teaching assistants or research assistants. All employees earn a monthly salary that is increased every 2 years. To get a promotion to a tenure position, employees must have worked in that area for at least 5 years, with a PhD degree. The transition from untenured to tenure is a dynamic evolution. We show how  $EER_{VT}^{++}$  captures temporal attributes which  $\mathcal{ER}_{VT}$  could not.  $EER_{VT}^{++}$  captures temporal attributes and their evolution from one class to the other. The following scenarios illustrates the evolution of data.

- When a student graduates, he automatically becomes an alumnus and remains in that state, we show that he goes thorough a persistent evolution, PEV.
- If an academic gets a promotion to tenured, what happens to his attributes? The person goes through a dynamic evolution, DEV, with some of his attributes also going through a similar evolution, for example, `contractEndDate`, undergoes, ADEV to become `RetireDate`.


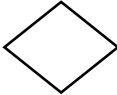

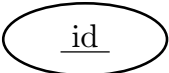
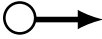

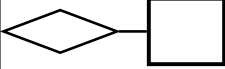
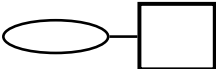
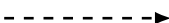

Symbol	Name	$EER_{VT}^{++}$ Syntax	Symbol	Name	$EER_{VT}^{++}$ Syntax
	Entity	$\mathcal{C}$		Relationship	$\mathcal{R}$
	Attribute	$\mathcal{A}$		Key Attribute	KEY
	ISA	$ISA_A$ $ISA_R$ $ISA_U$		ISA disjoint	$DISJA$ $DISJR$
	Relation link to class	REL $CARD_R$		Attribute link to class	ATT $CARD_A$
	Transition Constraint	$\mathcal{E}$		Completeness	COVER
S	Snapshot	S	T	Temporal	T
Transition constraint names					
Classes:	Dynamic: DEX, DEV, SDEV	Quantitative: QEX, QEV, SQEV	Persistent: PEX, PEV, SPEV		
Relations:	RDEX, RDEV, SRDEV	RQEX, RQEV, SRQEV	RPEX, RPEV, SRPEV		
Attributes:	ADEX, ADEV, SADEV	AQEX, AQEV, SAQEV	APEX, APEV, SAPEV		

FIGURE 4.1:  $EER_{VT}^{++}$  Graphical syntax.

- If an undergraduate graduates, and wants to become a graduate student? The student will go through dynamic evolution, DEV, from the undergraduate class to graduate class. The details of his history will be captured, his attributes will go through a similar transition, i.e. ADEV, for example from undergraduate access to postgraduate GrAccess, whereby now, he his access level increases.
- What happens when someone leaves the university? His details will now be captured in an easier way, by retaining his history when he was a student, now an alumnus. If he wants to return to graduate school later on, as an alumnus, he goes through dynamic extension, DEX, meaning that he is still a member of alumnus class and now is registered as a student.

Relating this to conceptual data model mapping, e.g., the subsumption in Fig. 4.2 can be represented in  $EER_{VT}^{++}$  textual syntax as Academic ISA Employee and its identifier as

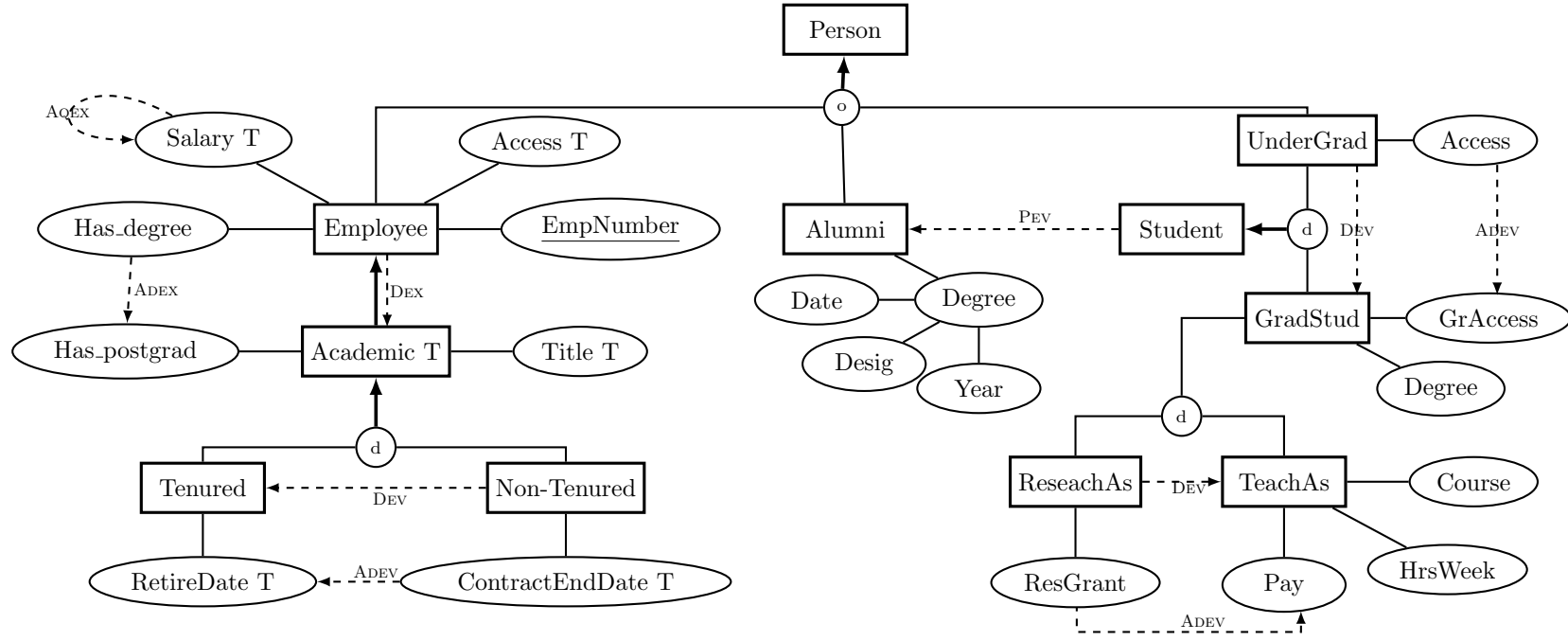


FIGURE 4.2:  $EER_{VT}^{++}$  diagram showing an application example with the interrelationship between temporal classes and temporal attributes as well as transition constraints

$ID(Employee) = EmpID$  and so forth, with its  $\mathcal{DLR}_{US}$  notation as  $Academic \sqsubseteq Employee$  and  $Employee \sqsubseteq \exists^{=1}[From] \sqcap *EmpID$  with  $\top \sqsubseteq \exists^{\leq 1}[To](EmpID \sqcap From : Employee)$ , respectively. Disjoint classes, tenured and nonTenured are written as  $Tenured \sqcap NonTenured = \perp$ . Let us show the mapping of  $EER_{VT}^{++}$  into  $\mathcal{DLR}_{US}$  notation for the undergraduate student who graduates and becomes an alumnus, then registers again as a graduate student.  $UG$  symbolises undergraduate student and  $AL$  for alumnus.

1. Undergraduate becomes an alumnus after graduation

$$PEV_{UG,AL} \sqsubseteq UG \sqcup \neg AL \sqcup \square^+(\neg UG \sqcap AL)$$

Some attributes also participate in the migration e.g. student Number, written as  $stdno$ , the alumnus will use his student number to access the alumnus databases

$$APEV_{stdno,oldStdno} \sqsubseteq stdno \sqcup \neg oldStdno \sqcup \square^+(\neg stdno \sqcap oldStdno)$$

2. An alumnus registers as a postgraduate student

$$DEV_{AL,PG} \sqsubseteq AL \sqcup \neg PG \sqcup \diamond^+(\neg AL \sqcap PG)$$

The attribute Degree will go through a similar evolution,  $ADEV$ .

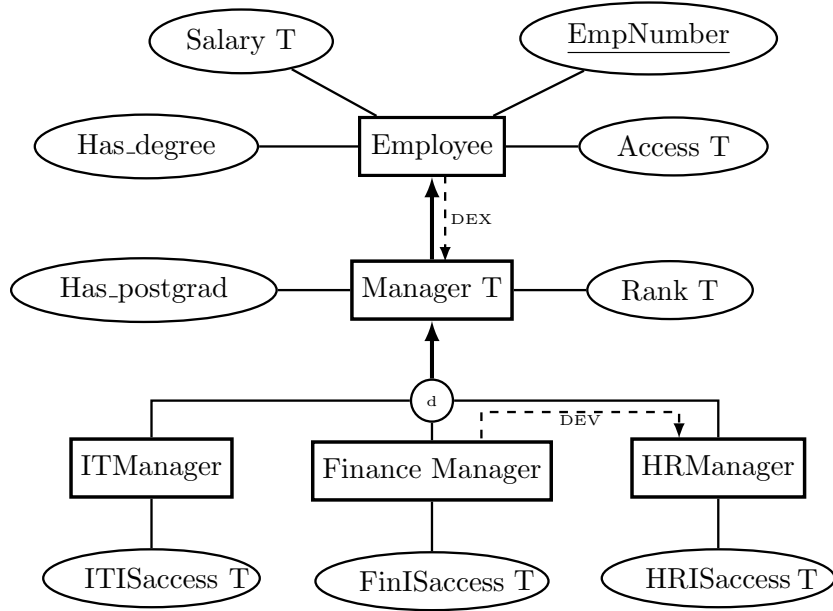
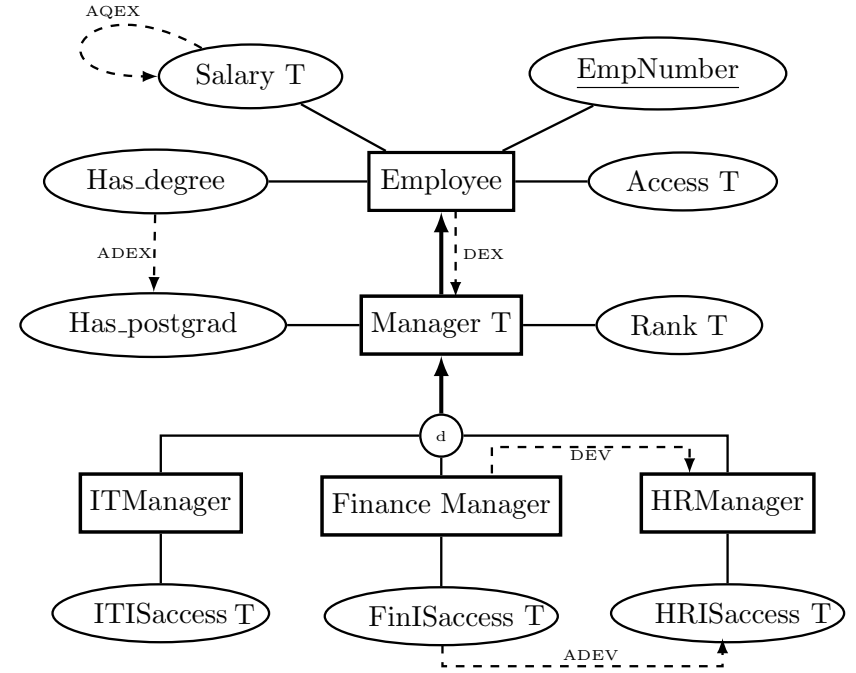
$$DEV_{deg,oldDeg} \sqsubseteq deg \sqcup \neg oldDeg \sqcup \diamond^+(\neg deg \sqcap oldDeg)$$

$$deg \sqsubseteq \diamond^+ oldDeg$$

## 4.2 Differences between $\mathcal{ER}_{VT}$ & $EER_{VT}^{++}$

The changes will be presented in two forms, in the language  $\mathcal{DLR}_{US}$  and the corrections to the language  $\mathcal{ER}_{VT}$  with the inclusion of temporal attributes. The changes in  $EER_{VT}^{++}$  are given below and a diagram showing the two models in Fig. 4.3

- Inclusion of temporal attributes in the formal definition of  $\mathcal{DLR}_{US}$  for a rigorous definition of temporal attributes.
- Inclusion of status attributes that help to classify temporal attribute lifecycle, as well as a proper formalisation of freezing of attributes.
- Transition of attributes inclusion to accurately capture evolution of any application domain.
- $\mathcal{ER}_{VT}$ 's formalisation of ATT in [13], had the binary relation  $\mathcal{C} \times \mathcal{A}$ , to state that a class which is not possible, because  $\mathcal{A}$  is the set of a binary relations from the set of classes  $\mathcal{C}$  to the domain symbols  $\mathcal{D}$ .
- Transition of attributes to accurately capture evolution of any application, plus having only active classes participating in object transitions.

$ER_{VT}$  Graphics $EER_{VT}^{++}$  GraphicsFIGURE 4.3:  $EER_{VT}^{++}$  and  $ER_{VT}$  graphical representation with the differences between the two

### 4.3 Summary

This chapter provided the definition of  $EER_{VT}^{++}$ , its semantics, as well as its mapping to  $\mathcal{DLR}_{US}$ . We also gave the graphical notation and an motivating example. Finally, we presented the difference between  $\mathcal{ER}_{VT}$  and  $EER_{VT}^{++}$ . Our graphical model will be accessed by a human computer interaction (HCI) researcher, who will look at the best graphical representation of temporal attributes in the temporal conceptual models.





# 5

## APPLICATIONS OF TEMPORAL ATTRIBUTES

Chapter 3 gave the formalisation of temporal attributes. In this chapter, we show the applicability of temporal attributes using real world problems in various fields and how  $EER_{VT}^{++}$  provides the solutions to these problems.

### 5.1 Application areas

Time is ingrained in every aspect of our lives, and ought to be dealt with and represented in information systems for easy information retrieval and to preserve history. Traditional databases cannot properly represent time changing data; they present only a snapshot of the database at the present time point and new data is added by simply replacing old attributes (values) with newer ones. Preserving records of evolution is important to model history and to plan for the future. Almost all applications domains require temporal data: and it is hard to find one that does not require it. As discussed in Chapter 3, temporal changes are captured in two ways, through: lifecycle and transition constraints. The results above draw us closer towards having a complete temporal model with formalised temporal classes, relations and attributes which can also reason over the data and spot inconsistencies.

Businesses use databases to store their data to keep track of their investments and ensure smooth running of the enterprise. In order for businesses to make maximum profit, they need to be very efficient and reduce cost-to-company which can be achieved by having a systematised way of representing constraints, reasoning over them and thus reduce errors in the database. By so doing companies will save time and money in cleaning up dirty or inconsistent data. This can be implemented by invoking business rules that permits change, and alerts the relevant authorities on the what to do and how they should react to the changes. The application areas may fall under the business process model defined in [67], as *"Supporting business processes using methods, techniques, and software to design, enact, control, and analyse operational processes involving humans, organisations, applications, documents and other sources of information"*.

There are several business process models that have been developed to try and solve problems, for example, security access, managing promotions. In the following section, some of the field of application will be examined closely and solutions will be proffered to problem areas which previous works in literature did not address. The application areas are in the following sections:

### 5.1.1 Administration

Administration is the process of running a business; the day-to-day management of a company. Companies need temporal attributes to store changes to ensure smooth running of the business which helps in making business decisions. A company may want to store the history of its employees, to track their progress and growth, HR policy makers need this data to have better analysis of employees, as a result, better decision support which allows them write better reports that can adapt to the changing business environment. For example, employee status may change as time passes, e.g. *salary increase, bonus payouts and promotions*, the HR manager can specify the conditions that permit the changes and what conditions can cause violations. For instance, consider a business rule that states that every employee has a **Salary** increase after 2 years or when an employee is on leave or facing fraud charges, some of the employee attributes (e.g., **Access**) are suspended. Consider also a CEO who evolves to a non-executive board member and then no longer earns a salary, but will receive a yearly bonus, one attribute (**Salary**), becomes ‘disabled’ and another (‘scheduled’) attribute (**hasBonus**) becomes active upon migration of the object. We give below an example of application of temporal attributes in administration.

**Example 2.** A company produces several similar products for example soap, the products have the following attributes, **productID**, **Product\_Name**, **Manufactured\_Date** and **Price**, is a temporal attribute because its value depends on events happening in the company at different times. The company may decide to have promotions for their products in a specific period in a particular location. These promotions may be in terms of **Coupons**, **Bonus\_product** or **Price\_reduction**, which will in turn produce additional temporal attributes for the products, e.g, **Promo\_price**, **Coupon\_number** and **added\_productID**, as shown in Fig. 5.1.

#### $EER_{VT}^{++}$ Solutions

1. If the sales of one product are lower than the others, the company may decide to continue with the promotion of the unpopular product and discontinue with the promotion of the popular product to increase demand of the unpopular product.

(a) What happens to the discontinued promotion?

At time  $t_0$ , the promotion for the products started, but at time  $t_1$ , the promotion for Product X have been disabled, the attributes Coupon is disabled, by ISA4<sub>A</sub>, which cause the disabling of the Product X coupons.

#### Disabled Class

```

prod(PrdID,date,coup) at t0
prod(PrdID,date,coup) at t1

prodZ(PrdID,att_Z) at t0
disabledProdZ(disabledPrdID,
disabled_attr_Z) at t1
prod(PrdID,date,coup) at t0
prod(PrdID,date,coup) at t1

prodX(PrdID,att_X) at t0
prodX(PrdID,attr_X,dis_coup)
at t1

```

Disabled Attributes by (CDIAB4)

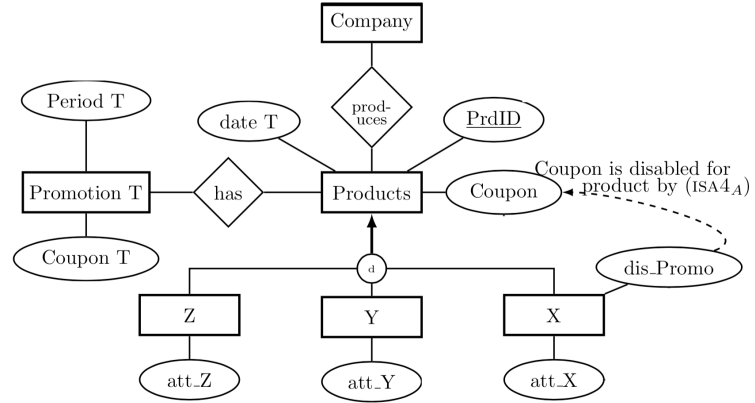


FIGURE 5.1:  $EER_{VT}^{++}$  model of promotion of a product in a company, with temporal attributes.

(b) What happens when the promotion expires?

When the promotion expires, thanks to the temporal attribute **Period** in the temporal class **Promotion** is disabled. Thereafter, all its attributes by CDISAB4 are disabled. This will cause the attribute **Coupon** in **Products** and by ISA1<sub>A</sub> to be disabled. As **Coupon** is disabled, all the subsequent attributes are disabled.

2. What if the company changes their company name, what happens?

They have to ensure that are the implication of that change to the users to ensure that this process runs smoothly and synchronously until the old name is now disabled as a brand name, (See Fig. 5.2).

- (a) Effect the new name to the database, by adding another class, say, NewBrand with the new attributes, i.e. new\_logo, new\_name.
- (b) Suspend the old class with all its attributes, by (CSUSP3) and create a trigger from the old link to the new link. This can be facilitated by setting a date for the transition, using QEV and AQEV.
- (c) Disable the old class, brand, which will disable its attributes by (CDISAB4).
- (d) The brand name has the following attributes, name, id, logo, while the new brand name is new\_name, product\_id, new\_logo may change their product brand and this should be effected in the database, showing the previous name and the new name.

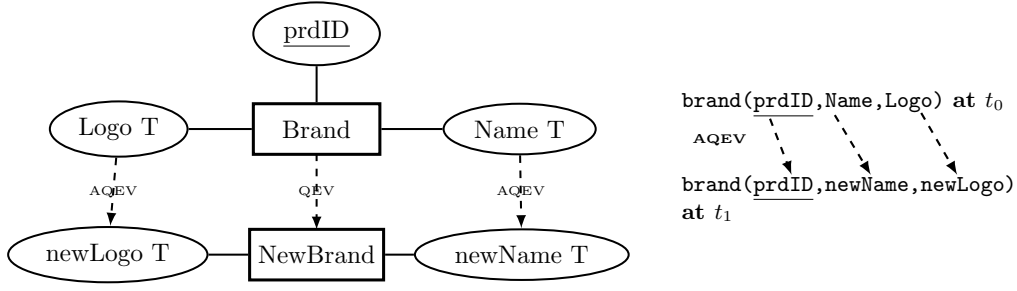


FIGURE 5.2: Diagram of the  $EER_{VT}^{++}$  showing the quantitative migration of temporal attributes due to the migration of a temporal class.

### 5.1.2 Information Security Systems

Procedures put in place to ensure that a system is free of danger or threat and that only the authorised users access the system. Temporal attributes can be used to control, monitor and authorise users as they log into systems. The attributes can be in the form of time-bound passwords; that users are given before they log in to a system which disallows reuse of previous passwords. Previously, users used role-based access to assign access to users according predefined roles, but with the increase of users over time, especially with the increase of internet usage and large databases with millions of users, security administrators can use attribute-based data access (ABAC) [44] to authenticate its users. ABAC controls access by evaluating the set rules against the attributes of entities, operation and environmental conditions relevant for the request [44]. Research in information security management systems are looking into specification of temporal attributes in ABAC especially for authentication on cloud servers, since data owners and data servers may very well be in different domains, which can be subject to insecurity. We give examples of how temporal attributes can permit a better facilitation to security.

**Example 3.** Suppose a company which has many retailers distributed across the country. The employees are given access according to their jobs, These attributes include **Period-of-Validity**, which is a time attribute on month basis; **Job**, which is a string attribute to denote employees position; **Day**, to denote working days (Example from [68]). An example of access is only engineers can view this document during regular working hours (say from Monday to Friday), managers can view it at other times. Another example is the sales record from 2011/01 to 2011/05, which can be accessed by accountants from Thursday to Friday.

#### $EER_{VT}^{++}$ Solutions

1. Suppose we want to record this on the database, we will have some days suspended to be activated after sometime. For example, someone working from Monday to Friday, will have the **access** attribute active only for those days and suspended on Saturday and Sunday.

### 5.1.3 Medical Information Systems

These are databases that store data for medical institutions, e.g. hospitals and pharmacies. If information is kept well in the database, doctors can better prescribe efficient drugs and faster diagnose the patients. Temporal attributes can be used to monitor how efficient a drug is, as long as the patient adheres to the schedule. Take an example of a patient who has been administered a drug, he gets immunity for a few years, until the immunity is weakened. If a patient gets sick, he is said to be healthy until he is infected. These databases use temporal attribute to store patient records that monitor the patient progress as well as the effect of drugs on patients. For example an HIV positive patient evolves to an AIDS patient after the attribute value of CD4 count drops below 180, or once the value for the boolean attribute transplant-received is set to 'yes', it cannot be changed anymore (it is so-called 'frozen'). A related application area is in pharmacovigilance [53], which aims to monitor adverse effects of drugs on patients. These results would be used by hospitals to determine if the drugs would be beneficial to its patients or harmful and by what quantity. They can also be used to support diagnosis, for example to monitor patient temperature and effectiveness of the treatment and when to react when the patient behaviour changes. Below we give a hospital database that wants to prevent the spread of infectious diseases.

**Example 4.** A hospital wants to record the patient records in a place that has an outbreak of an infectious disease, say, ebola. Patients admitted in the hospital are kept in isolation until they are healed or succumb to the illness. The patients attributes include, name, address, age, sex, is-suspected, date-confirmed, the temporal attributes include, state, which is the condition of the patient and is-dead is a boolean to say if he is alive or not.

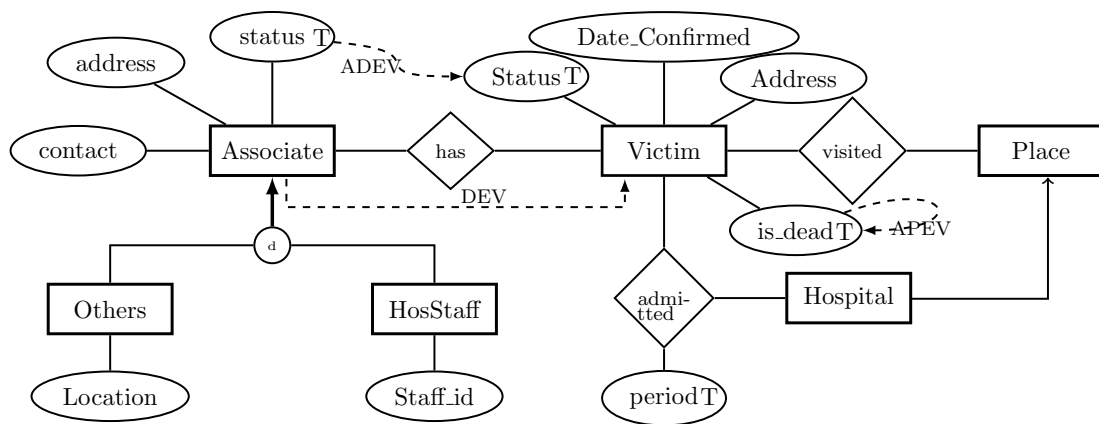


FIGURE 5.3:  $EER_{VT}^{++}$  diagram extended with migration constraints showing the example of hospital management system.

1. Once a person is suspected to have the disease, he is immediately admitted in hospital and his status recorded.
2. If he is confirmed is-confirmed to have the disease, the date is recorded.
3. The people he associated with, are recorded down to confirm if they are infected with the disease. If so, they are taken into isolation.
4. The staff who treat the patient are also recorded as part of his associates.
5. Every time an associate is admitted in hospital, the object moves to the Victim class, undergoes a dynamic evolution.
6. The attribute `is_dead` records whether the person is alive or not. If the patient dies, his attribute undergoes a persistent evolution PEV.

#### 5.1.4 Financial Institutions

Financial institutions regulate the allocation of assets and liabilities to organisation and people. They need to record history such as the history of customer transactions (deposit) and (withdrawal) for future use, necessary before giving a credit facility, showing the time value of money to determine the purchasing power of an economy. Determining the amount a client can receive and the time within which it can be repaid or determining the course of action they take during the decision making. For future recommendations, we have to inquire into the past to see what mistakes were made and what could have been avoided and what can be learnt from that data. This may also include analysis of sales or inventories which is important for business planning and prediction of the evolution of the financial data which is useful to project future growth. Below is the  $EER_{VT}^{++}$  model showing the loan repayment by a customer.

##### **Example 5.**

A company has a customer who took a loan from them. They want to store the payment plan of the customer as well as the interest rate change. The company, therefore needs to keep track of the payments from its customers. The attributes for the loan repayment are, `loanid`, `installmentNo`, `installmentAmount`, `penalty`, `previousOutstanding`, `totalReceivable`, `amountReceived`, the temporal attributes are interest rate, which is determined by the Central Bank, payment-frequency and the contract period which can change, depending on whether the customer will put additional payments at a later time.

1. A person with an existing car loan wants to get a mortgage, but his salary is not sufficient to permit him to take both loans at the same time, what happens?

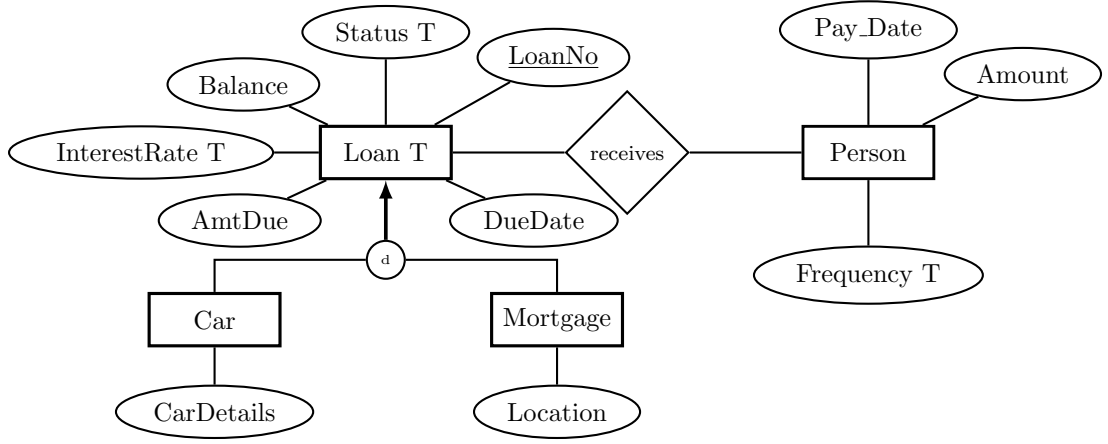


FIGURE 5.4:  $EER_{VT}^{++}$  diagram with transition constraints showing the financial institution example

- (a) At time  $t_0$ , first loan is active and the second loan is processed and approved, pending the clearance of the first loan.
- (b) At time  $t_1$ , loan2 is suspended, until the first loan is cleared.
- (c) At time  $t_2$ , the first loan is cleared, status is set to cleared, then the class is disabled.
- (d) At time  $t_3$ , the second loan is now activated.

## 5.2 Summary

We have shown how temporal attributes from different domains of application are captured in the  $\mathcal{ER}_{VT}$  model and how the formalisation of the conceptual model assists in achieving reasoning. The next chapter is the discussion of the results.





# 6

## DISCUSSION

The treatment of attributes was the remaining gap towards obtaining a fully temporalised conceptual data model, which is an important component, because without them, it would be difficult to fully represent and reason over temporal conceptual data models. This would cause it to be partial with respect to the temporal database or a knowledge base it was designed for. This chapter details the importance of formalising temporal attributes, how each of the research questions was answered and how the research can be used in other modelling languages as well as the research challenges.

### 6.1 Answering research questions

We return to our research questions and answer them in this section, on the research problem on how to represent temporal attributes in temporal models.

1. *Do conceptual models have temporal attributes? If so, how are they represented?*

Older temporal models extended temporal classes and relationships with little or no representation of temporal attributes as shown in Chapter 2. The only provision for temporal attributes in the  $\mathcal{ER}_{VT}$  model was timestamping, which showed the difference between temporal and snapshot attributes, and freezing of temporal attributes in a temporal class. However, this was not enough for a fully temporalised model, as we showed in Chapter 3, where we showed how temporal attributes changes can be captured in the model. In Chapter 3 and Chapter 5.1, temporal attributes needed to check the consistency of temporal models, which can permit automation of database systems, thus give an accurate representation of the mini world.

2. *Is it possible to have a fully temporal model with respect to the current EER model?*

Yes.

*If so,*

- (a) *how are temporal attributes added graphically on the temporal model?*

We provided a graphical syntax for the temporal model for transition constraints, a dashed line with the transition constraint.

- (b) *does the formalisation of temporal attributes add to reasoning properties in a temporal model?*

With the results presented in Chapter 3, we are able to get reasoning capabilities for temporal attributes. We are also able to reason over the interaction between temporal classes and temporal attributes and get implicit knowledge which allows us to check for inconsistencies.

3. *Do temporal attributes affect temporal classes and vice versa? If so, how?*

Our work in Chapter 3, Section 3.3 shows the interaction between temporal classes and temporal attributes and shows how temporal classes affect the attributes and also how temporal attributes can affect them, by having either less or greater values causes a change in the class status. For example, change of the CD4 count causes an HIV patient to become an AIDS patient.

The research questions above were the guide on how we approached this research, the methodology involved by doing an overview of existing temporal models and choosing to extend the modelling language  $\mathcal{ER}_{\mathcal{VT}}$ .

## 6.2 Discussion of the results

After formalising the temporal model, now known as  $\mathcal{ER}_{\mathcal{VT}}$ , we used it against the other three models, discussed in Chapter 2 to see if the model chosen is ideal with respect to the modified criteria for logic-based temporal models shown in Table 6.1. The temporal model,  $EER_{\mathcal{VT}}^{++}$  does not offer support for temporal granularities and it does not have a graphical editor, these useful extension are left for future works. With the additional constructs, we see that we have a fully temporal conceptual model with respect to classes, relationships and their attributes.  $EER_{\mathcal{VT}}^{++}$  meets the set criteria, with the exception of the lack of support for granularities and the graphical editor, which we put as future works.

This formalisation will allow modellers to design better temporal models with respect to the constraints specified. Now, we have a fully temporalised model that shows the interaction of temporal classes, relations and attributes. With these novel results, it is now possible to develop a temporal conceptual modelling tool that would enable a modeller to construct temporal conceptual data models in a precise, unambiguous way, by automatically checking the consistency of temporal models.

However, despite these achievements, there is scant information on temporal reasoners to permit reasoning over temporal conceptual data models. These results would serve as motivation for the development of a temporal reasoner. We leave to future works a more detailed investigation as to what would be the best trade-off between a subset of temporal constructs that is most desired from a viewpoint of conceptual data modelling

Modified criteria for modeling temporal models		$EER_{VT}^{++}$	$\mathcal{ER}_{VT}$	TIMEERplus MADS	
1. Time dimension support		VT & UDT	VT & UDT	BT & UDT	VT & UDT
2. Implicit verses Explicit Constructs		Explicit	Explicit	Implicit	Explicit
3. Mandatory and Optional		Optional	Optional	Optional	Optional
4. Time Datatypes support		Instant, Interval & TE	Instant, Interval & TE	Instant & TE	Instant, Interval & TE
5. Support for Granularities		No	No	Yes	No
6. Timestamping		Yes	Yes	Yes	Yes
7. Status	classes	Yes	Yes	No	No
	relations	Yes	Yes	No	No
	attributes	Yes	No	No	No
8. Transition	classes	Yes	Yes	No	Yes
	relations	Yes	Yes	No	Yes
	attributes	Yes	No	No	No
9. ISA	classes	Yes	Yes	No	Yes
	Relations	Yes	Yes	No	Yes
	attributes	Yes	No	No	No
10. Upward Compatibility		Yes	Yes	Yes	Yes
11. Snapshot Reducibility		Yes	Yes	Yes	Yes
12. Graphical notation provided		Yes	Yes	Yes	Yes
13. Graphical Editor		No	No	No	Yes

TABLE 6.1: Anaysis of  $EER_{VT}^{++}$  against  $\mathcal{ER}_{VT}$ , TIMERplus and MADS

and the complexity ‘costs’. For instance, it was not easy to invent examples for ADEX, but cases that fit AQEV or AQEX were abound, and once a temporal modelling tool is available, one can obtain quantitative results as to how often a construct is actually used, which further can inform the notion of ‘preferred constructs’ and the development of decidable temporal logics [54].

Other areas that use and need attributes as shown in the examples above will also permit the use of our results from subsumption and status classes to show which attribute class relations are permitted and which ones are not. The language can be used for both classes and relations although the full  $\mathcal{DLR}_{US}$  is undecidable.  $\mathcal{DLR}_{US}$  gave us a very broad field, in terms of expressivity, therefore we were able to represent many temporal constraints of attributes and derive new constraints.

$EER_{VT}^{++}$  does not have the capability of representing or showing value of attributes, say a boolean attribute that needs to be checked, we can enforce this by having the values alongside the attribute, as shown in Fig. 6.1. The figure shows the boolean attribute

transplant- received, once set ‘yes’, it undergoes a persistent evolution, APEV and cannot be changed anymore (it is so-called ‘frozen’).

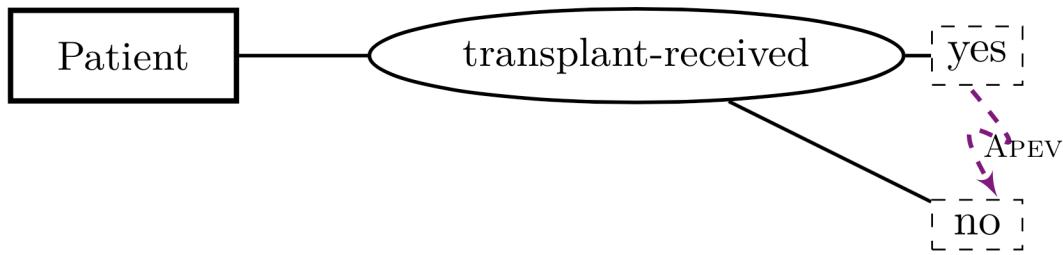


FIGURE 6.1: Example of evolution due to change in data. The dashed boxes represent values

### 6.3 Transferability to other modelling languages

Our results can be transferred to other conceptual modelling languages e.g. using UML, (see Fig. 6.3) and ORM, (see Fig. 6.2). We show the graphical representation of ORM and UML that allows us to capture temporal classes and attributes, along with transition constraints, which is similar to DL-based logical reconstructions of UML [18], ORM [50] and ORM2 [35] which eliminate inconsistencies by using the formalisation presented in Chapter 3. These results are not limited to conceptual models, but can be used in ontologies which require a formal proof to reason over complex data structures. Although the results presented are for temporal attributes, we can apply them for relationships to show the interaction of inheritance of classes and relations. For example, suspended relation in a subclass, say D, must be either suspended in the superclass C or the relation does not exist, which is similar to  $ISA4_A$ .

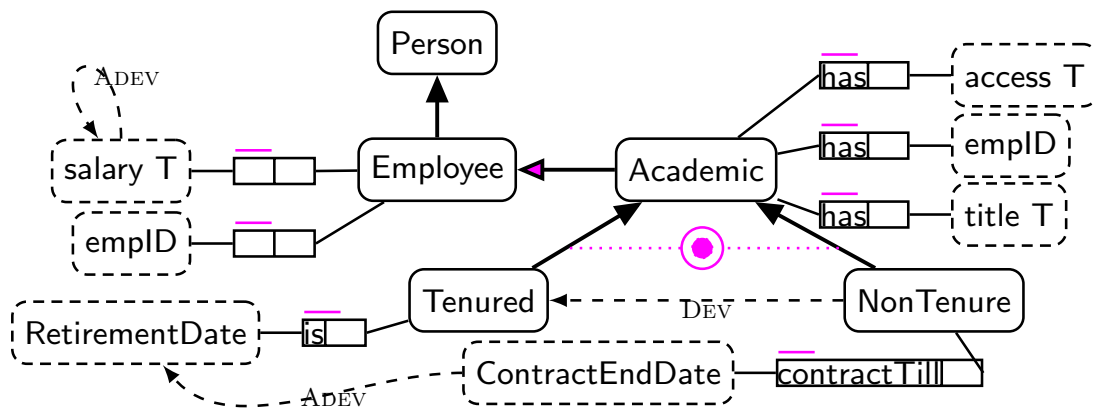


FIGURE 6.2: ORM diagram showing class and attribute transition constraints

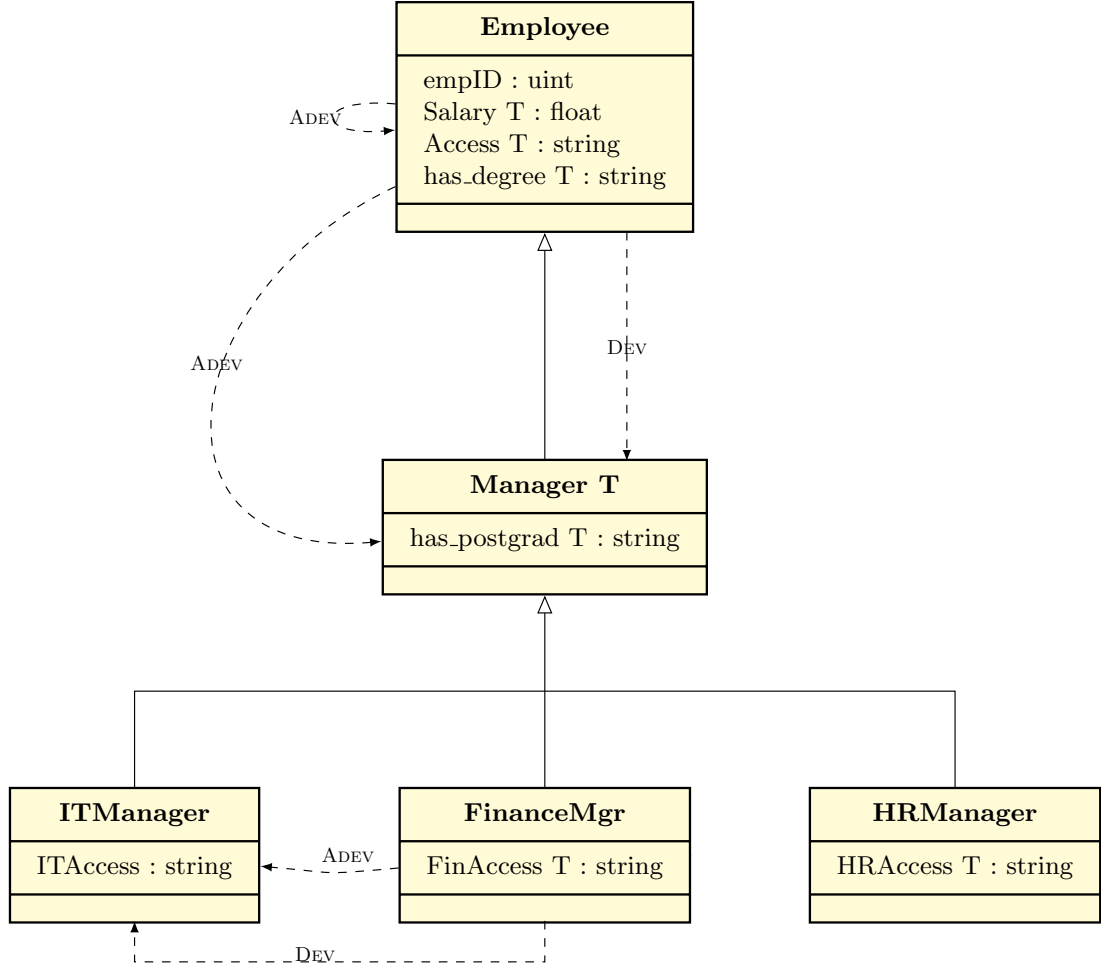


FIGURE 6.3: UML diagram showing class and attribute transition constraints

## 6.4 Challenges

$\mathcal{DLR}_{US}$  is undecidable, this may be an acceptable trade-off for a modeller focussing on expressiveness, however, a slightly less expressive temporal language and better performance with the reasoner may be preferred by others. However, some modellers might prefer a slightly less expressive temporal language than  $\mathcal{DLR}_{US}$  which gives better performance. In order to make  $\mathcal{DLR}_{US}$  decidable, it is deprived of its ability to talk about temporal persistence of n-ary relations which causes reasoning with atomic formulas to be EXPTIME-complete and satisfiability and logical implication of arbitrary formulas to be EXPSPACE-complete. There are also several decidable sub-languages that one can use such as TDL-Lite, which uses only two fragments of the  $\mathcal{DLR}_{US}$ , i.e. at the next moment and Until. (For recent complexity results of TDL-Lite on temporal conceptual models, see, [11]). The sub-languages however usually do not mix timestamping with evolution constraints therefore, you can only have either timestamping or evolution constraints, but not both [3].

With regards to DL-Knowledge bases, a complete translation from EER to a description logic language is needed to check quality properties of conceptual schema. Without the proper full temporisation of attributes, it becomes difficult to develop an algorithm that checks consistency of the model.

Although several transitions have been formalised using  $\mathcal{DLR}_{US}$ , not all attribute transitions can be represented in  $\mathcal{DLR}_{US}$  such as CASE B. This is because  $\mathcal{DLR}_{US}$  does not have a value comparison operator for attributes needed to give a comparison between two attributes. Due to this limitation, we cannot give a full formalisation. It is important, however, to note CASE B as a proposal for future investigation into a suitable description logic language, and to ensure that all the areas pertaining to attributes are covered.

# 7 | CONCLUSION

In this dissertation, the formalisation of temporal attributes using the extended description logics  $\mathcal{DLR}_{US}$  was presented. We also refined  $\mathcal{ER}_{VT}$  as well provided practical applications, in different domains. In this chapter, we conclude the work carried out in this study. Section 7.1 provides the summary of the contents of the previous chapters. Section 7.2 discusses the significance of the results achieved in the study. Section 7.3 provides the directions for future work.

## 7.1 Dissertation Achievement

Chapter 2 introduced the related work and analysed the existing temporal models and chose to extend  $\mathcal{ER}_{VT}$ . In Chapter 3, we answered the question “how do we capture evolving attributes in conceptual models?”. This was carried out using the extended  $\mathcal{DLR}_{US}$ , which allowed us to provide syntax and semantics for temporal attributes. Firstly, we extended the description logic language  $\mathcal{DLR}_{US}$  to include temporal attributes.  $\mathcal{DLR}_{US}$  was the description logic of choice in our formalisation, because, temporal attributes integrate well with the temporal entities [13] and temporal relations [51] already defined by the  $\mathcal{DLR}_{US}$  axioms. This allowed us to extend the conceptual modelling language  $\mathcal{ER}_{VT}$  to the  $EER_{VT}^{++}$  with the new semantics for temporal attributes.

Secondly, a rigorous definition of temporal attributes, refining the definition of time-tamping for attributes, and lifecycle. Lifecycle is the state of membership of an attribute, we tackled this by introducing the notion of status attributes. We extended the notion of status classes to status attributes to map the lifecycle of attributes and gave its logic based semantics, to rule the permissible evolution of attributes along an objects lifecycle. Thirdly, representing transition constraints for temporal attributes over evolving objects. We presented description logic axioms that describe attribute transition, both in an attribute, which showed the change in attribute values and during transition of evolving objects that caused the migration of temporal attributes. The definition of transition allowed us to see how a temporal attribute affects a temporal class and how temporal classes can be affected also with temporal attributes.

Fourthly, the research took an object oriented approach, which centres on the object in a class that can also form a hierarchy. We then looked at effect of subsumption object in a class that can also form a hierarchy. We then looked at effect of subsumption (ISA relations) of temporal classes (ISA) on temporal attributes which further enrich the data. Lastly, giving solution to real world problems in several application areas using the results from Chapter 3 and Chapter 4.

This work was the remaining gap towards obtaining a full logic-based temporised modelling language that is essential for designing and maintaining temporal databases and knowledge bases.

## 7.2 Significance of the Results

With the increase of information and the need to manage evolving data, there is a need for automation to ensure that the development of an error-free model. We can use our model to check automatically the properties of the schema for satisfiability and subsumption. For instance, our model will disallow the undesirable state where all employees and their attribute are suspended but managers still receive a salary, or forcing a customer through a payment procedure for a payment of R0,00 to download free software. The major benefit of this formalisation is the modelling of temporal information as  $\mathcal{DLR}_{US}$  provides a complete logic-based reconstruction of  $\mathcal{ER}_{VT}$  conceptual data modelling language. These results can also be used in ontologies, more specifically, OBDA [15] which is more of a formalised conceptual data model.

## 7.3 Future work

The research direction is two fold. One way is looking 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.

$\mathcal{DLR}_{US}$  is very expressive and allows full temporisation of attributes Despite its disadvantage of being undecidable, it is useful to first have a language with high expressivity to model what may be required, rather than complexity, to enable a more comprehensive understanding of temporal attributes and their role in temporal conceptual data models. We have been able to show how attributes can be formalised using description logics and in the process have fully formalise the  $\mathcal{ER}_{VT}$  model.

The second way is building a complete temporal ER model that would be translated from the temporally extended EER to a more decidable description logic and therewith enable the option of checking the consistency of a conceptual schema, thus, improving



or guaranteeing its quality. This is especially attractive for large complex diagrams that are used to implement systems, that need to be error-free before implementation.

The third way is to extend the work to develop an algorithm that can automatically map the temporal conceptual data model  $EER_{VT}^{++}$  to a relational database.



# A | $\mathcal{ER}_{\mathcal{VT}}$ CLASSES AND RELATIONSHIPS

We give the rules used in [13] for status classes and status relations [8] with their corresponding  $\mathcal{DLR}_{\mathcal{US}}$  axioms as well as their transition constraints.

## A.1 Those involving Classes

(EXISTS) Existence persists until Disabled.

$$\begin{aligned} o \in \text{Exists-}C^{\mathcal{I}(t)} &\rightarrow \forall t' > t. o \in (\text{Exists-}C^{\mathcal{I}(t')} \vee \text{Disabled-}C^{\mathcal{I}(t')}) \\ \text{Exists-C} &\sqsubseteq \Box^+(\text{Exists-C} \sqcup \text{Disabled-C}) \end{aligned}$$

(DISAB1) Disabled persists.

$$\begin{aligned} o \in \text{Disabled-}C^{\mathcal{I}(t)} &\rightarrow \forall t' > t. o \in \text{Disabled-}C^{\mathcal{I}(t')} \\ \text{Disabled-C} &\sqsubseteq \Box^+ \text{Disabled-C} \end{aligned}$$

(DISAB2) Disabled was Active in the past.

$$\begin{aligned} o \in \text{Disabled-}C^{\mathcal{I}(t)} &\rightarrow \exists t' < t. o \in C^{\mathcal{I}(t')} \\ \text{Disabled-C} &\sqsubseteq \Diamond^- \text{C} \end{aligned}$$

(SUSP) Suspended was Active in the past.

$$\begin{aligned} o \in \text{Suspended-}C^{\mathcal{I}(t)} &\rightarrow \exists t' < t. o \in C^{\mathcal{I}(t')} \\ \text{Suspended-C} &\sqsubseteq \Diamond^- \text{C} \end{aligned}$$

(SCH1) Scheduled will eventually be active

$$\begin{aligned} o \in \text{Scheduled-}C^{\mathcal{I}(t)} &\rightarrow \exists t' > t. o \in C^{\mathcal{I}(t')} \\ \text{Scheduled-C} &\sqsubseteq \Diamond^+ \text{C} \end{aligned}$$

(SCH2) Scheduled can never follow Active.

$$\begin{aligned} o \in C^{\mathcal{I}(t)} &\rightarrow \forall t' > t. o \notin C^{\mathcal{I}(t')} \\ \text{C} &\sqsubseteq \Box^+ \neg \text{Scheduled-C} \end{aligned}$$

(FREEZ) Freezing attributes of suspended classes.

$$o \in \text{Suspended-}C^{\mathcal{I}(t)} \sqcap \langle o, a \rangle \in A^{\mathcal{I}(t)} \rightarrow \langle o_1, o_2 \rangle \in A^{\mathcal{I}((t_1))}$$

$$\text{Suspended-}C \sqsubseteq \neg \exists [\text{From}](A \sqcap \ominus A)$$

### Logical implications from status classes.

(DISA3) Disabled will never become active anymore.

$$\text{Disabled} - C \sqsubseteq \diamond^+ \neg C$$

(SCH3) Scheduled persists until active.

$$\text{Scheduled} - C \sqsubseteq \text{Scheduled-}C \mathcal{U} C$$

(SCH4) Scheduled cannot evolve directly to Disabled.

$$\text{Scheduled} - C \sqsubseteq \oplus \neg \text{Disabled-}C$$

### ISA for Objects

(ISA1) Objects active in B must be active in A, i.e.  $B \sqsubseteq A$ ,

(ISA2) Objects suspended in B must be either suspended or active in A,

$$\text{i.e., } \text{Suspended-}B \sqsubseteq (\text{Suspended-}A \sqcup A),$$

(ISA3) Objects disabled in B must be either disabled, suspended or active in A,

$$\text{i.e., } \text{Disabled-}B \sqsubseteq (\text{Disabled-}A \sqcup A \sqcup \text{Suspended-}A),$$

(ISA4) Objects scheduled in B must exist in A,

$$\text{i.e., } \text{Scheduled-}B \sqsubseteq \text{Exists-}A,$$

(ISA5) Objects disabled in A, and active in B in the past, must be disabled in B,

$$\text{i.e., } \text{Disabled-}A \sqcap \diamond^- B \sqsubseteq \text{Disabled-}B.$$

### Transition Constraints for temporal classes

DEX

$$o \in \text{DEX}_{C_1, C_2}^{\mathcal{I}(t)} \rightarrow o \in (\text{Suspended-}C_1^{\mathcal{I}(t)} \sqcup C_1^{\mathcal{I}(t)}) \wedge o \notin C_2^{\mathcal{I}(t)} \wedge o \in C_2^{\mathcal{I}(t+1)}$$

$$\text{DEX}_{C_1, C_2} \sqsubseteq (\text{Suspended-}C_1 \sqcup C_1) \sqcap \neg C_2 \sqcap \oplus C_2$$

DEV

$$o \in \text{DEV}_{C_1, C_2}^{\mathcal{I}(t)} \rightarrow o \in (\text{Suspended-}C_1^{\mathcal{I}(t)} \sqcup C_1^{\mathcal{I}(t)}) \wedge o \notin C_2^{\mathcal{I}(t)} \wedge o \in C_2^{\mathcal{I}(t+1)} \wedge o \notin C_1^{\mathcal{I}(t+1)}$$

$$\text{DEV}_{C_1, C_2} \sqsubseteq (\text{Suspended} - C_1 \sqcup C_1) \sqcap \neg C_2 \sqcap \oplus (C_2 \sqcup \neg C_1)$$

### Logical Implication from Transition of classes

The classes  $\text{DEX}_{C_1, C_2}$  and  $\text{DEV}_{C_1, C_2}$  are temporary classes; actually, they hold at single time points.

$$\text{DEX}_{C_1, C_2} \sqsubseteq \oplus \neg \text{DEX}_{C_1, C_2} \sqcup \ominus \neg \text{DEX}_{C_1, C_2}$$

$$\text{DEV}_{C_1, C_2} \sqsubseteq \oplus \neg \text{DEV}_{C_1, C_2} \sqcup \ominus \neg \text{DEV}_{C_1, C_2}$$

Objects in the classes  $\text{DEX}_{C_1, C_2}$  and  $\text{DEV}_{C_1, C_2}$  cannot be disabled as  $C_2$ .

$$\text{DEX}_{C_1, C_2} \sqsubseteq \neg \text{Disabled-}C_2$$

$$\text{DEV}_{C_1, C_2} \sqsubseteq \neg \text{Disabled-}C_2$$

The target class  $C_2$  cannot be snapshot (it becomes temporary in case of both TTT and TTE constraints).

$$\text{DEX}_{C_1, C_2} \sqsubseteq \diamond^* [C_2 \sqcap (\diamond^+ \neg C_2 \sqcup \diamond^- \neg C_2)]$$

As a consequence of dynamic evolution, the source class,  $C_1$ , cannot be snapshot (and it becomes temporary in case of STE constraints).

$$\text{DEV}_{C_1, C_2} \sqsubseteq \diamond^* [C_1 \sqcap (\diamond^+ \neg C_1 \sqcup \diamond^- \neg C_1)]$$

Dynamic evolution cannot be specified between a class and one of its sub-classes.

$$C_2 \sqsubseteq C_1 \models \text{DEV}_{C_1, C_2} \sqsubseteq \perp$$

Dynamic extension between disjoint classes logically implies Dynamic evolution.

$$\text{DEX}_{C_1, C_2}, C_1 \sqsubseteq \neg C_2 \models \text{DEV}_{C_1, C_2}$$

## A.2 Those involving Relations

(ACT) Active relations involve only active classes

$$\langle o_1, o_2 \rangle \in R^{\mathcal{I}(t)} \rightarrow o \in C^{\mathcal{I}(t)}, i = 1, 2$$

$$R \sqsubseteq \bigcup_i : C, i = 1, 2$$

(REXISTS) Existence persists until Disabled

$$\langle o_1, o_2 \rangle \in \text{Exists-}R^{\mathcal{I}(t)} \rightarrow \forall t' > t. \langle o_1, o_2 \rangle \in (\text{Exists-}R^{\mathcal{I}(t')} \vee \text{Disabled-}R^{\mathcal{I}(t')})$$

$$\text{Exists-}R \sqsubseteq \Box^+(\text{Exists-A} \sqcup \text{Disabled-A})$$

(RDISAB1) Disabled persists

$$\langle o_1, o_2 \rangle \in \text{Disabled-}R^{\mathcal{I}(t)} \rightarrow \forall t' > t. \langle o_1, o_2 \rangle \in \text{Disabled-}R^{\mathcal{I}(t')}$$

$$\text{Disabled-}R \sqsubseteq \Box^+ \text{Disabled-}R$$

(RDISAB2) Disabled was Active in the past.

$$\langle o_1, o_2 \rangle \in \text{Disabled-}R^{\mathcal{I}(t)} \rightarrow \exists t' < t. \langle o_1, o_2 \rangle \in R^{\mathcal{I}(t')}$$

$$\text{Disabled-}R \sqsubseteq \Diamond^- R$$

(RSUSP1) Suspended was Active in the past.

$$\langle o_1, o_2 \rangle \in \text{Suspended-}R^{\mathcal{I}(t)} \rightarrow \exists t' < t. \langle o_1, o_2 \rangle \in R^{\mathcal{I}(t')}$$

$$\text{Suspended-R} \sqsubseteq \Diamond^- \text{R}$$

(RSCH1) Scheduled will eventually be active

$$\langle o_1, o_2 \rangle \in \text{Scheduled-}R^{\mathcal{I}(t)} \rightarrow \exists t' > t. \langle o_1, o_2 \rangle \in R^{\mathcal{I}(t')}$$

$$\text{Scheduled-R} \sqsubseteq \Diamond^+ \text{R}$$

(RSCH2) Scheduled can never follow Active.

$$\langle o_1, o_2 \rangle \in R^{\mathcal{I}(t)} \rightarrow \forall t' > t. \langle o_1, o_2 \rangle \notin R^{\mathcal{I}(t')}$$

$$\text{R} \sqsubseteq \Box^+ \neg \text{Scheduled-R}$$

## Logical Implications

(RACT) Active will possible evolve into Suspended or Disabled.

$$\text{R} \sqsubseteq \Diamond^+ (\text{R} \sqcup \text{Suspended-R} \sqcup \text{Disabled-R})$$

(RDISAB3) Disabled will never become active anymore.

$$\text{Disabled-R} \sqsubseteq \Box^+ \neg \text{R}$$

(RDISAB4) Disabled classes can participate only in disabled relations.

$$\text{Disabled-C}_i \sqcap \Diamond^- [\text{U}_i] \text{R} \sqsubseteq \exists [\text{U}_i] \text{Disabled-R}$$

(RDISAB5) Disabled relations involve active, suspended, or disabled classes.

$$\text{Disabled-R} \sqcup \text{U}_i : (\text{C} \sqcup \text{Suspended} - \text{C} \sqcup \text{Disabled} - \text{C}), \forall i = 1, \dots, n$$

(RSCH3) Scheduled persists until active.

$$\text{Scheduled-R} \sqsubseteq \text{Scheduled-R} \mathcal{U} \text{R}$$

(RSCH4) Scheduled cannot evolve directly to Disabled.

$$\text{Scheduled-R} \sqcup \oplus \neg \text{Disabled-R}$$

(RSCH5) Scheduled relations do not involve disabled classes.

$$\text{Scheduled-R} \sqcup \text{U}_i \neg \text{Disabled-C}_i; \forall i = 1, \dots, n$$

## REFERENCES

- [1] ARTALE, A., AND FRANCONI, E. Introducing temporal description logics. In *6th International Workshop on Temporal Representation and Reasoning, TIME '99, Orlando, Florida, USA, May 1-2, 1999* (1999), IEEE Computer Society, pp. 2–5.
- [2] ARTALE, A., AND FRANCONI, E. Temporal ER modeling with description logics. In *Conceptual Modeling - ER '99, 18th International Conference on Conceptual Modeling, Paris, France, November, 15-18, 1999, Proceedings* (1999), J. Akoka, M. Bouzeghoub, I. Comyn-Wattiau, and E. Métais, Eds., vol. 1728 of *Lecture Notes in Computer Science*, Springer, pp. 81–95.
- [3] ARTALE, A., AND FRANCONI, E. Foundations of temporal conceptual data models. In *Conceptual Modeling: Foundations and Applications - Essays in Honor of John Mylopoulos* (2009), A. Borgida, V. K. Chaudhri, P. Giorgini, and E. S. K. Yu, Eds., vol. 5600 of *Lecture Notes in Computer Science*, Springer, pp. 10–35.
- [4] ARTALE, A., FRANCONI, E., AND MANDREOLI, F. Description logics for modeling dynamic information. In *Logics for Emerging Applications of Databases [outcome of a Dagstuhl seminar]* (2003), J. Chomicki, R. van der Meyden, and G. Saake, Eds., Springer, pp. 239–275.
- [5] ARTALE, A., FRANCONI, E., MOSUROVIC, M., WOLTER, F., AND ZAKHARYASCHEV, M. The DLRUS temporal description logic. In *Working Notes of the 2001 International Description Logics Workshop (DL-2001), Stanford, CA, USA, August 1-3, 2001* (2001), C. A. Goble, D. L. McGuinness, R. Möller, and P. F. Patel-Schneider, Eds., vol. 49 of *CEUR Workshop Proceedings*, CEUR-WS.org.
- [6] ARTALE, A., FRANCONI, E., WOLTER, F., AND ZAKHARYASCHEV, M. A temporal description logic for reasoning over conceptual schemas and queries. In *Logics in Artificial Intelligence, European Conference, JELIA 2002, Cosenza, Italy, September, 23-26, Proceedings* (2002), S. Flesca, S. Greco, N. Leone, and G. Ianni, Eds., vol. 2424 of *Lecture Notes in Computer Science*, Springer, pp. 98–110.

- [7] ARTALE, A., GUARINO, N., AND KEET, C. M. Formalising temporal constraints on part-whole relations. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008* (2008), G. Brewka and J. Lang, Eds., AAAI Press, pp. 673–683.
- [8] ARTALE, A., AND KEET, C. M. Essential and mandatory part-whole relations in conceptual data models. In *Proceedings of the 21st International Workshop on Description Logics (DL2008), Dresden, Germany, May 13-16, 2008* (2008), F. Baader, C. Lutz, and B. Motik, Eds., vol. 353 of *CEUR Workshop Proceedings*, CEUR-WS.org.
- [9] ARTALE, A., KONTCHAKOV, R., LUTZ, C., WOLTER, F., AND ZAKHARYASCHEV, M. Temporalising tractable description logics. In *14th International Symposium on Temporal Representation and Reasoning (TIME 2007), 28-30 June 2007, Alicante, Spain* (2007), IEEE Computer Society, pp. 11–22.
- [10] ARTALE, A., KONTCHAKOV, R., RYZHIKOV, V., AND ZAKHARYASCHEV, M. Complexity of reasoning over temporal data models. In *Conceptual Modeling - ER 2010, 29th International Conference on Conceptual Modeling, Vancouver, BC, Canada, November 1-4, 2010. Proceedings* (2010), J. Parsons, M. Saeki, P. Shoval, C. C. Woo, and Y. Wand, Eds., vol. 6412 of *Lecture Notes in Computer Science*, Springer, pp. 174–187.
- [11] ARTALE, A., KONTCHAKOV, R., RYZHIKOV, V., AND ZAKHARYASCHEV, M. A cookbook for temporal conceptual data modelling with description logics. *ACM Trans. Comput. Log.* 15, 3 (2014), 25.
- [12] ARTALE, A., KONTCHAKOV, R., WOLTER, F., AND ZAKHARYASCHEV, M. Temporal description logic for ontology-based data access. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013* (2013), F. Rossi, Ed., IJCAI/AAAI.
- [13] ARTALE, A., PARENT, C., AND SPACCAPIETRA, S. Evolving objects in temporal information systems. *Ann. Math. Artif. Intell.* 50, 1-2 (2007), 5–38.
- [14] ARTALE, A., RYZHIKOV, V., AND KONTCHAKOV, R. Dl-lite with attributes and datatypes. In *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31, 2012* (2012), L. D. Raedt, C. Bessière, D. Dubois, P. Doherty, P. Frasconi, F. Heintz, and P. J. F. Lucas, Eds., vol. 242 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, pp. 61–66.



- [15] BAADER, F., BORGWARDT, S., AND LIPPMANN, M. Temporalizing ontology-based data access. In *Automated Deduction - CADE-24 - 24th International Conference on Automated Deduction, Lake Placid, NY, USA, June 9-14, 2013. Proceedings* (2013), M. P. Bonacina, Ed., vol. 7898 of *Lecture Notes in Computer Science*, Springer, pp. 330–344.
- [16] BAADER, F., CALVANESE, D., MCGUINNESS, D. L., NARDI, D., AND PATEL-SCHNEIDER, P. F., Eds. *The Description Logic Handbook: Theory, Implementation, and Applications* (2003), Cambridge University Press.
- [17] BAADER, F., HORROCKS, I., AND SATTLER, U. Description logics. In *Handbook of Knowledge Representation*, F. van Harmelen, V. Lifschitz, and B. Porter, Eds. Elsevier, 2007, pp. 135–179.
- [18] BERARDI, D., CALVANESE, D., AND DE GIACOMO, G. Reasoning on UML class diagrams. *Artif. Intell.* 168, 1-2 (2005), 70–118.
- [19] BETTINI, C., JAJODIA, S., AND WANG, X. S. *Time granularities in databases, data mining, and temporal reasoning*. Springer, 2000.
- [20] BOOCH, G., RUMBAUGH, J. E., AND 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.
- [21] CABOT, J., OLIVÉ, A., AND TENIENTE, E. Entity types derived by symbol-generating rules. In *Conceptual Modeling - ER 2003, 22nd International Conference on Conceptual Modeling, Chicago, IL, USA, October 13-16, 2003, Proceedings* (2003), I. Song, S. W. Liddle, T. W. Ling, and P. Scheuermann, Eds., vol. 2813 of *Lecture Notes in Computer Science*, Springer, pp. 376–389.
- [22] CALVANESE, D., AND DE GIACOMO, G. Expressive description logics. In *The Description Logic Handbook: Theory, Implementation, and Applications* (2003), F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., Cambridge University Press, pp. 178–218.
- [23] CALVANESE, D., DE GIACOMO, G., AND LENZERINI, M. On the decidability of query containment under constraints. In *Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 1-3, 1998, Seattle, Washington, USA* (1998), A. O. Mendelzon and J. Paredaens, Eds., ACM Press, pp. 149–158.
- [24] CALVANESE, D., LENZERINI, M., AND NARDI, D. Unifying class-based representation formalisms. *Journal of Artificial Intelligence Research (JAIR)* 11 (1999), 199–240.

- [25] CHEN, P. P. The entity-relationship model: Toward a unified view of data. In *Proceedings of the International Conference on Very Large Data Bases, September 22-24, 1975, Framingham, Massachusetts, USA.* (1975), D. S. Kerr, Ed., ACM, p. 173.
- [26] CHEN, P. P. Entity-relationship modeling: historical events, future trends, and lessons learned. In *In: Software Pioneers: Contributions to Software Engineering* (2002), Springer, pp. 297–310.
- [27] CHOMICKI, J., AND TOMAN, D. Temporal logic in information systems. In *Logics for Databases and Information Systems (the book grew out of the Dagstuhl Seminar 9529: Role of Logics in Information Systems, 1995)* (1998), J. Chomicki and G. Saake, Eds., Kluwer, pp. 31–70.
- [28] CODD, E. F. A relational model of data for large shared data banks. *Commun. ACM* 13, 6 (1970), 377–387.
- [29] DA ROCHA, L. V., EDELWEISS, N., AND IOCHPE, C. Geoframe-t: A temporal conceptual framework for data modeling. In *ACM-GIS 2001, Proceedings of the Ninth ACM International Symposium on Advances in Geographic Information Systems, Atlanta, GA, USA, November 9-10, 2001* (2001), W. G. Aref, Ed., ACM, pp. 124–129.
- [30] DEMEY, Y. T., AND PANETTO, H., Eds. *On the Move to Meaningful Internet Systems: OTM 2013 Workshops - Confederated International Workshops: OTM Academy, OTM Industry Case Studies Program, ACM, EI2N, ISDE, META4eS, ORM, SeDeS, SINCOM, SMS, and SOMOCO 2013, Graz, Austria, September 9 - 13, 2013, Proceedings* (2013), vol. 8186 of *Lecture Notes in Computer Science*, Springer.
- [31] DYRESON, C. E. Chronon. In *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Springer US, 2009, p. 329.
- [32] ELMASRI, R., AND NAVATHE, S. B. *Fundamentals of Database Systems, 3rd Edition.* Addison-Wesley-Longman, 2000.
- [33] ETZION, O., GAL, A., AND SEGEV, A. *Temporal Databases – Research and Practice*, vol. 1399 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998, ch. Extended update functionality in temporal databases, pp. 56–95.
- [34] FILLOTTRANI, P. R., FRANCONI, E., AND TESSARIS, S. The ICOM 3.0 intelligent conceptual modelling tool and methodology. *Semantic Web* 3, 3 (2012), 293–306.
- [35] FRANCONI, E., MOSCA, A., AND SOLOMAKHIN, D. ORM2 encoding into description logic (extended abstract). In *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012* (2012), Y. Kazakov,

- D. Lembo, and F. Wolter, Eds., vol. 846 of *CEUR Workshop Proceedings*, CEUR-WS.org.
- [36] GREGERSEN, H. Timeer *plus*: A temporal EER model supporting schema changes. In *Database: Enterprise, Skills and Innovation, 22nd British National Conference on Databases, BNCOD 22, Sunderland, UK, July 5-7, 2005, Proceedings (2005)*, M. Jackson, D. Nelson, and S. Stirk, Eds., vol. 3567 of *Lecture Notes in Computer Science*, Springer, pp. 41–59.
- [37] GREGERSEN, H., AND JENSEN, C. S. Temporal entity-relationship models - A survey. *IEEE Trans. Knowl. Data Eng.* 11, 3 (1999), 464–497.
- [38] GREGERSEN, H., MARK, L., AND JENSEN, C. S. Mapping temporal er diagrams to relational schemas. Tech. rep., TimeCenter Technical Report, 1998.
- [39] HALL, G., AND GUPTA, R. Modeling transition. In *Proceedings of the Seventh International Conference on Data Engineering, April 8-12, 1991, Kobe, Japan (1991)*, IEEE Computer Society, pp. 540–549.
- [40] HALPIN, T. A. Temporal modeling and ORM. In *On the Move to Meaningful Internet Systems: OTM 2008 Workshops, OTM Confederated International Workshops and Posters, ADI, AWeSoMe, COMBEK, EI2N, IWSSA, MONET, OnToContent + QSI, ORM, PerSys, RDDS, SEMELS, and SWWS 2008, Monterrey, Mexico, November 9-14, 2008. Proceedings (2008)*, R. Meersman, Z. Tari, and P. Herrero, Eds., vol. 5333 of *Lecture Notes in Computer Science*, Springer, pp. 688–698.
- [41] HALPIN, T. A. Object-role modeling: Principles and benefits. *IJISMD* 1, 1 (2010), 33–57.
- [42] HALPIN, T. A., AND MORGAN, T. *Information modeling and relational databases (2. ed.)*. Morgan Kaufmann, 2008.
- [43] HODKINSON, I. M., WOLTER, F., AND ZAKHARYASCHEV, M. Decidable fragment of first-order temporal logics. *Ann. Pure Appl. Logic* 106, 1-3 (2000), 85–134.
- [44] HU, V. C., FERRAILOLO, D., KUHN, R., FRIEDMAN, A. R., LANG, A. J., COGDELL, M. M., SCHNITZER, A., SANDLIN, K., MILLER, R., AND SCARFONE, K. Guide to attribute based access control (abac) definition and considerations (draft). *NIST Special Publication 800* (2013), 162.
- [45] JARRAR, M. Mapping ORM into the SHOIN/OWL description logic. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, OTM Confederated International Workshops and Posters, AWeSoMe, CAMS, OTM Academy Doctoral Consortium, MONET, OnToContent, ORM, PerSys, PPN, RDDS, SSWS, and SWWS 2007*,

- Vilamoura, Portugal, November 25-30, 2007, *Proceedings, Part I* (2007), R. Meersman, Z. Tari, and P. Herrero, Eds., vol. 4805 of *Lecture Notes in Computer Science*, Springer, pp. 729–741.
- [46] JARRAR, M. Towards automated reasoning on ORM schemes. In *Conceptual Modeling - ER 2007, 26th International Conference on Conceptual Modeling, Auckland, New Zealand, November 5-9, 2007, Proceedings* (2007), C. Parent, K. Schewe, V. C. Storey, and B. Thalheim, Eds., vol. 4801 of *Lecture Notes in Computer Science*, Springer, pp. 181–197.
- [47] JENSEN, C. S., AND SNODGRASS, R. T. Temporal data management. *IEEE Trans. Knowl. Data Eng.* 11, 1 (1999), 36–44.
- [48] JUNGCLAUS, R., SAAKE, G., HARTMANN, T., AND SERNADAS, C. TROLL - A language for object-oriented specification of information systems. *ACM Trans. Inf. Syst.* 14, 2 (1996), 175–211.
- [49] KEET, C. M. Prospects for and issues with mapping the object-role modeling language into dlrlfd. In *Proceedings of the 2007 International Workshop on Description Logics (DL2007), Brixen-Bressanone, near Bozen-Bolzano, Italy, 8-10 June, 2007* (2007), D. Calvanese, E. Franconi, V. Haarslev, D. Lembo, B. Motik, A. Turhan, and S. Tessaris, Eds., vol. 250 of *CEUR Workshop Proceedings*, CEUR-WS.org.
- [50] KEET, C. M. Ontology-driven formal conceptual data modeling for biological data analysis. In *Biological Knowledge Discovery Handbook: Preprocessing, Mining and Postprocessing of Biological Data*, M. Elloumi and A. Y. Zomaya, Eds. Wiley, 2014, ch. 6, pp. 129–154.
- [51] KEET, C. M., AND ARTALE, A. A basic characterization of relation migration. In *On the Move to Meaningful Internet Systems: OTM 2010 Workshops - Confederated International Workshops and Posters: International Workshops: AVYTAT, ADI, DATAVIEW, EI2N, ISDE, MONET, OnToContent, ORM, P2P-CDVE, SeDeS, SWWS and OTMA. Hersonissos, Crete, Greece, October 25-29, 2010. Proceedings* (2010), R. Meersman, T. S. Dillon, and P. Herrero, Eds., vol. 6428 of *Lecture Notes in Computer Science*, Springer, pp. 484–493.
- [52] KHATRI, V., RAM, S., SNODGRASS, R. T., AND TEREZIANI, P. Capturing telic/atelic temporal data semantics: Generalizing conventional conceptual models. *IEEE Trans. Knowl. Data Eng.* 26, 3 (2014), 528–548.
- [53] LORA, R., SABAINI, A., COMBI, C., AND MORETTI, U. Designing the reconciled schema for a pharmacovigilance data warehouse through a temporally-enhanced

- er model. In *SHB* (2012), C. C. Yang, H. Chen, H. D. Wactlar, C. Combi, and X. Tang, Eds., ACM, pp. 17–24.
- [54] LUTZ, C., WOLTER, F., AND ZAKHARYASCHEV, M. Temporal description logics: A survey. In *15th International Symposium on Temporal Representation and Reasoning, TIME 2008, Université du Québec à Montréal, Canada, 16-18 June 2008* (2008), S. Demri and C. S. Jensen, Eds., IEEE Computer Society, pp. 3–14.
- [55] MCBRIEN, P. Temporal constraints in non-temporal data modelling languages. In *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings* (2008), Q. Li, S. Spaccapietra, E. S. K. Yu, and A. Olivé, Eds., vol. 5231 of *Lecture Notes in Computer Science*, Springer, pp. 412–425.
- [56] MYLOPOULOS, J. The extended entity-relationship model. University Lecture, 2004.
- [57] OBJECT MANAGEMENT GROUP. Superstructure specification. Standard 2.4.1, Object Management Group, 2012. <http://www.omg.org/spec/UML/2.4.1/>.
- [58] PARENT, C., SPACCAPIETRA, S., AND ZIMÁNYI, E. Spatio-temporal conceptual models: Data structures + space + time. In *ACM-GIS '99, Proceedings of the 7th International Symposium on Advances in Geographic Information Systems, November 2-6, 1999, Kansas City, USA* (1999), C. B. Medeiros, Ed., ACM, pp. 26–33.
- [59] PARENT, C., SPACCAPIETRA, S., AND ZIMÁNYI, E. *Conceptual modeling for traditional and spatio-temporal applications - the MADS approach*. Springer, 2006.
- [60] PARENT, C., SPACCAPIETRA, S., AND ZIMÁNYI, E. The MADS data model: Concepts to understand the structure of your spatial and temporal data. *Journal of Informative Modelling for the Architectural Heritage* 0, 1 (July 2006), 51–64.
- [61] SNODGRASS, R. T. *Developing Time-oriented Database Applications in SQL*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000.
- [62] SNODGRASS, R. T., AND AHN, I. Temporal databases. *IEEE Computer* 19, 9 (1986), 35–42.
- [63] SONG, I.-Y., EVANS, M., AND PARK, E. K. A comparative analysis of entity-relationship diagrams. *Journal of Computer and Software Engineering* 3, 4 (1995), 427–459.
- [64] SPACCAPIETRA, S., PARENT, C., AND ZIMÁNYI, E. Modeling time from a conceptual perspective. In *Proceedings of the 1998 ACM CIKM International Conference on*

- Information and Knowledge Management, Bethesda, Maryland, USA, November 3-7, 1998* (1998), G. Gardarin, J. C. French, N. Pissinou, K. Makki, and L. Bouganim, Eds., ACM, pp. 432–440.
- [65] STEINER, A. *A generalisation approach to temporal data models and their implementations*. PhD thesis, ETH Zürich, Switzerland, 1998.
- [66] TANSEL, A. U. Temporal databases. In *Wiley Encyclopedia of Computer Science and Engineering*, B. W. Wah, Ed. John Wiley & Sons, Inc., 2008.
- [67] VAN DER AALST, W. M. P., TER HOFSTEDÉ, A. H. M., AND WESKE, M. Business process management: A survey. In *Business Process Management, International Conference, BPM 2003, Eindhoven, The Netherlands, June 26-27, 2003, Proceedings* (2003), W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, Eds., vol. 2678 of *Lecture Notes in Computer Science*, Springer, pp. 1–12.
- [68] ZHU, Y., HU, H., AHN, G.-J., GONG, X., AND CHEN, S. Poster: Temporal attribute-based encryption in clouds. In *Proceedings of the 18th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2011), CCS '11, ACM, pp. 881–884.
- [69] ZIMÁNYI, E., PARENT, C., SPACCAPIETRA, S., AND PIROTTE, A. TERC+: a temporal conceptual model. In *Proceedings of the International Symposium on Digital Media Information Base, DMIB'97* (Nara, Japan, November 1997).