Chapter 2.2
Database Design Based on B

Elvira Locuratolo
ISTI, Consiglio Nazionale delle Ricerche, Italy

ABSTRACT

This chapter is devoted to the integration of the ASSO features in B. ASSO is a database design methodology defined for achieving conceptual schema consistency, logical schema correctness, flexibility in reflecting the real-life changes on the schema and efficiency in accessing and storing information. B is an industrial formal method for specifying, designing, and coding software systems. Starting from a B specification of the data structures and of the transactions allowed on a database, two model transformations are designed: The resulting model, called Structured Database Schema, integrates static and dynamics exploiting the novel concepts of Class-Machine and Specialized Class-Machine. Formal details which must be specified if the conceptual model of ASSO is directly constructed in B are avoided; the costs of the consistency obligations are minimized. Class-Machines supported by semantic data models can be correctly linked with Class-Machines supported by object Models.

INTRODUCTION

The B Method (Abrial, 1996) is an industrial general-purpose formal method which uses a model, the *Abstract Machine*, to encapsulate a fragment of the state within a provably consistent specification and a refinement theory to derive correct programs. The direct use of B for developing database applications can lead to several advantages including the possibility of guaranteeing the correctness of the design process; however, it presents some shortcomings: The B Method lacks the abstraction mechanisms supported by the database conceptual languages, and its refinement has not been designed to obtain efficient database implementations. Specifying a database application with the B notation is a tedious process, since many properties implicitly declared within the database conceptual schemas must be explicated. Further, the consistency proofs are too expensive, since they must be performed with respect not only to the application constraints, but also to the conceptual schema constraints.

ASSO (Locuratolo, 1997, 2002, 2004; Locuratolo & Matthews, 1999a, b, c) is an innovative database design methodology defined for achieving conceptual schema consistency, logical schema correctness, flexibility in reflecting the real-life changes on the schema and efficiency in accessing and storing information. This makes it possible to overcome some inadequacies of existing informal methodologies (Batini, Ceri, & Navathe, 1992; Booch, 1994; Rumbaugh, Booch, & Jacobson, 1999) such as to guarantee the conceptual schema consistency and the logical schema correctness. Background information on ASSO can be found in initially disjointed approaches of work: A former approach aimed at establishing formal relationships between classes of objects based on semantic data models and classes of objects based on object models. The objective was to achieve the flexibility of semantic data models and the efficiency of the object-oriented database systems. This approach, called *Partitioning Method*, was proposed as a static method in 1992 (Locuratolo & Rabitti, 1998). A latter approach aimed at integrating features from conceptual modeling and abstract machines in order to guarantee the conceptual schema consistency (Castelli & Locuratolo, 1994). ASSO (Castelli & Locuratolo, 1995) tried to integrate these two approaches; however, the proposed model was not suitable to the Partitioning Method applicability. Approaches of study to design the conceptual model of ASSO, called *Structured Database Schema* and the *ASSO refinement* can be found in Andolina and Locuratolo (1997) and Locuratolo (1997). The Structured Database Schema integrates the specification of both structural and behavioral information at a high abstraction level. It extends the model on which the Partitioning works by designing elementary operations that add objects, remove objects, modify attributes, or let the class unchanged in order to still allow the Partitioning applicability. Approaches of translation from ASSO to B have been proposed in papers by Locuratolo and Matthews (1999, a, b, c). The approach employed to define ASSO, called MetaASSO has been described in Locuratolo (2002). The unitary element in the definition of ASSO is that of correct model transformation, as evidenced in Locuratolo (2004). Differently from a recent approach to the specification and development of database applications based on B (Mammar & Laleau, 2003), where the database application refinements have been implemented using the relational database model, in ASSO classes of objects supported by semantic data models and classes of objects supported by object systems are linked together.

This chapter aims at raising the abstraction level of B exploiting features of ASSO. Starting from a B specification of the data structures and of the transactions allowed on a database, two model transformations are designed. The former transformation, called *Database Schema Model*, restricts the state of the model supported by B in order to represent aspects which are typical of database applications. The latter transformation, called *Structured Database Schema*, reduces the possible B operations on the state of the Database Schema Model. The *Structured Database Schema* is a Conceptual/Semantic model based on a graph which integrates static and dynamics. The Structured Database Schema specifications are written using a formal notation which exploits the concepts of *Class-Machine* and *Specialized Class-Machine*, two concepts which enrich the corresponding concepts supported by the database conceptual languages with transactions and application constraints. In the Structured Database Schema specifications, many formal details are avoided with respect to the B specifications and only the state transformations, which satisfy the class and the specialization constraints, are allowed.

Two different forms of refinement are applied to a conceptual schema: *behavioral refinement* and *data refinement*. The behavioural refinement is defined by steps of B refinements which leave the state unchanged while modifying transactions,