The Adaptation Model of a Runtime Adaptable DBMS

Florian Irmert, Thomas Fischer, Frank Lauterwald, and Klaus Meyer-Wegener

Friedrich-Alexander University of Erlangen and Nuremberg,
Department of Computer Science,
Chair for Computer Science 6 (Data Management),
Martensstrasse 3,
91058 Erlangen, Germany
{florian.irmert, thomas.fischer, frank.lauterwald, kmw}@cs.fau.de

Abstract. Nowadays maintenance of database management systems (DBMSs) often requires offline operations for enhancement of functionality or security updates. This hampers the availability of the provided services and can cause undesirable implications. Therefore it is essential to minimize the downtime of DBMSs. We present the CoBRA DB (Component Based Runtime Adaptable DataBase) project that allows the adaptation and extension of a modular DBMS at runtime. In this paper we focus on the definition of an adaptation model describing the semantics of adaptation processes.

1 Introduction

In recent years the database community has realized that common database systems do not fit into every environment [6]. The obvious solution is the development of specialized DBMSs for each environment. However this approach is not suitable with respect to development cost, time to market and maintenance.

Tailor-made DBMSs [5] try to answer this challenge by adapting a DBMS towards a specific environment by providing a common code base from which customized DBMSs may be derived. Changing the functional range of such a DBMS however requires a shutdown and redeployment of a new version. Taking an application offline is often not feasible in some environments.

In the CoBRA DB (Component Based Runtime Adaptable DataBase) project [2] we propose an approach to tailor a DBMS at runtime. It uses a kind of DBMS “construction kit” and basic modules that are necessary in every DBMS. A DBMS can be assembled by choosing the appropriate modules for the intended functionality of the system. An important challenge is the modification of modular DBMSs at runtime. In our prototype it is possible to add, exchange, and remove modules while the database system is running. As a foundation we have developed an adaptation framework [3] that provides the exchange of components at runtime in a transparent and atomic operation. In this paper we present the adaptation model of CoBRA DB and its adaptation types.
CoBRA DB Runtime Environment

Runtime adaptation as a prerequisite to the proposed adaptable DBMS enables addition, removal and exchange of DBMS components at runtime without any downtime of the whole system.

To meet these requirements we have designed a runtime environment [3] based on a service-oriented component model [1,4]. The runtime environment is based on the OSGi Service Platform [4]. A component in the CoBRA runtime environment implements at least one service. The architecture of the CoBRA DB itself with its functional properties must therefore be sufficiently described by its services. To enable the adaptation to different environments, components implementing the same service can be exchanged at runtime. The resulting requirements for runtime adaptation are met by a transparent dynamic proxy concept which ensures the atomicity of the adaptation and the consistency of the state transfer from one component to the replacing one. Details can be found in [3].

Adaptation Model

Based on the prerequisites given by the runtime environment we developed an adaptation model in order to formalize the constraints for possible adaptations. This includes the structural model of the CoBRA DB architecture as well as the modeling of the different adaptation possibilities.

For the concrete specification of the adaptation model of the CoBRA DB we have chosen a service-oriented point of view. Figure 1 shows the structure of our model which consists of three design levels. The functionality of the DBMS is modeled by the specification of services (L3) like a PageService providing page-oriented access. The service descriptions of L3 must remain valid, even over adaptations. One or more services are embodied as components on L2, which are themselves implemented in one or more classes (L1). The components defined on L2 are the object of adaptation.

![Fig. 1. Model hierarchy](image-url)