EnviroInfo 2002 (Wien)
Environmental Communication in the Information Society - Proceedings of the 16th Conference
 ISBN: 3-9500036-7-3

Object-relational Database as a Data Source for Interactive Dynamic Web Application

Jadranka Pečar-Ilić[1], Ivica Ružić[1] and Zoran Skočir[2]

Abstract

A specialized Information system for the data of water quality monitoring for the Danube River Basin has been developed. The Information system enables efficient environmental water-related data management and its verification was performed using the data from the *Environmental Programme for the Danube River Basin (EPDRB)*, in which we actively participated (Pečar-Ilić/Ružić 2001). This paper brings out the most significant aspects of the *Object Oriented (OO)* modeling of the database for the surface water quality monitoring within the *EPDRB*. The database was implemented in the *Oracle® ORDBMS* and served as a data source for the interactive dynamic *Web* application for temporal and spatial presentation of relevant data. The use of *ORDBMS* as a database server in a *Web based three-tier client/server architecture*, such as integrated Information system architecture, has many advantages compared to the use of relational *DBMS*. The main concepts influencing the increase in *Web* application performances have been described.

1. Introduction

The specialized Information system for the data of water quality monitoring for the Danube River Basin was developed with the main goal to enable efficient environmental water-related data management. The development process of the specialized Information system was based on the defined architecture of the Integrated information system for temporal and spatial presentation of complex data (Pečar-Ilić 2001).

Efficient development of such complex information systems requires a modern approach. This comprises the following items (Pečar-Ilić 2001):

- Object-Oriented approach – *OO* analysis and design by applying *CASE* tools based on the *Unified Modeling Language (UML)* during object modeling.

[1] Ruđer Bošković Institute, Center for Marine and Environmental Research, Bijenička 54, HR-10000 Zagreb, Croatia, Tel. 00 385 (1) 456 11 40, Fax: 00 385 (1) 468 01 17; email: mailto:pecar@irb.hr, mailto:ruzic@irb.hr, http://www.irb.hr/

[2] University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000 Zagreb, Croatia Tel. 00 385 (1) 612 98 31, Fax: 00 385 (1) 612 96 16; email: mailto:zoran.skocir@fer.hr, http://www.fer.hr/

- Iterative and incremental development based on the system architecture model (serves as a framework for detailed development of the system structure and for iterative development of specialized information systems with similar characteristics).
- Application of modern technologies (software, tools and languages).
- *Web* as a platform for interactive dynamic applications.

The formal architecture description of the specialized Information system for the data of water quality monitoring for the Danube River Basin included additional class (object types) specializations in *UML* package diagrams, which correspond to spatial objects and the database source. They were necessary to describe the monitoring of the surface water quality in the Danube River Basin (Pečar-Ilić 2001).

Based on our own experience and the activities within the *Information Management Sub Group (IMESG)* of the *EPDRB* we analyzed the existing data management process and recognized the procedures that could significantly improve such data management (Ružić/Pečar-Ilić 2000). A *Data Exchange File Format (DEFF)* relational database implemented in *Paradox RDBMS* was used for the storage, processing and exchange of the collected data within the *EPDRB* (Kuipers 1996). For this reason, we started the analysis with *DEFF*.

2. Object-oriented modeling using *UML*

Use of *CASE* tools that support *UML* as a standard notation for object-oriented development methods was crucial for efficient object-oriented modeling. Nevertheless, although the *UML* supports a large number of existing *OO* methods and *OO* programming languages, it does not represent the standardization of the *OO* development process itself (OMG 2001).

Object-oriented modeling is performed using the concepts and some diagram techniques of the standard *UML*, supported by the available *Oracle® Object Database Designer (ODD) CASE* tool. The *UML* class diagram is used for formal description of a complex problem, i.e., creation of the water quality monitoring object model. *ODD CASE* tool is based on most of the *UML concepts* supporting the *reverse engineering* method and the *iterative* development process (Oracle® 1999a). However, there are some differences between the names of the concepts applied (Oracle® 1999a) and the original names from the *UML* standard (OMG 2001). For example, in the *ODD*, the type diagram corresponds to the *UML* class diagram.

By applying the *reverse engineering* method with the *ODD CASE* tool, the following procedures were performed (Pečar-Ilić 2001):

- Connection through *ODBC* to the existing *DEFF* system.
- Capturing the data structure from the *DEFF* database schema.

- Generation of the object type model (corresponds to the *UML* class diagram) as well as the server model based on the captured database schema.
- Tuning and improving the produced models.

In Fig. 1, the type diagram (i.e., *UML* class diagram) containing object types, and their mutual associations necessary for the description of the static data structure of surface water quality monitoring, is presented (Pečar-Ilić 2001).

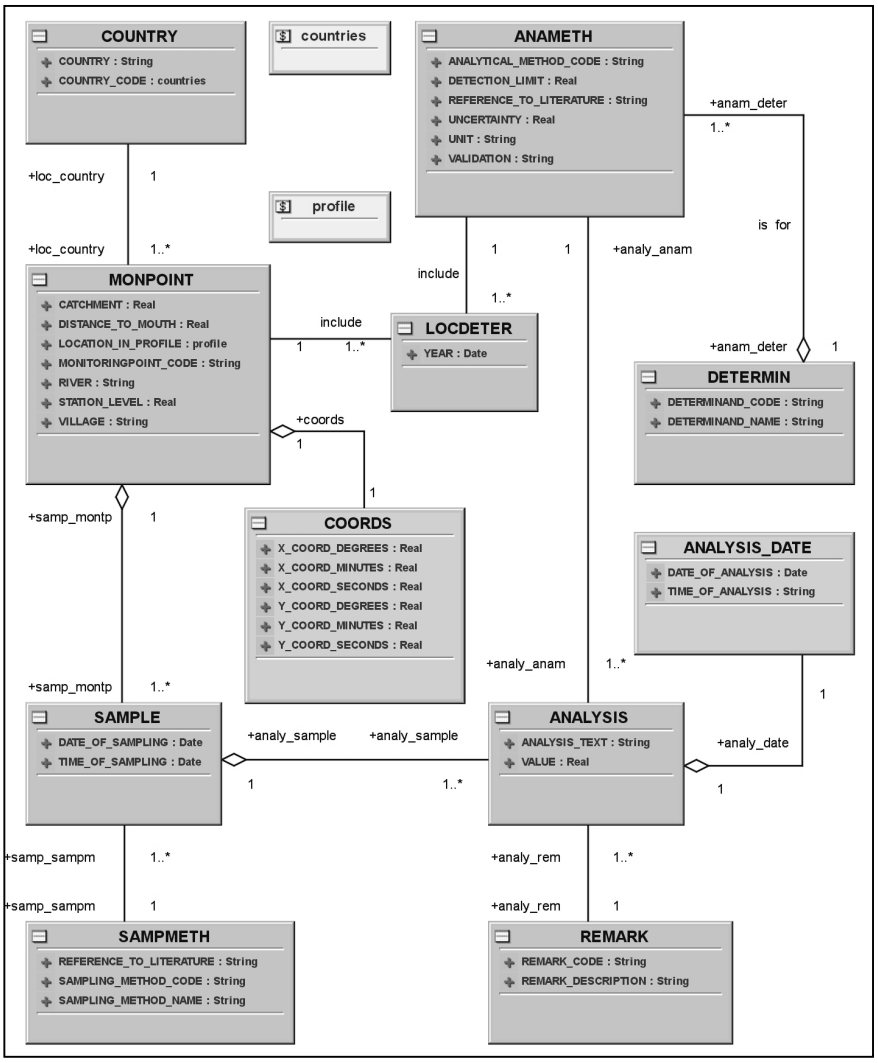

Fig. 1: *UML* class diagram for surface water quality monitoring in the Danube River Basin, presented using the *ODD CASE* tool

In this diagram, different types of associations are presented according to the well-known *UML* specifications for class diagrams (Oracle® 1999a). For example, weak aggregation (describing a weak or whole-part relationship) has a diamond on the comprising object type. In Fig. 1, weak aggregation between *MONPOINT* and *SAMPLE* object types describes that the sample has to be taken from a particular monitoring point, and for its unique identification, the monitoring point code should be part of its identifier.

Operations associated with the classes (object types in *ODD CASE* tool) represent the services that may be expected from these classes. For the object types shown in Fig. 1, the attributes have been defined but not the operations. Namely, this results from the server model (produced in *ODD* by transforming the object type model) implementation in *Oracle® ORDBMS* as a relational design with *Oracle 8* object views (Pečar-Ilić 2001). Object views represent a concept that provides an object-oriented view on hidden relational or object tables through *Oracle 8* object types (Oracle® 1999b).

3. Object model implementation in the *ORDBMS*

Depending on particular vendors, *ORDBMSs* more or less remain object-oriented systems. They support both the relational and the object aspects of the real world, and in that context they represent a hybrid between *RDBMSs* and *OODBMSs*, providing a combination of object-oriented and relational data structures (Stonebraker, Brown 1999). Nevertheless, there is no universal object-relational data model as yet while there are many variations of similar models whose characteristics depend on the level and the nature of performed variations.

For example, the *Oracle® 8 ORDBMS* realizes its object-relational model in two different ways, by using *Oracle 8* objects or by using object views (Oracle® 1999b). Views that we selected enable object-oriented views on relational tables. This can be achieved because object views can use particularly defined *Oracle 8* object types for accessing hidden relational tables. *Oracle 8* object types include: *object identification (OID)*, a unique name, as well as attributes and methods for modeling their behavior.

The design of the above-mentioned elements was realized using the *ODD CASE* tool. *ODD* enables transformation of an object types model (corresponds to the *UML* class diagram) to a server model based on a relational design with *Oracle 8* views. The server model obtained includes relational tables and the necessary *Oracle 8* object types. Such a server model diagram is presented in Fig. 2.

The Oracle 8 object types are connected with two different types of associations:

- *REF* (reference) – one object refers to the other through its attribute serving as a reference (for example *SAMPLE_T* refers to *MONPOINT_T*, which is represented in Fig. 2 as a void diamond on the side of the referencing object).

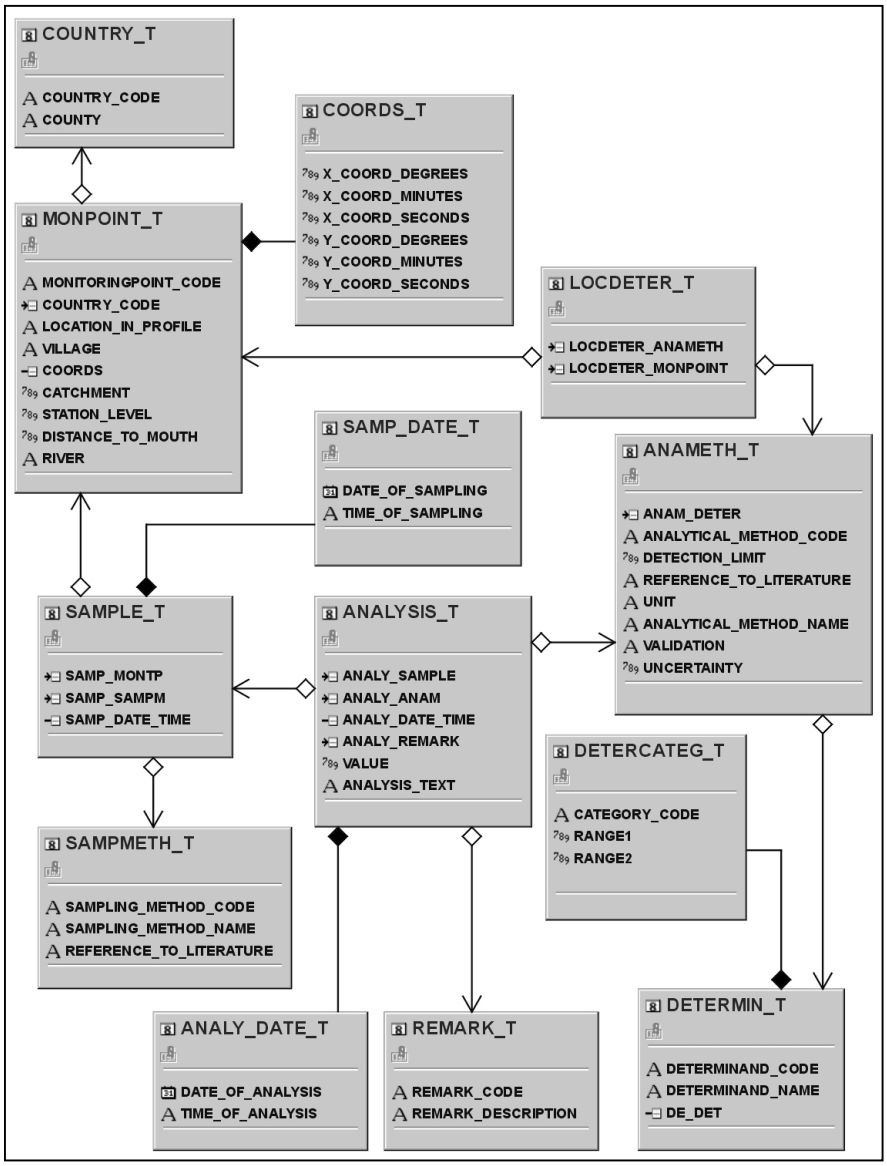

Fig. 2: Server model produced by transformation of an *UML* class diagram in the *ODD CASE* tool

- *Embedded* – one object is incorporated within the other object (for example *MONPOINT_T* incorporates the *COORDS_T*, which is represented in Fig. 2 as a filled diamond on the side of the incorporating object).

Object views are created on the basis of the defined *Oracle 8* object types. Their names correspond to object type names where an additional letter "V" is added, which include the object view name, *Oracle 8* object type name on which it is based, object *OID* based on a unique key for hidden data and the *SELECT* command for taking over data from the hidden object or relational tables. On the basis of a server model defined using the *ODD CASE* tool, *SQL Data Definition Language (DDL)* commands are produced for creation of *Oracle 8* object views over relational tables. After that, a user's schema is generated in a database instance on the database server where *Oracle® 8 Enterprise Suite* is installed. In Fig. 3, some objects in the *deffdb* user's schema of the database are presented. For example, this *deffdb* user's schema contains *Oracle 8* object types, object views, packages, stored procedures and relational tables. *Oracle 8* object type names have the extension '_T', while object views have the extension '_T_V' (Oracle® 1999b).

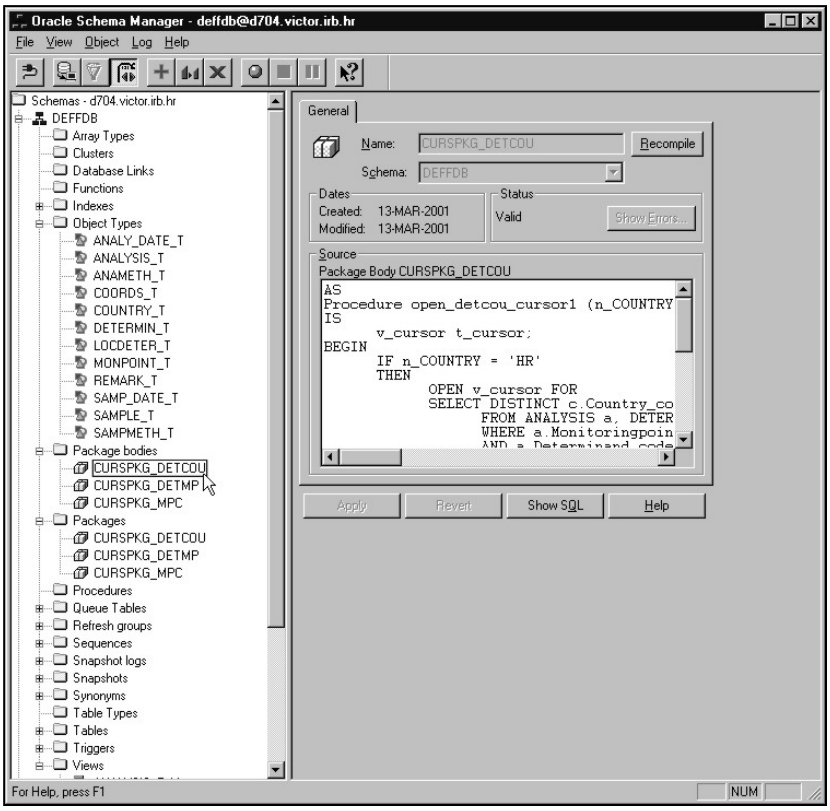

Fig. 3: Example of the *CURSPKG_DETCOU* package body content in the *deffdb* user's database schema presented by the *Oracle Schema Manager*

4. Increase of *Web* application performance by using *Oracle 8* packages

Regarding the database design, the main advantages of *Oracle® 8 ORDBMS* are the reusability of objects for creating new complex objects (by building them in the table columns or rows) and well-structured design (by using objects, the database architecture based on components can be created). In that context, stored procedures and functions can be defined to provide protected visibility of attributes (i.e., object structure) and to defined object behavior (i.e., operations). Such stored procedures, functions and other subroutines can be encapsulated into *Oracle 8* packages. We can reuse such packages in many applications to perform different tasks since we know exactly their basic functions. At the same time, we do not know anything about their data structure since it is encapsulated (Oracle® 1999b).

The developed *Web* application provides timely availability of relevant information for privileged *EPDRB* participants (Pečar-Ilić/Ružić 2001). This application enables automatic generation of various report types (such as maps, time series diagrams, tables, etc.), as well as generation of thematic maps based on built-in knowledge and user queries. To incorporate an automatic categorization of average parameter values, as additional knowledge, and to provide a better performance of the dynamic *Web* application, *Oracle 8* packages containing stored procedures were defined. Packages and stored procedures represent stored subprograms, which is the concept that allows *Oracle®* to enable reuse of a defined and tested program code (Oracle® 1999b). Automatic reporting was provided through defined *ASP (Active Server Pages)* containing *ADO (ActiveX Data Object)* and *VBScript*. For example, the application response time was substantially improved when the package for querying a particular parameter value for a defined time period from all *TNMN* monitoring stations was used.

Nevertheless, the *Oracle® 8 ORDBMS* has some disadvantages that have to be mentioned. Its main disadvantages seem to be identical with the general disadvantages of similar object-relational *DBMS* technologies. For example, the increase in complexity is caused by the support of both the relational and object-oriented systems. Another problem is the database objects migration from one *ORDBMS* vendor to another, which is possible only if the transformation from the *OO* to relational design is performed, since the *OO* design requires a completely new database architecture (Oracle® 1999b).

5. Conclusions

The most significant tasks in the *OO* development of the database for the surface water quality monitoring within the *EPDRB* are presented. Object-oriented modeling was performed using the concepts and some diagram techniques of the standard *Uni-*

fied Modeling Language (UML), supported by the available *Oracle® Object Database Designer (ODD)* CASE tool. The database was implemented in *Oracle® Object-Relational DataBase Management System (ORDBMS)* and served as a data source for the interactive dynamic *Web* application. The dynamic *Web* application was developed to enable spatial and temporal presentations of relevant data by selecting the objects from a digital map (Pečar-Ilić/Ružić 2001).

The use of *Oracle 8* packages and stored procedures had a substantial influence on the increase of the *Web* application performance. We applied these concepts while building in the automatic categorization of the average parameter values procedure, as additional knowledge in the database, necessary during the automatic generation of thematic maps. The *Oracle 8* packages with stored procedures were also used in *Active Server Pages (ASP)* scripts since they represent concepts for decreasing the number of calls between the client, *Web* server and database server. These combinations were crucial in the cases where large amounts of data were transmitted through the network by the application (Pečar-Ilić/Ružić 2001).

Bibliography

Pečar-Ilić, J., Ružić I. (2001): Dynamic Web Application for Temporal and Spatial Presentation of Environmental Data, 15th International Symposium Informatics for Environmental Protection - Sustainability in the Information Society, Part 1: Impacts and Applications, Zurich 2001, pp. 431-436.

Pečar-Ilić, J. (2001): Information system for temporal and spatial presentation of complex data, Ph.D. thesis, Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb (Croatia), pp. 193.

Ružić, I., Pečar-Ilić, J. (2000): Integration of GIS and Relational Database for the Trans-National Monitoring Network in the Danube River Basin, MTM III – International Workshop on Information for Sustainable Water Management, September 25-28, 2000, Nunspeet (The Netherlands), pp. 457-465.

Kuipers, T. (1996): DEFF User Manual - EPDRB, TNO Institute of Applied Geo-science, Delft, the Netherlands, TNO-report GI-96/456, June 1996, pp. 33.

OMG, Inc., (2000): Object Management Group Unified Modeling Language Specification, Version 1.3, March 2000.

Oracle®, (1999a): ODD Online Help, from Object Database Designer, Release 6.0, 1999.

Stonebraker, M., Brown, P., (1999): Object-Relational DBMSs - Tracking the Next Great Wave, Morgan Kaufmann Publishers, Inc., 1999.

Oracle®, (1999b): Oracle8i™ Enterprise Edition - Getting Started, Release 8.1.5 for Windows NT, Oracle®, March 1999.