ISBN 978-981-11-3671-9
Proceedings of 2017 the 7th International Workshop on Computer Science and Engineering (WCSE 2017)
Beijing, 25-27 June, 2017, pp. 331-336
doi: 10.18178/wcse.2017.06.057

Extending and Implementing Anchor Modeling, with a Simple Application

YANG Yongzhi[+], ZHOU Wenwen, CHENG Bin, ZOU Peng

The Academy of Equipment Beijing, China

Abstract. Anchor Modeling (AM) is proposed to enhance the entity relationship model so that the constantly changing data can be appended to the data warehouse agilely. This paper discusses the model extending and tool chain implementing of AM. First, free meta-dimensions except for the time dimension are introduced to improve the descriptive capability of AM. Second, another visual modeling tool based on the Generic Modeling Environment (GME) is presented by building the meta-model of AM in the GME. Third, 3 types of access interfaces for AM database are implemented on the Java platform. Finally, a graphic user interface for data creating, retrieving, updating and deleting is provided for AM tables. And a simple application is provided to demonstrate the above works on AM.

Keywords: anchor modeling, entity relation model, tool chain, generic modeling environment

1. Introduction

As an agile information modeling technique, Anchor Modeling (AM) [1] offers non-destructive extensibility mechanisms, thereby enabling robust and flexible management of changes. From the perspective of the traditional Entity Relationship Model (ERM) [2-4], the time as a meta dimension/information is attached to the attributes and relations (ties in AM). With the time dimension, the whole history of the changing data is recorded in the database. And a key benefit of AM is that changes in a data warehouse environment only require extensions, not modifications, to the data warehouse.

However, Jovanovic and Bojicic point out that AM recognizes (implicitly) only Time, thus severely limiting its usefulness for complex environments [5]. And software developers always need a set of tool chain before they can apply AM in their products.

This paper extends the Anchor Modeling and implements necessary tool chain for designing and implementing of applications based on AM. Free meta-dimensions except for the time dimension are introducedto improve the descriptive capability of AM. Both meta-model and mathematical definitions of the extended AM are discussed. And some ERM style extension elements for programming practice are introduced. Another visual modeling tool based on the Generic Modeling Environment (GME) [6] is presented by building the meta-model of AM. In GME, a UML based meta model is defined to generate the visual editor and a Java plugin is implemented to export Schema from graphical elements to XML text. Three types of access interfaces for AM database are implemented on the Java platform. A graphic user interface for data creating, retrieving, updating and deleting is provided for AM tables. And a simple application is used to demonstrate the above works on AM.

In the following sections, the related works are described in section 2; the extending and implementing of AM are presented in section 3 and section 4, and a simple application is used as demonstration. And the conclusion comes in senction 5.

[+] Corresponding author. Tel.: + (86 010 66366884); fax: +(86 010 66366884).
E-mail address: cnziberg@gmail.com.

2. Related Works

Anchor Modeling (AM) is first introduced by Rönnbäck et al. in 2010 [1]. As an agile information modeling technique, AM offers non-destructive extensibility mechanisms, thereby enabling robust and flexible management of changes. From the perspective of the traditional Entity Relationship Model (ERM), the time as a meta dimension/information is attached to the attributes and relations (ties in AM). With the time dimension, the whole history of the changing data is recorded in the database. AM decouples the evolution and application of a database, which when building a data warehouse enables shrinking of the initial project scope.

In their later works, the rules for translating an anchor schema into a relational database schema are presented [7]. The anchor schema in AM is presented in XML [8]. And an online visual modeling tool based on JavaScript is available in their website (www.anchormodeling.com).

In 2012, Knowles [9] evaluates AM as a conceptual design method in which to capture temporal data after exploring the concepts of temporal data, 6NF conceptual database models, and their relationship with Data Warehouse 2.0.

In 2015, Golov and Rönnbäck [10] introduce a novel technique, highly normalized Big Data using Anchor modeling, which provides a very efficient way to store information and utilize resources, thereby providing ad-hoc querying with high performance for the first time in massively parallel processing databases.

However, Jovanovic and Bojicic point out that AM recognizes (implicitly) only Time, thus severely limiting its usefulness for complex environments [5]. And software developers always need a set of tool chain before they can apply AM in their projects.

3. Extending Anchor Modeling

Suppose AM is happened to be applied in a simple barcode-based goods management application (BGMA for short). Two Anchors (goods, person) and two Ties (person possess goods, goods contains goods) are discovered in this case as shown in Figure 1. Ignoring the difference between icons in Figure 1 and those in Rönnbäck et al. [1], the BGMA can be easily described in AM.

Now we want to record the authentication information when a person possessing a goods. Let the authentication information be a string encrypted by the person's private key (e.g. in Public-Key Code System on RSA [11]), and the plain content looks like "Person X possesses Goods Y at 2017/01/01". The holder generates a barcode with the cipher content, and the BGMA scans it and stores the result information. The first question is how to describe the authentication information in AM?

3.1. Meta-Demensions Except for Time

Obviously, the authentication information belongs to the Tie PersonPossessGoods in Figure 1. However, in the 18 definitions of AM given by Rönnbäck et al. [1], there's no rule to describe it. And in some particular situations, the historized attributes can be changed if and only if there are some authentications. Such kind of information cannot be recorded in AM.

The reason for those is that AM only entitles one meta-dimension i.e. time to the Attribute and Tie, as shown in Figure 2.

The facts tell a more complex story, there is more meta-information for Tie, Attribute and even Anchor in practice. This pushes us to extend AM.

The intuitive idea is to regard the TIME TYPE in Figure 2 as a sub class of the META DATA TYPE in Figure 3. And the ANCHOR, ATTRIBUTE and TIE all have (0..*) relations to the META DATA TYPE.

With the extended meta model for AM in Figure 3, it is straightforward to update the 18 definitions in AM with the following definitions for the meta data type and its tuple.

Definition A (Meta Data Type). Let $\mathbb{M}$ be a meta data type. The domain of $\mathbb{M}$ is a set of data values.

Definition B (Meta Data Type Tuple). Let $\mathbb{M}^n = \underbrace{\mathbb{M}_1 \times \ldots \times \mathbb{M}_n}_{1\ldots n}$, $\mathbb{M}^n$ is a tuple of meta data types. When n is 0, $\mathbb{M}^0$ means that there is no meta data type.

The following is the extended 6th definition about Static Attribute.

Definition 6E (Extension of Static Attribute). A static attribute B_S is a string. A static attribute B_S has an anchor A for domain, a data type $\mathbb{D}$ for range, and a meta data type tuple $\mathbb{M}^n$ as meta ranges. An extension of a static attribute B_S is a relation over $\mathbb{I} \times \mathbb{D} \times \mathbb{M}^n$.

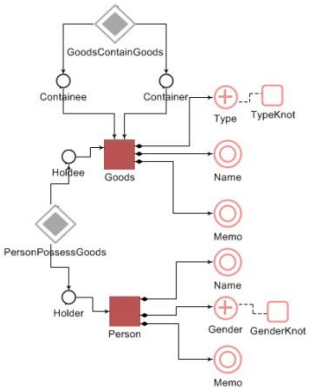


Fig. 1: Anchors and Ties in BGMA

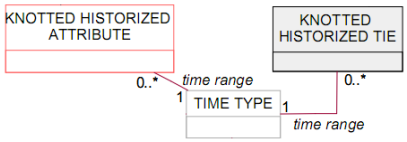


Fig. 2: Time for Attribute and Tie

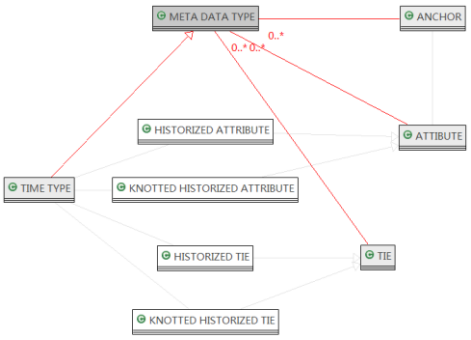


Fig. 3: Extending AM in Meta Model

Other definitions can be updated in the same way. The affected definitions are Definition 4 (Anchor), Definition 7 (Historized Attribute), Definition 8 (Knotted Static Attribute), Definition 9 (Knotted Historized Attribute), Definition 12 (Static Tie), Definition 13 (Historized Tie), Definition 14 (Knotted Static Tie), and Definition 15 (Knotted Historized Tie).

After extending AM with free meta-dimensions except for time, we can describe the authentication information in the BGMA as shown in Figure 4.

3.2. ERM Compatible Elements

There is a gap between storing and utilizing data in relational database according to AM [7]. From the perspective of ERM, AM splits big entity table and relation table into several small anchor, tie and attribute tables (subtable for short) so that meta-dimensions such as time can be recorded. However, nothing has been done to facilitate accessing these trivial subtables.

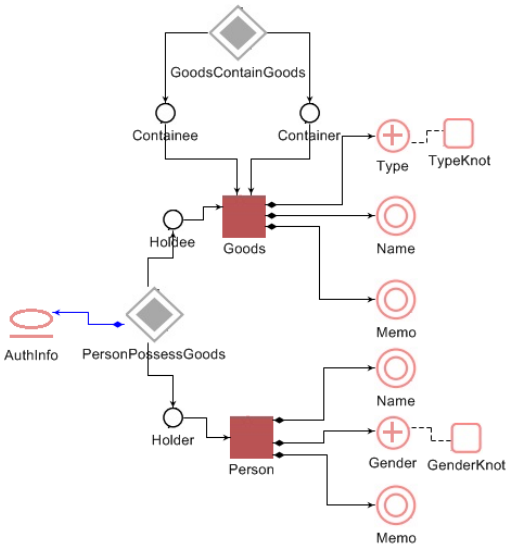


Fig. 4: TIE with Authentication Information

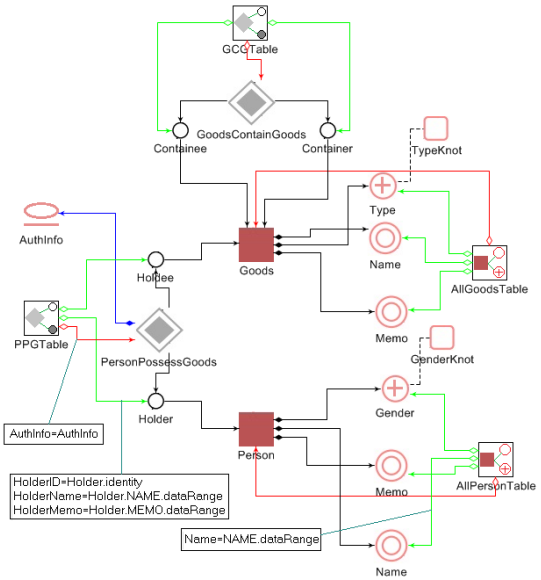


Fig. 5: AnchorEntity and TieEntity in BGMA

Since ERM is the most popular way to design and implement database applications, this section tries to introduce two elements AnchorEntity and TieEntity to meet the needs.

AnchorEntity can be regarded as ERM Entity in AM. It is aggregation of Attributes of an Anchor. The difference between AnchorEntity and Entity is that the former can be the snapshot of the historic data in anytime instead of the current. A variant of AnchorEntity is the AttributeEntity, where only one attribute

(and its meta information) is cared and the records between the concerned begin time to end time are extracted.

TieEntity is more complex since it is the join of the tie and its anchor roles and knot roles. And the attributes of anchors are also joined.

Both AnchorEntity and TieEntity are not essential and necessary elements in Anchor Modeling, thus the following only tries to explain how they work.

In Figure 5, two AnchorEntities (AllGoodsTable and AllPersonTable) and two TieEntities (PPGTable and GCGTable) are defined. For the AnchorEntity AllPersonTable, it aggregates name, gender and memo attributes of the anchor Person. For the name attribute, the field dataRange is selected as the field Name in the view, which is generated by SELECT and JOIN clauses in SQL. For the TieEntity PPGTable, it aggregates the Holder (the anchor role who refers to the anchor Person including the anchor's identity, attribute name's dataRange, and attribute memo's dataRange), the Holdee and the tie PersonPossessGoods (which refers to the meta information AuthInfo).

Put simply, suppose there are <#p01> in anchor Person, <#p01, "tom", 2000-01-01> in Person_Name, <#p01, "person tom", 2000-01-01> in Person_Memo, …, and <#g02> in anchor Goods, <#g02, "pen", 2000-02-02> in Goods_Name, …, and <#p01, #g02, 2000-03-03, "Tom-p01 has Pen-g02"> in tie PersonPossessGoods. There should be <#p01, "tom", "person tom", #g02, "pen", …, "Tom-p01 has Pen-g02"> in the TieEntity PPGTable and <#p01, "tom", "person tom", … > in the AnchorEntity AllPersonTable.

4. Tool Chain for Anchor Modeling

By extending Anchor Modeling, data objects can be expressed in the BGMA theoretically. However, more practical works are supposed to be done before BGMA runs in computers.

4.1. GME Based Modeling Tool

First of all, the model in Figure 5 should be converted into XML schema if we do have a graphical modeling tool for the extended AM.

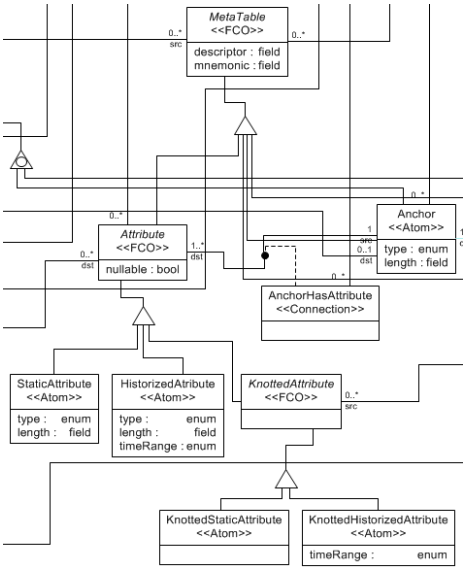


Fig. 6: GME AM Meta Model (part)

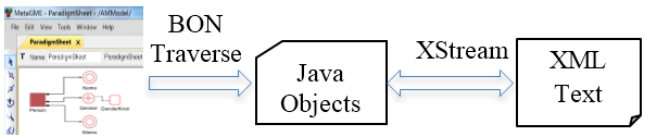


Fig. 7: Export GME Model to XML Schema

The AM website provides a JavaScript graphical modeling tool. But we cannot add other meta information other than time, and AnchorEntity and TieEntity is not supported.

This section introduces a visual model editor for the extended AM. It is based on the Generic Modeling Environment (GME) of Institute for Software Integrated Systems at Vanderbilt University. GME is a configurable toolkit for creating domain-specific modeling and program synthesis environments. It allows users to define new modeling languages using UML based meta models, and generates the corresponding visual modeling environment.

Figure 6 shows the meta model for AM in GME, which is similar to those in Figure 3. With this meta model, GME can generate the required modeling environment. And the model in Figure 1, Figure 4 and Figure 5 is drawn by the generated visual editor.

Furthermore, a GME plugin is required to transform the graphic elements in the editor to schema in XML format. To make it simple, the elements in visual editor is traversed by the Builder Object Network (BON) interface in GME, and the corresponding java objects are generated at the same time. Later, the java objects are serialized into XML text by the XStream library. Figure 7 gives the process. No SQL clauses are generated in Figure 7, for the database operations will be performed in later sections.

4.2. Data Access Interfaces

With the visual modeling tool, we can get the schema as in Figure 8. However, the tables in database are not ready and interfaces are required to access data in subtables and AnchorEntity/TieEntity tables.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema id="1" idlength="0">
  <knot id="5" mnemonic="GENDERLib" descriptor="-">
    <metainfo id="6" name="identity" description="identi
    <metainfo id="7" name="dataRange" description="dataR
  </knot>
  <anchor id="8" mnemonic="PERSON" descriptor="-">
    <metainfo id="9" name="identity" description="identi
    <attribute id="10" mnemonic="NAME" descriptor="-">
      <metainfo id="12" name="dataRange" description="da
      <metainfo id="13" name="timeRange" description="ti
    </attribute>
  </anchor>
  <tie id="40" mnemonic="PossessRelation" descriptor="-"
    <metainfo id="41" name="timeRange" description="time
    <metainfo id="42" name="AuthInfo" description="authe
    <anchorRole id="43" role="Holder" identifier="false"
      <type reference="8"/>
    </anchorRole>
  </tie>
  <anchor-entity id="45" mnemonic="AllPerson" descriptor
    <anchor reference="8"/>
    <time class="string">-1</time>
    <use-attribute id="50" filter="">
      <attribute reference="10"/>
      <column id="51" from="dataRange" name="Name"/>
    </use-attribute>
```

Fig. 8: GME Generated Schema for BGMA (part)

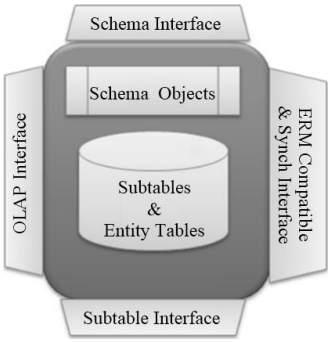


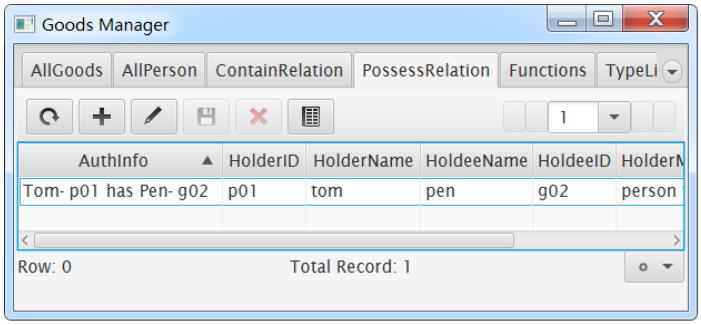
Fig. 9: Interfaces for Data Accessing in AM

Four kinds of interfaces are necessary for AM database applications as shown in Figure 9. The schema interface enables BGMA to access all meta-level information (such as anchors and ties of the schema) in the underlying database. The subtable interface is used to access data in subtables (described by Rönnbäck et al. [7]) in a way similar to the Hibernate framework. ERM compatible and synchronous interface is used to access data in AnchorEntity and TieEntity tables, and synchronize data in AM tables and ERM legacy tables. The OLAP [12] interface can be implemented by ETL tools for OLAP purposes.

It should be noted that schema in Figure 8 should be read and processed by the AM infrastructure. The infrastructure checks the corresponding subtables and entity tables, generating the missed tables and completing missed fields. That is to say, schemas for different applications can be merged in runtime.

4.3. GUI for ERM Compatible Tables

With data access interfaces in the AM infrastructure, tables are created according to the schema and records in them can be accessed. The last step is to display data and make it editable for users in the GUI.




```
<ErmTieTableControl ctx:id="tblPersonPossessMap"
    agileEditing="true" bank="am" entity="PersonPossessMap"
    prefWidth="-1.0" VBox.vgrow="ALWAYS">
    <columns>
        <TextColumn prefWidth="150.0" propertyName="AuthInfo"
            required="true" text="AuthInfo" />
        <TextColumn prefWidth="150.0" propertyName="HolderID"
            required="true" text="HolderID" />
        <TextColumn prefWidth="150.0" propertyName="HolderName"
            required="true" text="HolderName" />
        <TextColumn prefWidth="150.0" propertyName="HolderMemo"
            required="true" text="HolderMemo" />
        <TextColumn prefWidth="150.0" propertyName="HoldeeID"
            required="true" text="HoldeeID" />
        <TextColumn prefWidth="150.0" propertyName="HoldeeName"
            required="true" text="HoldeeName" />
        <TextColumn prefWidth="150.0" propertyName="HoldeeMemo"
            required="true" text="HoldeeMemo" />
    </columns>
</ErmTieTableControl>
```

Fig. 10: Graphic Data Browse & Edit Components

By extending TiwulFX [13], which provides JavaFX custom components specially designed to work with java POJO object, the data in entity tables can be browsed and edited as shown in Figure 10. Figure 10

shows the simple GUI of the BGMA, and the corresponding configuration of the component ErmTieTableControl, which will navigate data in the TieEntity (namely PersonPossessMap in Figure 10).

5. Conclusions

This paper extends the Anchor Modeling and implements necessary tool chain for designing and implementing of applications based on AM. Free meta-dimensions except for the time dimension are introduced to improve the descriptive capability of AM. Another visual modeling tool based on the generic modeling environment is presented by building the meta-model of AM. Three types of access interfaces for AM database are implemented on the Java platform. A graphic user interface for data creating, retrieving, updating and deleting is provided for AM tables. And a simple application is used to demonstrate the above works on AM.

The further work includes: 1) completing and optimizing the database access interfaces; 2) performance optimization for entity tables.

6. Acknowledgements

This research was supported by the College Education Project 2014YY024 sponsored by Military Training Department, PLA Genearl Staff.

7. References

[1] L. RöNnbäCk, O. Regardt, M. Bergholtz, P. Johannesson, and P. Wohed, "Anchor modeling — Agile information modeling in evolving data environments," Data & Knowledge Engineering, vol. 69, pp. 1229-1253, 2010.

[2] P. P. Chen, "A Preliminary Framework for Entity-Relationship Models," in Entity-Relationship Approach To Information Modeling and Analysis, Proceedings of the Second International Conference on the Entity-Relationship Approach, 1981, pp. 19-28.

[3] P. S. Chen, "The entity-relationship model - A basis for the enterprise view of data," 1977, pp. 77-84.

[4] P. S. Chen, "The entity-relationship model—toward a unified view of data," Readings in Artificial Intelligence & Databases, vol. 1, pp. 98-111, 1977.

[5] V. Jovanovic and I. Bojicic, "CONCEPTUAL DATA VAULT MODEL," 2012.

[6] A. Ledeczi, M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, et al., "The Generic Modeling Environment," Journal of Environmental Psychology, vol. 18, pp. 75-83, 2001.

[7] L. Rönnbäck, O. Regardt, M. Bergholtz, P. Johannesson, and P. Wohed, "From Anchor Model to Relational Database," 2010.

[8] L. Rönnbäck, O. Regardt, M. Bergholtz, P. Johannesson, and P. Wohed, "From Anchor Model to XML," 2010.

[9] C. Knowles, "6NF Conceptual Models and Data Warehousing 2.0," 2012.

[10] N. Golov and L. Rönnbäck, Big Data Normalization for Massively Parallel Processing Databases: Springer International Publishing, 2015.

[11] R. Amzad and R. Amzad, "Unbreakable Codes: RSA Public Key," 2014.

[12] A. Cuzzocrea, "Data warehousing and OLAP over Big Data: A survey of the state-of-the-art, open problems and future challenges," International Journal of Business Process Integration & Management, vol. 7, pp. 372-377, 2015.

[13] P. Indonesia. TiwulFx 2. Available: http://tiwulfx.panemu.com/