NATIONAL INSTITUTE OF TECHNOLOGY RAIPUR

Security in Objecte Oriented Database Management System

Pratik Pandey
6th Semester Biomedical Department, Database Management System

Guided by :
Prof. Saurabh Gupta

Abstract

TABLE OF CONTENT

1 Introduction

The data model for a general-purpose object-oriented database system attempts to model the "real-world" as a collection of objects, where each object in the "real-world" corresponds to one database object. These database objects can have a complex internal structure composed of other database objects to model the details of the "real-world" objects to which they correspond. Database objects such as these with a complex internal structure are sometimes called composite objects. The internal structure of an object is implemented using instance variables. An instance variable is nothing more than a named slot inside an object that can contain a value. This value may be a primitive data value (such as an integer number or a string of characters) or a pointer (object identifier) to another object. (Some systems do not distinguish between these two types of values all data is an object, including primitive numbers and character strings.) If the value of an instance variable is a pointer to another object, this object may in turn have instance variables pointing to other objects, and so on, to describe the internal structure of the "real-world" object being modeled.

Several features of the object-oriented paradigm for database management make it attractive from a security point of view First, an initial layer of protection is provided by the fact that all data is stored as values for instance variables that are encapsulated inside objects and available only through the methods defined for the object's class. These methods can be used to enforce security requirements for the data in the objects using techniques similar to those used in Hydra and other capability-based and abstract data type-based systems. In addition, the enriched semantic modeling capabilities of the object paradigm should allow the "real-world" and its security requirements to be modeled more naturally in the database. Finally, inheritance, at least on the surface, looks like a useful tool for simplifying the specification of security requirements. The security requirements for an application can be defined for object classes near the top of the inheritance hierarchy, and inherited by all classes lower in the hierarchy.