Data Model for Customizing DB Schemas Based on Business Policies

Jun Sekine, Atsushi Kitai, Yoshihito Ooshima and Yasuhiro Oohara

NTT Information and Communication Systems Laboratories
1-1 Hikarino-oka, Yokosuka, Kanagawa, Japan.
E-mail: {sekine, kitai, oshima, oohara}@nttjog.isl.ntt.co.jp

Abstract. One of the goals of data modeling has been to uniquely identify the data semantics of target applications and describe them in database schemas. In contrast to this approach, this paper proposes a data model that allows description of variable parts in data semantics and how each variable part is dependent on business policies. This is useful when describing database schemas for common software components that are used in applications with different business policies. A set of alternatives chosen from a specified set of business policies is converted to a set of primitives that describes how variable parts should be customized, thus enabling users to generate a database schema meeting some specific application requirements. The primitives uniformly describe a variety of customizations, such as inclusion and exclusion of database elements, changes in the characteristics of a database element, and the composition of database elements. The model also serves as a basis of collecting and organizing a variety of business policies. Users can have a global view of how a database schema is affected by different business policy alternatives. A case study has shown that this model is applicable to real database schemas.

1 Introduction

Database design research has long been focusing on describing the data semantics of target applications, and a number of semantic data models [9, 21, 16, 1, 26] have been proposed. According to these proposals, one of the most important tasks in database design is to uniquely identify the data semantics of the target applications and describe them in database (or conceptual) schemas using a data model. This approach works well when the target applications are fixed and their requirements are clear enough. However, this is not necessarily ensured when describing a reusable database schema that can be used for a variety of target applications. We found two problems with this approach when trying to design a database schema for a common reusable software component used in different applications in the same business domain. The first was a divergence of the data semantics due to the divergence of the business policies behind each application. For example, we had difficulty in determining which relationship to define or what multiplicity a relationship should have because different applications have different business policies affecting some parts of the database schema or others.
The other problem is the satisfiability of business policies, meaning that it is difficult to know if a database schema meets the business requirements of an application. Specifications for database schemas may be useful for this purpose, but because they are often written in technical terms, translating them into business terms is often time-consuming. These problems become more important as reusable software components are becoming more popular with the emergence of plug-and-play components in the market.

Three approaches can be taken to resolve the first problem: designing a general database schema that covers all possible variations of business policies, preparing a different database schema for each different set of business policies, or customizing a single database schema based on business policies. The first approach is taken in conventional database design methodologies, especially in the area of schema integration [2, 29, 12]. However, the result of this approach could be redundant database schemas that are difficult to use and not necessarily optimal in terms of performance and storage size. The second approach is taken in proposals for storing specifications in a repository for reuse [7, 22, 11], but the cost of storing and organizing the repository is sometimes high. It is also difficult for users to have a global view of what is stored in the repository and how business policies affect database schemas. This paper takes the third approach, i.e. customization of a database schema, and proposes a data model called Business-Policy-driven Customization (BPC) model for it. The BPC model resolves the first problem by allowing description of variable parts in database schemas and also resolves the second problem by describing the relationships between the variable parts and business policies affecting them.

In the BPC model, each variable part in a database schema is uniformly described as a database policy, such as the existence of a database element, the existence of certain characteristics in a database element and the composition of two database elements, and each database policy has more than one alternative. The BPC model also allows description of business policies, such as whether corporate organizations are allowed as contractors or not, and each business policy has more than one alternative. Since each business policy alternative is related to one or more database policy alternatives, all the variable parts in a database schema are determined by choosing one or more business policy alternatives.

Using this construct, the BPC model allows the description of database schemas applicable to a wide range of applications. Users who are not necessarily experts in database design can give business policy alternatives necessary to customize the database schemas. In addition, the BPC model can be used as a basis of collecting and organizing a variety of business policies. Users can have a global view of business policies. The BPC model tries to describe database schemas that are as flexible and reusable as software components often called frameworks [30, 5, 28] or design patterns [14].

This paper is organized as follows. Section 2 discusses related works and their issues. In Section 3, the target data model generated through customization is briefly defined. In Section 4, details of the BPC model and the customization