Communications of the Association for Information Systems

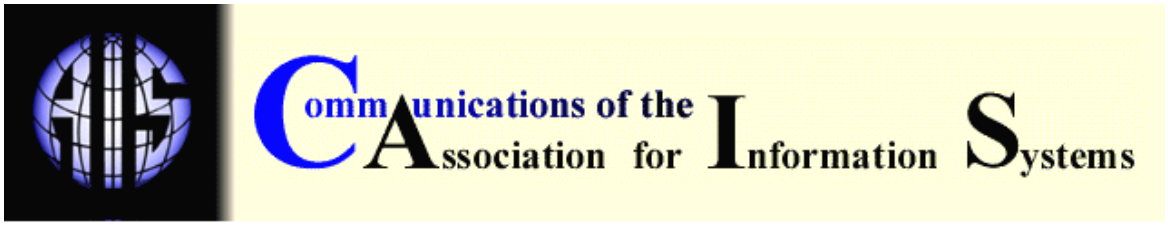
Volume 8 Article 12

February 2002

Special Issue on the AMCIS 2001 Workshops: Demonstrating the Database Client Environment through Oracle Developer

Mary Ann Robbert
Bentley College, mrobbert@bentley.edu

Recommended Citation
Robbert, Mary Ann (2002) "Special Issue on the AMCIS 2001 Workshops: Demonstrating the Database Client Environment through Oracle Developer," *Communications of the Association for Information Systems*: Vol. 8 , Article 12.
DOI: 10.17705/1CAIS.00812
Available at: https://aisel.aisnet.org/cais/vol8/iss1/12



SPECIAL ISSUE ON THE AMCIS 2001 WORKSHOPS: DEMONSTRATING THE DATABASE CLIENT ENVIRONMENT THROUGH ORACLE DEVELOPER

Mary Ann Robbert
Computer Information Systems
Bentley College
mrobbert@bentley.edu

ABSTRACT

This paper reports on a workshop held at AMCIS2001 to address configuration and pedagogical issues with Oracle tools as discussed by database faculty at prior conferences. This workshop demonstrates simple configurations for off-campus access to a database server, Oracle Developer basics and projects that have worked well with my courses.

KEYWORDS: Oracle, Oracle Developer, client/server, database, forms, student projects

I. INTRODUCTION

At recent conferences such as the ACM Special Interest Group on Computer Science Education and previous Americas Conference on Information Systems, database faculty raised the issue that Oracle is difficult to administer and complex to learn. The Morrison survey noted similar findings [Morrison and Morrison, 2001]. Though the Oracle Academic Initiative offers a great software package that includes most of the current Oracle development tools, technical support is limited. These products are sufficiently complex to require assistance, and there is a steep learning curve for students. The workshop attendees discussed solutions to these issues, as well as other issues experienced in using Oracle as a foundation for the database course.

This paper introduces Oracle Developer and examines its role in the database class. Information on an optimal set up and configuration for Oracle Developer (/2000, 6 and Forms and Reports 6i) is included together with an introduction to Developer's capabilities and degree of difficulty. It is assumed that the reader has some knowledge of Oracle, but not necessarily Oracle Developer, and is teaching or planning to teach a database course.

Oracle Developer can be used as an aid to attaining course goals. Included is a discussion of fit, time commitment, expectations of results, and team projects suitable for a semester course. The expected results detailed are based on ten years use of Oracle in the classroom, formal and informal discussions with other faculty, and the hands-on demonstrations from the workshop. The Access database from Microsoft is used as a basis for comparison with other databases.

II. SUPPORTING THE DATABASE COURSE

A database management system can be integrated into a database course to assist students in achieving the course goals. If the objectives of your database course include:

- students will know about databases,
- know how to use databases,
- understand database design and implementation,

then Oracle and Oracle Developer can be used effectively. Knowledge of concepts can be enhanced and reinforced through tool use. Students will learn a fully functional tool that is a market leader. The gap between theory and reality can be illustrated through the tool.

Oracle 8i can be configured to provide support for database courses through its ability to present concepts beyond those provided by lower level database products such as Access. Constructing a multi-level architecture with Oracle on the server and Developer on the client, exemplifies the levels of the ANSI/Sparc architecture. Concepts such as data independence, integrity control and security placement are naturally viewed in context. Allowing students access to their own and shared tables on the server via the Internet empowers students, enables them to work off campus, and gives them a feel for real world solutions.

If Oracle is the database used to solve problems and demonstrate theoretical underpinnings of databases for a course, then the environment provided by Oracle Developer is a natural to demonstrate client capabilities and application development. Oracle Developer makes it possible for students to gain a better understanding of front-end tool capabilities through actual problem solving. Students implement some of these database concepts covered in the course to solve a problem through application development.

The primary concepts I demonstrate through application development include:

- data independence
- security
- integrity
- data models
- referential integrity enforcement
- concurrency control

Other topics such as portability and performance can be included.

The client/server configuration begins to demonstrate data independence because the tables remain resident on the server, with data varying as applications are developed. Students model what goes on the front-end vs. the back-end and how the legacy systems are integrated, then test the model through the tool execution. Accessing data simultaneously, multiple users show the concurrency controls available through Oracle. This work should lead to a discussion of locking, returning relevant error messages, and adding additional controls. Constructing forms on clients shows the dangers inherent in multiple users changing front-end applications. Placement of integrity constraints can be tested and data entry through views can be demonstrated. Most importantly, students are taught to focus on the problem, not the solution. Defining the problem first, then analyzing causes and possible solutions before designing an application is still crucial. Concepts should arise naturally as the problem is being examined and problem resolutions proposed.

III.CONFIGURING THE ENVIRONMENT

Oracle Developer can be installed in a two or three-tier client/server architecture that is transparent to the students as they build their applications. The simplest solution is the two-tier system with Oracle8i on the server and Oracle Developer on the client connected through ODBC drivers. Once the connection is established, any ODBC compliant front-end tool or language can be used for application development. Comparisons of different front-end tools are discussed in Section VI.

I have been using Oracle Developer 6 on Windows 98, interfacing with Oracle8.1.5 on the server through the Internet. Though our configuration is minimum, NT with SP4 running on a Hi Q Pentium Pro at 200 megahertz with 128 megabytes of RAM and two 3 gigabyte hard drives, we have had no problems satisfying the needs of up to seventy students per semester for four years. Note this machine is placed outside the firewall to allow full access through the Internet.

Developer 6, distributed for student use by the Oracle Academic Initiatives program, is installed on all our lab computers. Students can also install Developer, which contains the Oracle 8i client, on their own computers and configure their environment following an instruction set and inserting server definition code posted on my web page,
http://cis.bentley.edu/mrobbert/cs652x.htm.

When installing the client, it is necessary to define the server in the tnsnames file and verify the sqlnet.ora file. Problems that have arisen with client installation include:

- home computers with less that 128 mb of RAM (will sometimes run with 64 but not well),
- insufficient processor speed,
- not following setup instructions (the largest cause of failure), or
- not removing a partial installation before reinstalling.

I have seen students with three tnsnames files and only one has the correct connection string. A live Internet service is also necessary. Cable works best. I've had no success with AOL because of time out and slow connection speeds. Students trying to connect from work encounter difficulty with the company's firewall. There is some instability with the installation. Sometimes it is necessary to run the installation program two or three times after which it installs fine. I tested Developer on Windows 2000 and found no obvious problems provided Windows NT is checked as the operating system during installation.

The configurations for Developer 6 match those for Developer/2000. Thus a common interface and application development platform can be achieved for students using Windows 95, 98 and 2000. If students work from home on their own machines, consistency among various versions of Windows is essential.

The new 12 CD tool set distributed by Oracle Academic Initiative this year includes Oracle Forms 6i and Reports 6i as part of the Internet Developer Suite. The tool suite, functionally similar to Developer, contains a set of integrated builders and wizards that help developers to construct highly interactive forms and charts with minimal effort. However, the extensible Java client offers developers vast potential to improve web applications with packaged samples. This enables publishing forms on the web and writing Java classes that are wrapped with PL/SQL code.

IV.ORACLE DEVELOPER

This section examines the development environment provided by Oracle Developer, i.e. the framework and the set of tools used to develop applications. All examples are based on Developer 6, which followed Developer/2000 and is comparable.

DEVELOPER COMPONENTS

The three major components of Oracle Developer are (1) Forms, (2) Reports and (3) Graphics. I will focus on Forms since it is the most complex module and exemplifies best the concepts being taught in a database class. The Forms component is interactive and includes the framework to develop menus and PL/SQL library modules. Once students can develop forms, the reports follow in a similar manner and the graphics are derived from the data.

FORMS

The major components (not the complete set) of the Form Module are the (1) blocks, (2) canvas-views, (3) windows (4) list of values (LOV) and (5) triggers.

Blocks. Blocks are the intermediate building unit containing a collection of records. The two kinds of blocks are self-explanatory, base-table and control blocks. Base table blocks

correspond to a table or view and provide an interface for record input or query-by-example. The control block represents a collection of single valued objects that do not correspond to a table. The blocks can be created through the data block wizard. Note that blocks cannot be integrated on the screen and items cannot be grouped within a block.

Canvas. Once a data block is created, the layout wizard is automatically called to determine the layout style – Form, one record displayed or Tabular, multiple records. A canvas is the result. The canvas contains frames which display the requested attributes and can be modified for ease of use. The canvas-view is the background on which the items are displayed and the canvas-view itself is displayed in a window.

LOV. An LOV can be associated with most items. It is easiest to create a base table containing the values that will be displayed. A user-friendly wizard steps the designer through the process of creating the table, inserting the values, and attaching it to the field. PL/SQL code must be written to assure proper performance.

Triggers. Triggers can be built-in or user-defined. The set of built-in triggers is sufficiently extensive that it is unnecessary to design new triggers. Again, the code to dictate the exact operations is written in PL/SQL. In fact, PL/SQL is required to create buttons, combo boxes and other desired features, and is a prerequisite for designing forms.

GRAPHICS

The graphics component contains the display modules, charts, graphs and other visuals that may be derived from the database or independent sources. SQL queries are used to generate the data. Data in the exact format required for the graph must be provided. This can be done with the group by and sum command. If time or skills' constraints do not permit code creation, sample code for generating graphs can be distributed for modification and experimentation. The results from this module are flashy and easy to construct. Most students will learn the basics on their own in order to enhance their project.

V. USING DEVELOPER

Once required forms are defined for a model, Oracle Developer Form Builder should be opened. The first data block in Module 1 is created by invoking the data block wizard. A relation is a special object used by the form to define a master detail structure. Students can create required input forms as individual canvases in a single module or in separate frames within a canvas.

WORKSHOP EXAMPLE

Workshop attendees were granted access to a set of server tables representing a simple order system. Using the tables, *customer, order, order lines, invoice, and products*, attendees attempted to develop forms with constrained items and menu navigation.

The wizards available in Developer make form construction obvious. Within fifteen minutes, master-detail forms were created. Participants commented on the ease of use and seamless connectivity. These basic forms created contained all the required fields but no enhancements or hints, and were similar to many student forms. They were non-functional – the server tables when accessed through the forms did not accept data. In Oracle Developer, the wizards and drag and drop features make form construction deceptively simple and form execution extremely frustrating.

To construct functional forms, the base tables must be known and understood. Constraints placed on the base tables must be indicated or enforced on the form. Character lengths and field restrictions defined on the attributes are not automatically reflected in the form fields. The designer must indicate the attributes' properties. The field properties table is the best place to start enforcing server side constraints by filling in all specified properties. Then, error messages and help lines should be added to the frame. The amount of upfront planning required was noted.

Referential integrity constraints, such as foreign keys must be addressed. In the workshop's orders case, a customer must exist prior to placing an order. When trying to insert the order, an error message, "parent not found", was generated. Adding an LOV to the customer

name field on the order form provides a list of valid customers. A navigation button can be added to transport to the new customer input form. Participants noted that the tasks are simple for students with programming skills.

This exercise simulated students' first attempts to use Oracle Developer. An hour of tool instruction was given and graduate students were available for help. Participants were able to create the basic form and began changing properties and adding LOV's. Participants appreciated Developer's interface and gained an understanding of its capability. Some decided the tool would enhance their class while others felt the instruction required would be too time consuming.

At the conclusion a sample form was distributed. A graduate student produced this form in four weeks beginning with no prior Developer knowledge and doing the assignment along with a full course load. This completed form is meant to be a measure of what can be expected rather than a perfectly constructed form[1].

VI. ENHANCING A DATABASE COURSE

STUDENT PROJECTS

The problems selected for students' solution must be complex enough to warrant developing front-end applications and simple enough to be completed in the semester. Developer's application generator and report writer simplifies development of the applications. The same PL/SQL used in the creation and querying of the database ties together all the needed Developer objects. Built-in subprograms allow manipulation of objects in various components enabling students to create projects with more functionality.

Projects that have worked well with Oracle and have fit within the confines of a semester course include a hospital, an auto dealership, a museum, a real estate office, and a landscaping firm. In the hospital example, a basic hospital is defined with patients, doctors, and staff. A patient is admitted to a room and incurs charges for the room, doctor visits, treatments, and extras. From this common point, the project is customized each semester. For example, a team can develop the admission system, creating means to determine empty beds, defining admission forms and generating bills and occupancy reports. Another team can develop a staffing system to assure twenty-four hour coverage of doctors and nurses and staffing for the laboratories at appropriate hours. I have even assigned a hospital payroll project. If time permits, I have the teams integrate the projects into a single working system, sharing the data and accessing the forms and reports through a common Developer menu. For example, the hours an employee works in the staffing system should be sent to the payroll system. The difficulty with this last segment of the assignment is team completion of their portion of the project. The team that finishes its segment first is always impatient waiting to synchronize with the next group.

FACULTY PREPARATION

Oracle Developer should be used to attain course goals and not as an add-on. The overhead on the faculty and the students is too great. Faculty must know the tool sufficiently to answer basic questions, but the students can learn the enhancements independently. I only use one hour of class time to introduce the tool, explain the concepts that can be enforced with this tool, and show the students how to logon and access the wizards. The major problem encountered is the instability of the Developer environment. Applications will run perfectly but an hour later the same application won't run, then it will resume running in the evening. It appears to be a software problem but we have been unable to determine a pattern.

In the workshop, all twenty-four faculty members were able to set up a basic form and retrieve data into the form. Only half succeeded in inserting records into the base tables because of server-side constraints. Addressing ALL the details is tedious and time consuming. A displayed message defining format or an attached LOV is required for each field. Participants felt more time was needed to implement a functional form.

---

[1] A copy of the output disk can be obtained by contacting the author at mrobbert@bentley.edu

If a faculty member is familiar with Oracle but has never used Developer, a course from Oracle is beneficial. Oracle offers seven courses on differing levels ranging in duration from one to four days. An alternative is an on-line course with capability to have questions answered. Sources of on-line courses include:

Sys_Ed (http://www.sysed.com/09Oracl/ORAC02.html)
Element K (http://www.elementk.com/), and
Element K through the University of California, San Diego Extension
(http://www.extension.ucsd.edu/Catalog/coursedeptlist.cfm?quarter=SU01&dept_id=8).

TEACHING STUDENTS DEVELOPER

Assuming students know PL/SQL, the only tool-specific skills students need to be taught in class to get Developer to "work" are how to configure the Oracle environment, the capabilities of the required modules, and how to access the wizards. Implementation of enhanced integrity constraints and security will require more instruction and can be expanded as desired. Laboratory assistants or graduate students, who have completed a project using Developer, can be of great assistance in aiding students with specific implementation problems.

I give my students their choice of front-end tools and most choose Access. They are highly successful generating reports but not always successful with forms. Since data cannot be inserted through all views, some base table configurations make establishing input forms impossible. Until last year (2000) no team spent the time required to learn Developer. Many tried and gave up because of the perceived complexity of the tool.

Sufficient supplementary materials are now available to shorten the learning curve. The "Quick Tour" included with Developer provides an overview of the concepts and components. Excellent on-line tutorials are available at:

http://www.prenhall.com/mcfadden/oracle/tutorial/lesson9/page1.html, and at
http://192.156.214.248/~dbase/oracle/dev2k/2.0/tutorial.html. These tutorials are complete and provide the answers to most student questions. Students are able to learn the tool within a week or two through the tutorials. The Starter Kit book [Muller, 1999] is well designed for self-teaching and includes a supplemental CD and a case. However, it does not provide the quick start of the tutorials. The handbook is still available [Muller 1996], but it is strictly a reference. The order of presentation does not make it useful as a tutorial.

Since Developer is a tool used to create the front-end for the project, all students are not equally committed to learning the tool. However, it is worth the effort for students. Those who have completed their projects with Developer have found the experience worthwhile.

VII. CONCLUSION

Oracle Developer glues together all the objects within and outside of the Oracle database students want to work with. The primary glue is the PL/SQL language. If Oracle SQLPLUS and PL/SQL are used for the database design, then the use of Oracle Developer is a natural extension that allows students to see the full environment. The overview of Oracle Developer in this paper is designed to assist in weighing the advantages of using this tool, to enable students to implement database concepts learned in class and, to solve complex problems.

EDITOR'S NOTE: This article is based on the author's workshop presented at AMCIS 2001. It was received on September 3, 2001 and was accepted on September 25, 2001. The article was with the author for approximately 3 weeks for one revision. This article was published on February 27, 2002 together with the other articles in the special issue on the MCIS 2001 Workshops.

REFERENCES

EDITOR'S NOTE: This article contains the address of World Wide Web pages. Readers who have the ability to access the Web directly from their computer or are reading the paper on the Web, can gain direct access to these references. Readers are warned, however, that
1. these links existed as of the date of publication but are not guaranteed to be working thereafter.
2. the contents of Web pages may change over time. Where version information is provided in the References, different versions may not contain the information or the conclusions referenced.
3. the authors of the Web pages, not CAIS, are responsible for the accuracy of their content.

4. the author of this article, not CAIS, is responsible for the accuracy of the URL and version information.

Morrison, J and M. Morrison (2001), "Using Oracle to Augment the Information Systems Curriculum," *Communication of AIS* (7)10, pp. 1-36.

Muller, R. (1996) *Oracle Developer/ 2000 Handbook,* Berkley, CA: Osborne/McGraw-Hill.

Muller, R. (1999) *Oracle Developer Starter Kit,* Berkley, CA: Osborne/McGraw-Hill.

ABOUT THE AUTHOR

Mary Ann Robbert is the chair of the Computer Information Systems Department at Bentley College in Waltham, MA. She completed her CSE PhD at University of Connecticut and worked as a NASA Summer Faculty Fellow for two years. Her research and consulting work focuses on database and data warehousing.

ISSN: 1529-3181