www.run-software.com

ODABA

Compared with other ODBMS

run Software-Werkstatt GmbH
Köpenicker Str. 325
12555 Berlin

Tel.: (49) 30 - 6576 2791

Table of Contents

1Introduction....3

2Conceptual Particularities....4

2.1Object Model....4
2.1.1Key Concept....4
2.1.2Set Concept....4
2.1.3Indices....5
2.1.4Relationships....5
2.1.5Typed References....5
2.1.6Generic Attributes....5
2.1.7Derivations and Inheritance....6
2.1.8ActiveObjects....6
2.1.9Versions....6

2.2Dynamic Model....6
2.2.1Events and Reactions....6
2.2.2Scenarios....7

2.3Functional Model....7
2.3.1ODABA2 API....7
2.3.2Methods in Problem Classes 7
2.3.3Methods in Context Classes....7

3Tools....9

3.1Development Environment (ODE)....9

3.2Database Services....10

3.3Communication with other Systems....10
3.3.1ODBC-Interface....10
3.3.2Data Exchange....10
3.3.3Document Interface....10

1 Introduction

Here we will not demonstrate the possibilities of ODABA2, but more what ODABA2 distinguishes compared to other object-oriented database systems (ODBMS). Special for ODABA2 is the philosophy the system is based on. This is marked through two essential features – it's language independence and it's language base.

Language Independence

The ODABA2 - concept is based, similar to e.g. O2, on a database-oriented attempt. This differs from program language oriented attempts like POET, ObjectStore or Versant as it is not limited through the concepts of a programming language. Nevertheless, the possibilities of object oriented programming languages are supported by providing language related APIs (C++, JAVA is planned). The required class declarations are generated completely from the database schema definitions.

Language Base

The database concept of ODABA2 is based on the analysis of natural language descriptions. Thus the modeling rules of ODABA2 approach on one hand to the principles of natural language, whereby it is relative simple to understand despite of its complexity. Simultaneously advantage is taken from the basic principles of natural language presentations to provide extensive documentation and on-line help.

Recursive model

In contrast to other ODBMS ODABA2 is based on a recursive model, i.e. data and schema databases are following the same model, the same rules and access functionality, and differ only in their contents and their position to each other. The ODABA2 model itself is stored in a schema database, so that ODABA2 like the natural language is a self-describing system. This way the cornerstone is given not only for a continuous further development of the ODABA2 model, but application developers can make model extensions also.

The following paragraphs will give a short overview over the particularities, which result from this philosophy for the ODABA2 concept. Subsequently some special features of the ODABA2-tools will be presented.

More exact information to the features you can find in "ODABA2 - Technical Overview" or in "Real Objects" (published by the Addison-Wesley publishing house, 1996, German only).

2 Conceptual Particularities

Many OODBMS support the definition of the object model, only. ODABA2 is an ODBMS that supports all sub-models of the OO-model on the schema level. This way in ODABA2 not only the data model but also the dynamic and the functional model can be specified. Thereby the integrity on the schema level can be ensured on a high degree.

Each of the sub-models contains expansions, which exceed the known possibilities and compensate partially considerable defects of the OO-model.

2.1 Object Model

The ODABA2 object model is based on principles that result on one hand from the demands of the ODMG - standard (Object Database Management Group, 1997), on the other hand, however, from the principles natural language representations are based on. Together with the database oriented attempt a series of expansions results from this attempt that goes far beyond the possibilities of other ODBMSs.

2.1.1 Key Concept

ODABA2 is (as known so far) the only ODBMS that supports a consistent key concept. Several keys can be defined for each Structure, which can be referred to in different Collections in index definitions. Keys may exist of several components, which may contain also transient (derived) Properties.

ODABA2 guaranties (in contrast to other database systems like e.g. POET) the consistency of all indices when modifying attributes that participate in one or more keys. This can be suppressed as well by defining uncontrolled Collections.

The key concept of ODABA2 includes the definition of identifying keys (IdentKey). These are keys that provide (beside the instances Identity) an identifying function in the database and can be used for establishing Relationships between object instances. The key value is of importance, only, in the moment when establishing the Relationship and can be modified later, if necessary.

2.1.2 Set Concept

ODABA2 supports a far-reaching set concept that distinguishes between global and local Collections (sets). Local Collections are available only in the context of an instance. Global Collections (Extents) are generally available. Instances in Extents are identified beside their instance identity by their IdentKey values. Between Extents subset relations (derived

Extents) can be defined. Thus subsets can be defined as intersections of their basic sets or as union sets of their subsets. The consistency of set relations is maintained by ODABA2.

For each type one or more RootExtent can be defined that is the superset for all derived Extents. Moreover, the definition of free (uncontrolled) Extent is possible.

The ODABA2-API supports different set operations. Results of set operations are stored in temporary (transient) Extents or can be stored as persistent Collections as well.

2.1.3 Indices

Indices can be defined for each local or global Collection. Each index is based on one of the defined keys of the Structure that is defined as Collection Type. Indices can be ordered ascending or descending for individual key components. Moreover, individual key components can be marked as not case sensitive. Duplicate key values can be prevented as part of a Collection by marking an index as unique. The consistence of indices is maintained by ODABA2 in any case.

2.1.4 Relationships

Relationships can be defined as local or Extent-based (global) Relationships. Extent-based Relationships always refer to a subset of instances of the BaseExtent (the Extent the Relationship is based on). The consistence requirements resulting from that are maintained by ODABA2 as well, even in a multi-user environment, what is not the case for most of the other ODBMS (see e.g. POET). As far as an inverse Reference is defined Relationships are treated as controlled References (Collections). The maintenance of inverse References is performed automatically in any case (also in case of parallel updates for an instance) by ODABA2.

2.1.5 Typed References

Beside Typed Reference (all instances of a Reference have the same Type) ODABA2 supports also Weak and Untyped References. Instances in Weak Typed References are based on the same BaseStructure. Instances of arbitrary Type can be stored for Untyped References.

2.1.6 Generic Attributes

A particularity of ODABA2 is the concept of generic attributes. These allow the storing of attributes for different Types. Such generic attributes can be used, to store e.g. language dependent information for different languages (each language corresponds to a Type in this case) simultaneously in one attribute. Since generic attributes can be used also as sort key components, generic (language) dependent indices can be defined as well.

2.1.7 Derivations and Inheritance

ODABA2 supports not only multiple inheritance on the schema level but also multiple derivation on the instance level, i.e. several instances can be derived from one and the same base instance (shared base instances). This rather practical method is not supported by all the ODBMS that are conceptually based on C++ as (e.g. POET or ObjectStore).

2.1.8 ActiveObjects

The concept of ActiveObjects allows the definition of complex objects with it's own Extents. In this sense a ActiveObject represent a sub-database within a database from a semantic point of view. That allows the arrangement of databases in individual sub-bases (under test).

2.1.9 Versions

ODABA2 allows the creation of different versions for instances as well as for ActiveObjects or the entire database. When creating database versions, it becomes possible, to reactivate the state of the database according to an earlier version value.

The same way project versions can be defined on the schema level. This allows modifications or expansions in a new version without affecting older versions. Thus modifications become possible in structures and methods without reorganizing the application database.

2.2 Dynamic Model

The dynamic model of ODABA2 supports both, the definition of reactions on events as well as the definition of scenarios.

2.2.1 Events and Reactions

Events can be defined in ODABA2 on different levels. Thus Property, Structure, ActiveObject and Database events can be defined. In general an event signals a state transition of an observed object. ODABA2 supports both, system events like Read, Modify, Insert, Delete etc. as well as user defined events that are signaled, as soon as the defined state transition of the observed object has been detected.

It is possible to react on events with defined actions. Actions are methods that can be implements in different way, e.g. as function or expression or as dialogue or document.

2.2.2 Scenarios

ODABA2 supports the definition of complex processes by means of scenarios. Complex processes can be designed as simple sequential processes as well as sequences allowing parallel processing (up to now implemented as prototype, only).

2.3 Functional Model

The functional model of ODABA2 includes both the ODABA2 API as well as the specification of standardized methods like dialogues or documents. Besides the implementation of methods in a special context is possible via context classes.

2.3.1 ODABA2 API

The ODABA2 API provides a multitude of functions for C++ and ODABA2-OQL. This covers both, data definition language (DDL) as well as data manipulation language (DML). The ODABA2 API supports functionality on the instance (Structure) level as well as on the field (Property) level. The field level allows the access and the manipulation of field objects, which are associated not only with their data, but with their description and context, too.

2.3.2 Methods in Problem Classes

ODABA2 supports an extended functional model. Presently methods can be implemented in five ways

- as C++ - functions
- as MS Visual Basic functions
- as ODABA2 OQL expression
- as dialogues
- as document templates

Anticipated is the support of JAVA.

2.3.3 Methods in Context Classes

The definition of context classes allows the definition of special behavior in a certain context. Thus it becomes possible to design the special behavior for individual Properties. Context classes can be implemented for database objects (Property, Structure, ActiveObject, Database) as well as for GUI objects (Controls, Dialogue, Application, Project) or document objects. Methods in context classes can be implemented as C++ - functions or ODABA2 OQL expressions, but also as dialogue or document.

For context classes an expanded basic functionality is available, that opens extensive possibilities especially for GUI contexts.

3 Tools

The ODABA2 - tools include an integrated development environment (IDE) for specifying all three sub-models of the object model as well as different database services. Moreover data exchange facilities and project management features are provided.

3.1 Development Environment (ODE)

ODABA2 provides an IDE, which supports the development phases from the verbal specification of requirements over the design of graphical user interfaces up to the implementation of program functions and OQL expressions. It is possible to change to another development environment, however, at any time and in any development phase.

For the problem analysis expressed in natural language the **LexiconBuilder** is provided, that allows the generation of nearly 80% of the technical object model from the verbal problem description. But also in time providing document objects at any place in the IDE supports documentation and generating document objects from development objects as well.

ODABA2 provides a **StructureEditor** with extensive validation facilities for defining and checking the object model. With the **CausalityDesigner** the dynamic model can be defined. GUI methods (applications, dialogues) can be implemented with the **ProjectDesigner**. Document templates can be prepared under a text processing system (e.g. WinWord). Than they can be imported with the **DocumentComposer**. C++ -functions and ODABA2 OQL expressions can be implemented in the **ObjectDeveloper**.

All methods can be stored in the ODABA2 schema database (repository). Thus the consistence between external (e.g. C++-methods) and the ODABA2 Structure definitions can be maintained.

In addition ODABA2 supports document objects for the most development objects (classes, class members, function parameters, Dialogues, Controls, Menu Items,. ..). Especially supported is the principle "First document than implement", by generating implementation skeletons from the documentation objects.

From document objects a complete on-line help systems can be generated for the application as well as project documentation for different purposes.

3.2 Database Services

Database services provide several functions for data base maintenance. Distributed databases can be defined, database contents can be copied up to individual instances, and database statistics can be prepared as well as browsing or updating the database contents.

3.3 Communication with other Systems

ODABA2 databases can be accessed from other systems on one hand via the API functions. In addition there is, however, a variance of other communication possibilities.

3.3.1 ODBC-Interface

ODABA2 databases can be accessed via an ODBC interface that, however, allows a relational view on the ODABA2 instances, only. ODABA2 itself has access to all database systems that provide a corresponding ODBC-interface. From within the ODABA2 application external relations are looking like ODABA2-Extents. Thus from the application point of view it does not matter whether the data is stored in the ODABA2 database or in an external relational database.

For special database systems (e.g. MS SQL-Server) the ODBC interface is replaced by corresponding API functions, to improve the access efficiency.

3.3.2 Data Exchange

ODABA2 provides possibilities to define a partial or complete data exchange between ER-models and ODABA2 databases by field related assignments. Thus it becomes possible to import or export whole application databases without any loss of information.

Over and above ODABA2 provides different exchange formats as extended SDF (ESDF) or OEL (Object Exchange Language), which can be considered as de facto standards.

3.3.3 Document Interface

ODABA2 supports the generation different ASCII documents from simple text file up to RTF or HTML documents. Document generation is based on pre-defined document templates that allow similarly to serial letters the mixture of fixed texts and formatting information with variable database information. In contrast to simple text replacements in a document template, however, iterations or conditional expressions are supported as well.