5F−8

Graphical Multi-Precision Dynamic Schema Design Interface

Hui Yao Michael Björn Hyeonkon Kim Ryosuke Hotaka
University of Tsukuba

1. Introduction

Applications, such as those in the multimedia environments, usually have object classes which have nested structures. In our experience, using graphical description and a graphical schema design tool is an efficient way to define and manage complex object classes/schemata.

In this paper, we present a graphical multiple precision dynamic schema design tool called SchemaBuilder, which has been implemented using Prograph [4]. We expect that our work will help users to handle complicated applications and improve the uniqueness and generality of the results of schema designs.

2. Motivation

In order to handle complex objects/schemata, an object-oriented data model called A Data Modeling Facility: JDMF-M92 (JDMF-M92 for short hereafter) [1] allows an attribute of a class to have an arbitrarily complex domain class. To support the definitions and management of such complex structures, graphical descriptions/graphical design tools are efficient.

3. Graphical data diagrams

3. 1 Schema definition in Bachman Diagram

The Bachman Diagram (B-D for short hereafter) uses a rectangle to represent a class and an arrow to represent a reference. Fig.1 shows a student registration management schema using B-Ds.

As shown in Fig.1, the schema description in B-Ds is very simple and straightforward.

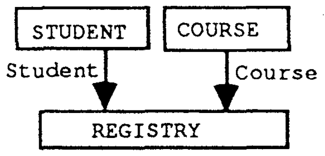

Fig.1 An example of Bachman Diagrams

3. 2 Generalization/Specialization—an extension to the Bachman Diagram

Although the B-D is very suitable for rough schema definition, it lacks semantic precision. An extension to the B-D has been made by R. Hotaka [2], adding the generalization/specialization relationship.

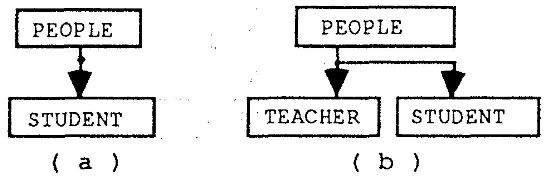

Fig.2 Examples of Extended B-Ds

Fig.2 (a) shows that STUDENT is a subclass of PEOPEL using a dotted arrow, and (b) shows that STUDENT and TEACHER are exclusive subclasses of PEOPLE.

3. 3 Semantic Diagram

Since the above graphical data diagrams lack the expressive power to describe detailed class information (especially for cases in which the domain class of an attribute of a class has an arbitrarily nested structure.), a new data diagram called a Semantic Diagram (S-D for short hereafter) had been proposed by R. Hotaka and M. Björn [3] and, for the first time, implemented here.

Fig.3 shows a simplified movie_sample management schema (assuming that a movie has only one track) [6]. In Fig.3, MOVIE and SAMPLE are NamedObject classes (a NamedObject in JDMF-M92 is managed by a MOKey, similar to a primary key in relational systems). Class MOVIE refers to class SAMPLE through a SetObject class called MEDIA. But class SAMPLE does not need to refer to class MOVIE.

The semantics of the above example can not be explicitly described using B-Ds.

Fig.4 is its counterpart designed using B-Ds lacking the necessary precision.

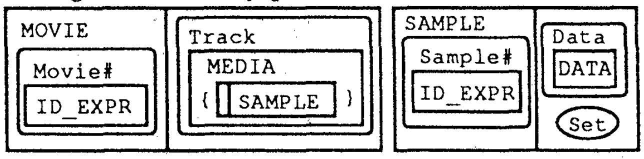

Fig.3 An example of Semantic Diagrams

In S-Ds, rectangles like MOVIE with vertical bar separating Movie# and Track represent NamedObject classes; Rectangles like MEDIA represent SetObject classes and the rectangle appears in the "{ }" area represents the component class of a SetObject class. Rectangles like DATA represent AtomicObjet classes. Round cornered rectangles like Track represent Attributes of AttributedObject class(es); an oval like Set represents a Method. A class within an attribute means it is the domain of the attribute.

MOVIE
SAMPLE

Fig.4 Descriptions of Fig. 3 using B-Ds.

4 SchemaBuilder—a user-friendly interface

4.1 Design purpose and general consideration

The prime goal of our work is to present users with a visual meta schema design tool which seamlessly integrates with the human design process. This is achieved mainly by implementing both the B-D and the S-D.

In SchemaBuilder, as shown in Fig.5 and Fig.6, multiple windows corresponding to the different abstract design levels are provided. At first, classes and references between classes are roughly designed in the B-D window. Then, the schema is translated into the S-D window and detailed information are added in classes. For screen space saving, an additional window can be opened for the definition of a complex domain class for any attribute.

4.2 System features

Some features of SchemaBuilder include:

1) Multiple views of a schema: Multiple abstract level views of a schema can be seen simultaneously. The user is relieved of the burden of browsing through pages of textual description;

2) Simplified schema editing: All schema editing (creation, display, update and deletion) can be done dynamically by "mouse-clicking". Other functions such as moving, draging, showdomain,··· , are also implemented.

3) Maintenance of consistency :

(i) Since multiple views of a schema exist, whenever a concrete definition is done in the S-D window, the necessary change was propagated to the B-D window;

(ii) Let X be a class. Since X's rectangle in a S-D window can stand for both the definition of or an reference to X at the same time, deleting X (other than NamedObject class) is generally done in one of two ways :

a) If there is only one appearance of X, then the selected appearance and X's definition are deleted (deletes its definition and a reference);

b) If there are more than one appearances of X, then only the selected appearance is deleted (deletes a reference only).

5. An example of schema design

Fig.6 shows the design result of Fig.3 produced from SchemaBuilder.

6. Further study

There are mainly two further study topics: one is window space saving; the other is performance speed. We are looking for a more efficient way to help users to manage schemata with large number of classes .

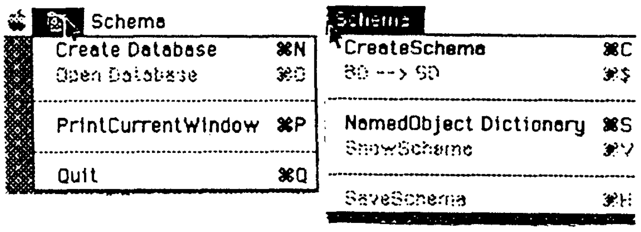

Fig.5 Menus of SchemaBuilder

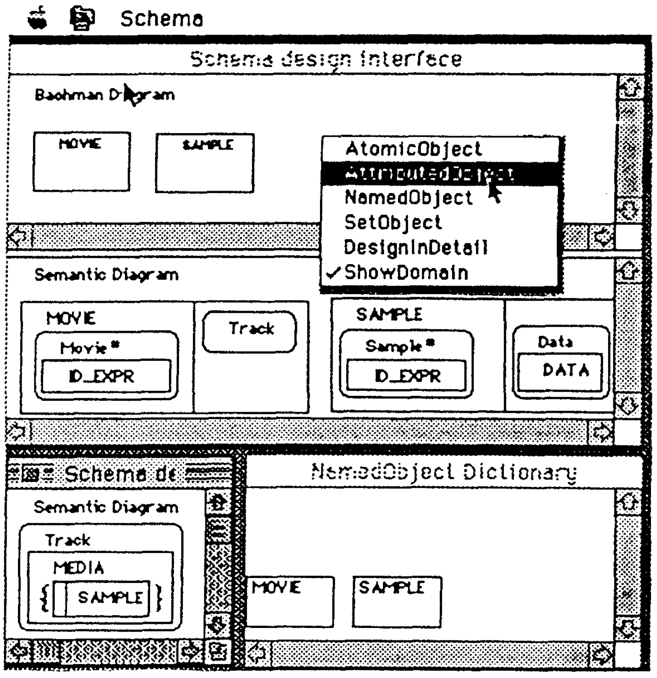

Fig.6 A design example from SchemaBuilder

References

[1] Japanese Standards Association: A Data Modeling Facility: JDMF/MODEL-1992, May 1993.
[2] Ryosuke Hotaka: Database system and data model, Ohm Publishing Co. Ltd, 1989.
[3] Ryosuke Hotaka, Michael Björn: Data Oriented Approach to Business Information Modeling, ICODP-93, 1993.
[4] TGS Systems Ltd.: Prograph Reference, TGS System Ltd., Canada,1990.
[5] Stephen Chernicoff: Macintosh™ Revealed, HAYDEN BOOKS, USA, 1987.
[6] Satoshi Tanaka, et al: An observation of muti-media extention of JDMF, IPSJ, 1994 spring.