**DNR Business Reporting System, Version 2.0**

**Shawn Rivera and Matt Rykaczewski**

**CS 470 Final Writeup**

**April 29, 2009**

**Table of Contents**

**Abstract** 1

**1. Introduction** 1

**2. Project Overview** 1

*2.1 Application Data* 2

*2.2 Prior User Interface Design* 2

**3. Project Requirements** 3

*3.1 Functional Specifications* 3

*3.2 System Specifications* 4

**4. System Design** 4

*4.1 User Interface Design* 4

*4.2 Data Structures* 6

*4.3 System Architecture* 6

*4.4* *Database Structure* 6

**5. Software Development Process** 6

*5.1 Testing and Debugging* 7

*5.2 Prototyping* 7

*5.3 Work Breakdown* 8

**6. Results** 8

*6.1 Future Steps* 8

**7. Summary and Conclusions** 8

**Appendix A: List of Figures** 9

**Appendix B: User Guide** 22

**Appendix C: Code Listing** 23

**DNR Business Reporting System, Version 2.0**

**Shawn Rivera and Matt Rykaczewski**

# Abstract

The DNR Business Reporting System (DBRS) allows users to generate reports from the DNR Business databases. The reports offered via DBRS are web-based and are generated through interactive queries determined by the user. Many reports are linked to documents or other functional web pages maintained within the DNR Business System. Reports include options for saving reports in an MS-Excel format or a PDF document, and a flexible view front which allows users to easily navigate through records. Version 2.0 of DBRS adds additional functionality to the previous release and incorporates features requested by users.

**1.** Introduction

For our past CS 395 (Matt) and CS 495 (Shawn) internship projects, we collaborated on the development of a web-based reporting system for the Department of Natural Resources (DNR). Shawn was assigned the project in August 2007 and within a few weeks a functional prototype of the system, now known as the “DNR Business Reporting System (DBRS)”, was delivered. Once the concept was sold to the other managers, Matt was hired to help with development. From October 2007 to May 2008, we worked closely together to develop, enhance, debug, test and release a production version of the application.

While the application we developed met the needs of the users at the time, as with most projects the requirements changed as the needs of the users expanded. During development, testing and use of version 1.0 of DBRS, users began asking for new features. In order to meet deadlines, rather than adding these features incrementally we had to compile a list of enhancements to make at a later date. That list grew significantly, and as a result we were asked to develop version 2.0 of DBRS.

**2. Project Overview**

This project was primarily developed for Frank Wallis, our manager at DNR. All of the users of the system contact him with comments and suggestions, and he is responsible for determining what features to include in future releases. The users of DBRS include other DNR employees, as well as employees of other government agencies such as the Bureau of Land Management (BLM).

The goal of this project was to develop a new, more functional version of DBRS that incorporates most of the features users have requested.

## 2.1 Application Data

DBRS is a table-driven application. Rather than hard-coding information about reports in the application, these details are stored in a database that is accessed by the application. This approach decreases the amount of maintenance required, as changes to any of the reports can be made simply by modifying a set of tables, and also allows us to dynamically generate JSP pages.

The database used by DBRS previously consisted of two main tables (see **Appendix A - Figure 1**), as well as a number of views and materialized views that provide the data needed for the reports. The REPORT\_DRIVER and REPORT\_PARAMETERS tables provided all of the information needed for the reports; the REPORT\_DRIVER table held all of the reports found in DBRS and the REPORT\_PARAMETERS table held all of the fields users could search on for each of the reports.

*2.2 Prior User Interface Design*

Prior to the development of version 2.0 of DBRS, the majority of the application’s user interface consisted of the following pages:

* Index page
* Available Reports page
* Report Parameter page
* Report Display page

The index page (see **Appendix A - Figure 2**) is the first page users see when accessing DBRS. This page provides a description of the application as well as options for entering/logging into the system. Previously, DBRS could only be accessed within the DNR network; if a user tried to access the page without being connected to the DNR network it would not display. State employees could access DBRS by clicking on the “Log In As State Employee” link, which was somewhat misleading since the link simply opened a new window with a list of available reports.

The available reports page (see **Appendix A - Figure 3**) contains a list of all of the available reports in DBRS. The reports were arranged alphabetically by Category – Sub-Category – Report Name – Report ID. Selecting a category and/or sub-category would filter the reports and only list those that belong to the selected category and/or sub-category. The table displayed on the page included a list of the report names and descriptions. Clicking on any of the report names brought the user to the report parameter page for that report.

The report parameter page (see **Appendix A - Figure 4**) allows users to select from a list of input parameters that can be used to limit the results returned by the report. Once the user enters/selects the desired information, they can click the “Run Report” button to generate the desired report. Clicking the “Reset Form” button clears all of the selections that the user has made. Clicking the “Select Field Names” button opens the “Field Name Selector” window that allows users to limit the field names to be displayed on the report they generate. Clicking on the help icon ([Help](#fig2)) next to the report name at the top of the report parameter page opens a help page that explains how to use the features (list boxes, text boxes, etc.) found on the report parameter page.

The report display page (see **Appendix A - Figure 5**) is where the results of the report are displayed. The page consists of option buttons, a navigation toolbar and a table of results. The “Save to Excel” and “Save to PDF” option buttons allow users to save the results of the report to an Excel spreadsheet or a PDF document. Clicking the “Report Information” button opens the “Report Information” window that displays information about the report. The navigation toolbar allow users to easily navigate through the results, and includes options for jumping to any page or results, navigating from one page to the previous or next page and setting the number of rows that are displayed in each page of the report. The table below the navigation toolbar is where the results of the report are displayed. Previously, users could select any column heading that was colored blue to sort the results on that column name, and were limited to sorting on one column at a time.

**3. Project Requirements**

The requirements for this project were somewhat concrete, but since there wasn’t an exact product in mind we used the prototyping methodology and expressed the requirements in English. The same development strategy was also used for version 1.0 of DBRS and we decided to use it again since it worked well.

*3.1 Functional Specifications*

1. Access to the system must be restricted, and users outside DNR must be able to access the system.
2. All reports must be secure. There are four security types a report can have:

Public

The report is open to all users.

Group(s)

The report is open to one or more specific LDAP groups.

Example: “DNR All”

Individual(s)

The report is open to one or more individuals.

Example: “jhsmith”

Attribute(s)

The report is open to all users who have a certain attribute associated with their username.

Example: Employee works for DNR

1. The application should use the DBServices library instead of DatabaseTools.
2. Users should be informed when the system is processing a request (i.e. via a “please wait” message on the screen).
3. For mapping reports, users should be informed when the system is transferring control between DBRS and Alaska Mapper (i.e. via a “please wait” message on the screen).
4. Reports should have the ability to belong to multiple categories.
5. It should be possible to specify the ordering of the categories.
6. Users should be able to toggle if report descriptions are displayed on the available reports page.
7. Use regular expressions to “normalize” user-entered values for “MTR” text fields.
8. User-entered values for “MTR” text fields should only be accepted if they are valid.
9. Report column headings should have tooltip descriptions if they require further explanation.
10. It should be possible to specify which report columns users can sort on.

*3.2 System Specifications*

The end user must access the application over the internet using any JavaScript-enabled web browser (Internet Explorer, Netscape Navigator, Mozilla Firefox). The system is optimized for IE 6 or above and Netscape 7. Some functions may or may not work in other web browsers. The user should also have Adobe Reader installed on their computer.

**4. System Design**

All of the programming for this project was done in Java. This includes Java servlets, JavaServer Pages (JSP) technology and the Java Database Connectivity (JDBC) API. Scripting was done using JavaScript. Database access and modification was done using Structured Query Language (SQL). Dynamic web pages were created using Hypertext Markup Language (HTML), with embedded JSP syntax.

*4.1 User Interface Design*

This project required modifying most of the existing pages of the user interface and adding a new login page.

The index page (see **Appendix A - Figure 6**) is now accessible to users outside the DNR network. Public users can view a list of reports available to them by clicking on the “Enter Public Site” link. State of Alaska employees can log into the system by clicking on the “Log In as State Employee” link.

The new login page (see **Appendix A - Figure 7**) allows users to authenticate themselves using their State of Alaska Lightweight Directory Access Protocol (LDAP) username and password.

The available reports page (see **Appendix A - Figure 8**) now has two columns, one that conveniently shows all report categories and subcategories together in a hierarchical list and another that displays a list of available reports. Clicking on a category will update the list of reports that is displayed. By default, report descriptions are not displayed in the table of available reports. Clicking the “Show Descriptions” link will update the page and display the descriptions below each report name (see **Appendix A - Figure 9**).

The report parameter page (see **Appendix A - Figure 10**) provides the same functionality as before, but some modifications were made to increase usability. Previously, users could hover over a parameter name to view a description of the parameter, if one was available. Since there was no indication of which parameters had descriptions, some users were not aware of this feature. An information icon was added to resolve this issue and is displayed next to any parameter names that have a corresponding description. Users now view parameter descriptions by hovering over the information icon (see **Appendix A – Figure 11**).

The report display page (see **Appendix A - Figure 12**) also provides the same functionality as before, but some changes were made to meet the requirements of this project. Similar to report parameter descriptions, version 2.0 adds descriptions to report fields. An information icon is displayed below the name of any field that has a corresponding description, and users can hover over it to view field descriptions (see **Appendix A – Figure 13**). The sorting of the results has also been expanded and users are no longer limited to sorting on one column at a time. Clicking a column header sorts the results in ascending order (as indicated by a green triangle under the column header). Clicking a column header a second time sorts the results in descending order (as indicated by an upside-down green triangle under the column header). Clicking a column header a third time removes the sorting and corresponding green triangle. Previously, clicking a second column header would remove the original sort, but now if another column header is selected multiple sorts are applied (in the order the columns appear).

Version 2.0 also incorporates a “please wait” message (see **Appendix A - Figure 14**). The message is displayed whenever DBRS is processing a request. This feature was added because some operations (e.g. generating a report with a large number of records) can take a considerable amount of time and we wanted to prevent users from interfering with the system’s operation. There is also a similar message that is displayed when the system is transferring control between DBRS and Alaska Mapper.

*4.2 Data Structures*

The data structures used in version 2.0 of DBRS are simple objects and are primarily used to store the information retrieved from the database. The existing JSPRS and ReportDriver objects had to be modified and the ReportParameter objects underwent a much-needed refactoring. We also created two new objects, Category and ReportField, to store categories and report fields.

To meet the requirement of using the DBServices library instead of DatabaseTools, we also had to make the necessary changes to the application code.

*4.3 System Architecture*

The new structure of DBRS (see **Appendix A - Figure 15**) includes the various JSPs that make up the user interface, servlets/classes and the new Category and ReportField objects.

The Reports servlet is the main focus of DBRS. This servlet processes requests from dynamic web pages, modifies data and forwards to the appropriate JSP page. The JSPRS object is used to transport data between the servlet and JSP pages.

The ReportDriver object contains information needed for each report, such as the report name, report ID, description, etc. The ReportParameter object contains information for the parameters found on the report parameter page, such as the parameter name, flag to indicate if the parameter is required, type of parameter (text, number, date, spatial), etc. The ReportField object contains information for the fields found on the report display page, such as field name, description, flag to indicate if the field can be used for sorting, etc. The Category object contains information for the categories found on the available reports page, such a category name, description, ordering, etc.

DBRS also makes use of a ReportDBManager class and now uses the DBServices library. Using DBServices, the ReportDBManager class retrieves and passes data to the application from DNR’s Oracle databases.

* 1. *Database Structure*

The new DBRS database (see **Appendix A - Figure 16**) is composed of six tables, including REPORT\_DRIVER, REPORT\_PARAMETERS, REPORT\_FIELDS, REPORT\_CATEGORIES, REPORT\_SECURITY and CATEGORY.

**5. Software Development Process**

Since our client didn’t have an exact product in mind, we used the prototyping methodology. We implemented an initial prototype, showed it to our client, gathered new requirements, designed and implemented them, and repeated the process until it was satisfactory. For this project, the product we developed was accepted by our client after only two prototypes.

*5.1 Testing and Debugging*

We spent a large amount of time in testing and debugging. The first bug arose when we were implementing the new database design. The new design required a significant refactoring of our existing code, and in the process we introduced errors. When the system is initialized, the data from the database is retrieved and stored in the appropriate objects. If certain objects are not created successfully, the system will either fail to initialize or some reports will not run. A mistake in the SpatialParameter class prevented the system from creating parameters for mapping reports, and it took a substantial amount of time to track down and debug.

Another bug that took a considerable amount of time to debug was related to the “please wait” message. A pop-up is displayed when the system transfers control between DBRS and Alaska Mapper, and we noticed it didn’t always behave as expected. During our work on DBRS, other applications maintained by our division underwent domain name changes. Alaska Mapper uses the domain name dev.dnr.alaska.gov and DBRS uses the domain name devreports.dnr.alaska.gov. When we tried to communicate with a static HTML page, which displays the “please wait” message, Internet Explorer would lose communication between DBRS and Alaska Mapper. This was solved by replacing the static HTML page with a JSP placed under each domain name.

The last major bug we encountered occurred during implementation of the validation of “MTR” text fields. Neither of us has much experience working with regular expressions, and we had some difficulty getting the validation to work properly with certain input. We spent several hours testing various valid and invalid input values and were able to fix the bug by starting over and re-writing the code that performs the validation.

*5.2 Prototyping*

Fortunately, the requirements we started with for this project did not change. We used prototyping during development of the first version of DBRS and it also seemed to work very well for the development of version 2.0. Rather that implementing all of the requirements and then showing the prototype to our client, we chose to add features incrementally.

For the first prototype, we implemented the security requirement, converted the application to use DBServices instead of DatabaseTools and added the “please wait” message. Our client only asked for small cosmetic changes after we presented this prototype, and no major changes or additions had to be made to the requirements. The rest of the requirements for this project were implemented in the second and final prototype. Our client was very pleased with the end result and didn’t ask for any additional changes. Since we were aware of what our client was expecting, prototyping was fairly straightforward and we didn’t encounter any many challenges.

*5.3 Work Breakdown*

The Gantt chart (see **Appendix A - Figure 17**) displays the projected and actual work schedule. The values in red represent the estimated amount of time per task determined during the planning stage of the project, and the values in blue represent the actual amount of time taken. Although some tasks took longer than expected to complete, there aren’t any significant deviations from our overall projected and actual work schedule.

We originally intended on dividing the requirements and working on them individually, but our schedules permitted us to use pair programming for most of the development of the project. The one exception to this is in the beginning when we were implementing the new database design and developing the security tool. These tasks took longer than expected so Shawn completed implementation of the new database design while Matt worked on the security tool.

**6. Results**

Version 2.0 of DBRS was completed on-time and has been successfully deployed and used in DNR’s Development, Test and Production environments. Our client was very pleased with the product we delivered and all of the requirements were met. We have also received positive feedback from users of the system.

*6.1 Future Steps*

Although we haven’t been asked for any additional features yet, during the development of this project we attended an Advanced JavaScript class and would like to incorporate various uses of AJAX, such as updating pages without needing to refresh, in a future release. The error messages returned on the login page are not very user-friendly so we also plan on making modifications to this page to improve usability.

**7. Summary and Conclusions**

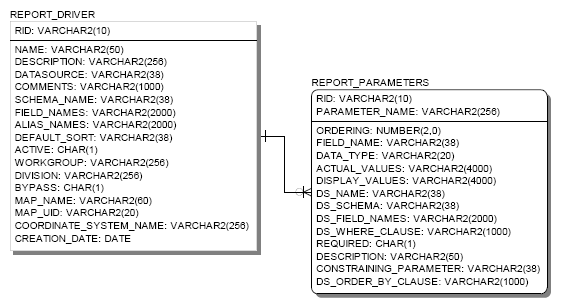
Version 2.0 of DBRS was developed with the goal of adding additional functionality to the previous release and incorporating features requested by users. The project was completed on-time and we were able to successfully implement all of the requested features.

Overall, this project was a great learning experience and allowed us to learn new aspects of coding with Java, JavaScript and SQL, as well as using security within applications. We also gained more experience working for a client and under time constraints.

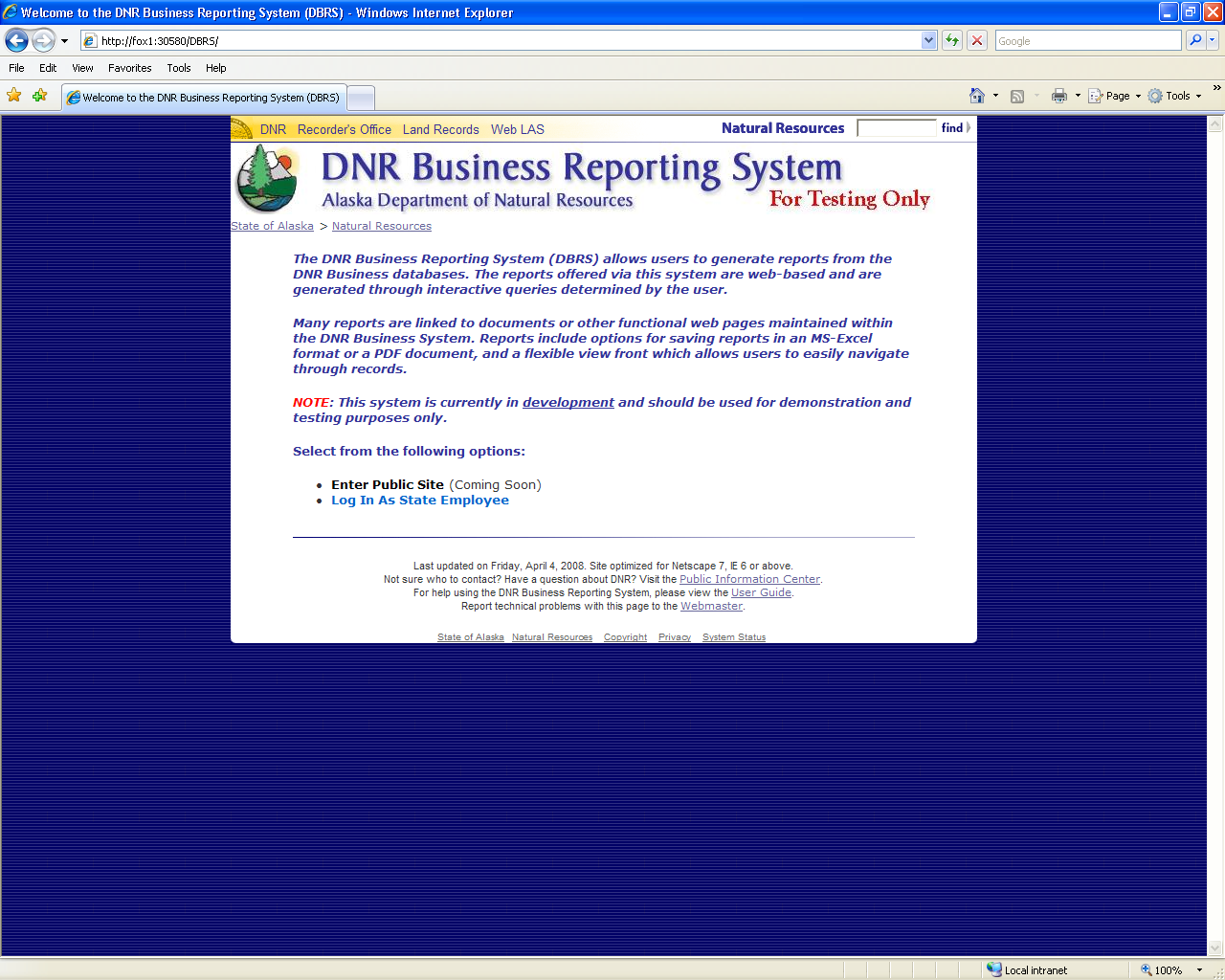
Although we were able to stick to our overall schedule, we underestimated the time required to complete some tasks. In the future, we will add extra time in our projected schedule to ensure we can complete the project in the allocated time.

**Appendix A: List of Figures**

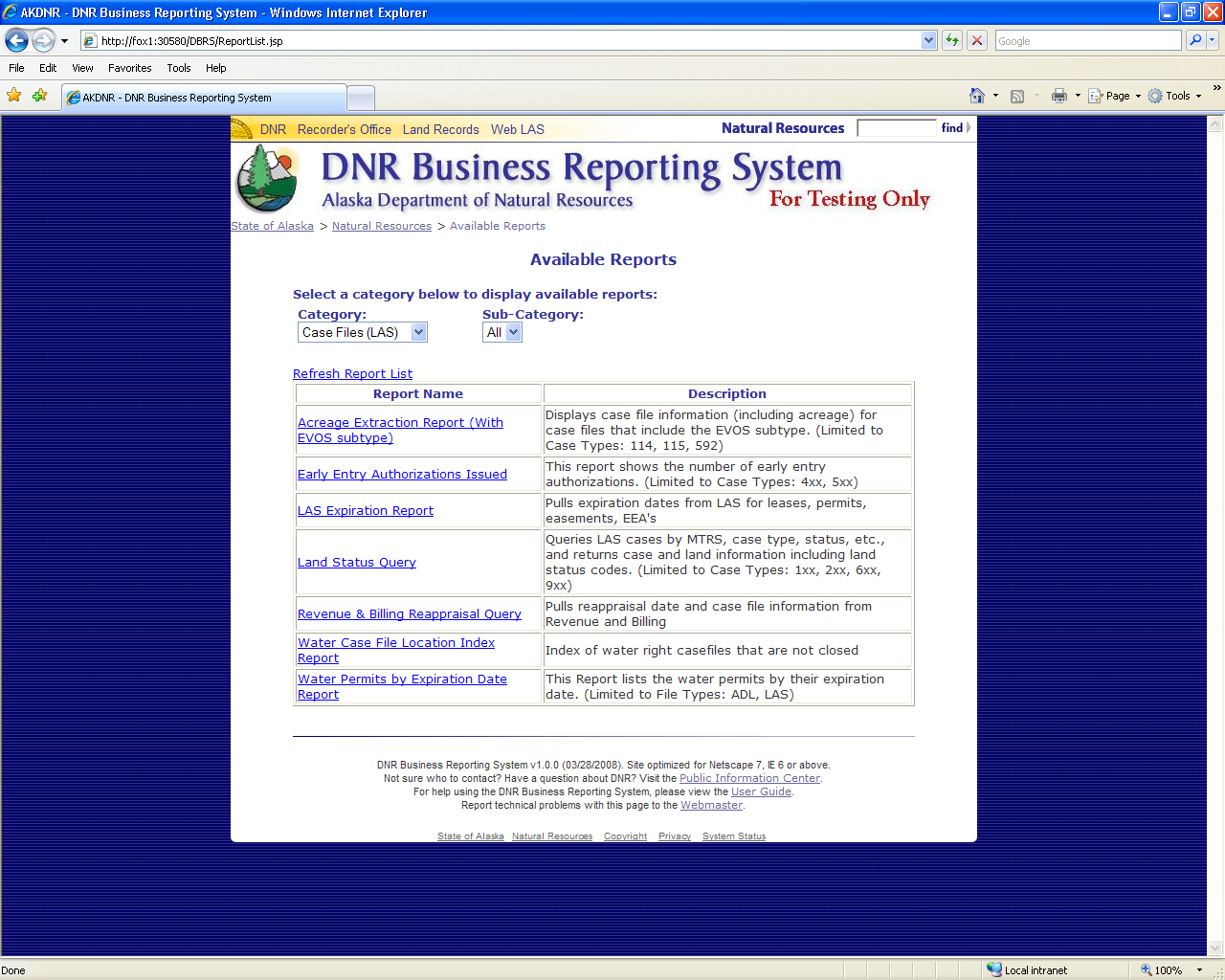
**Figure 1**: DBRS database tables (version 1.0)



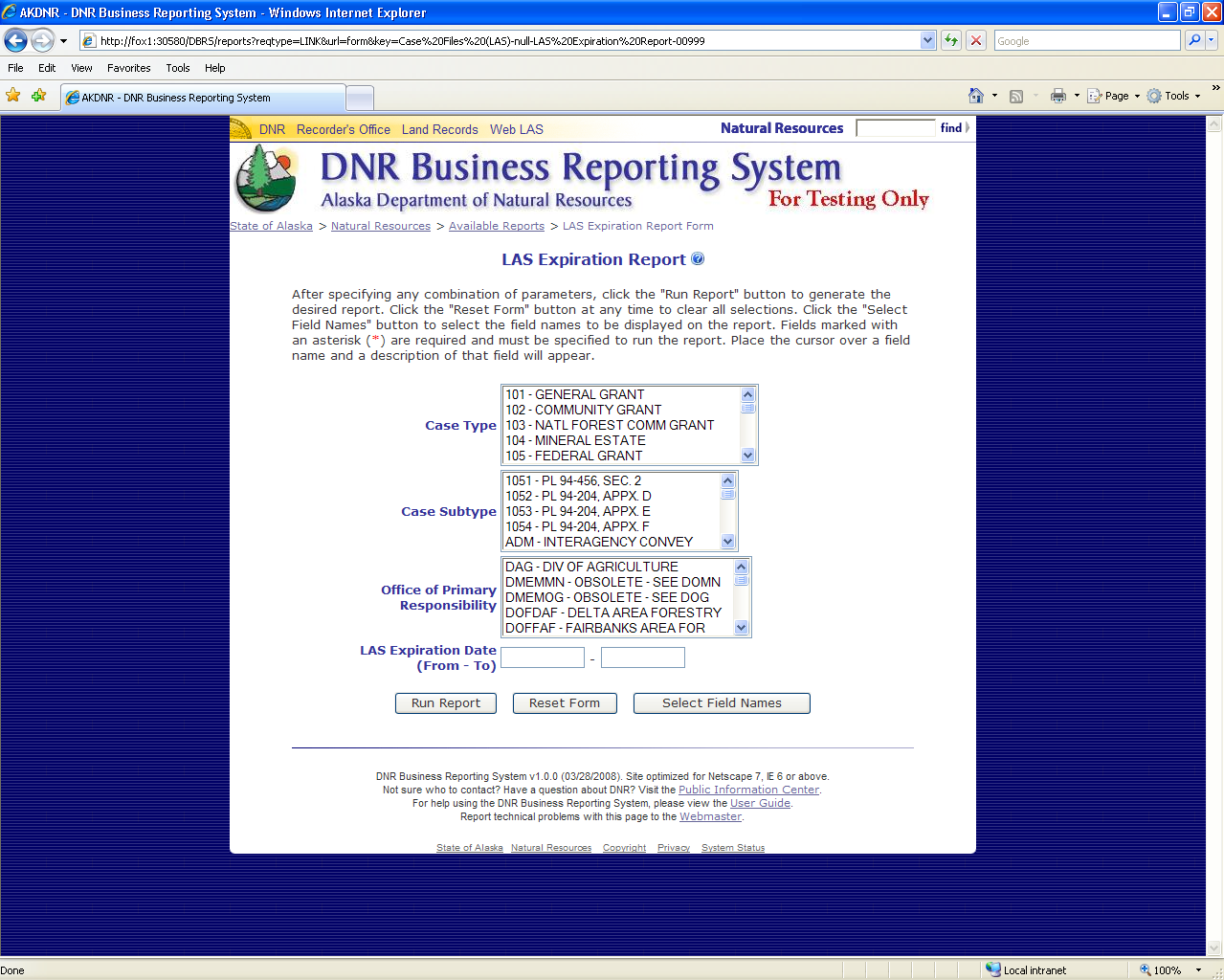
**Figure 2**: DBRS index page (version 1.0)



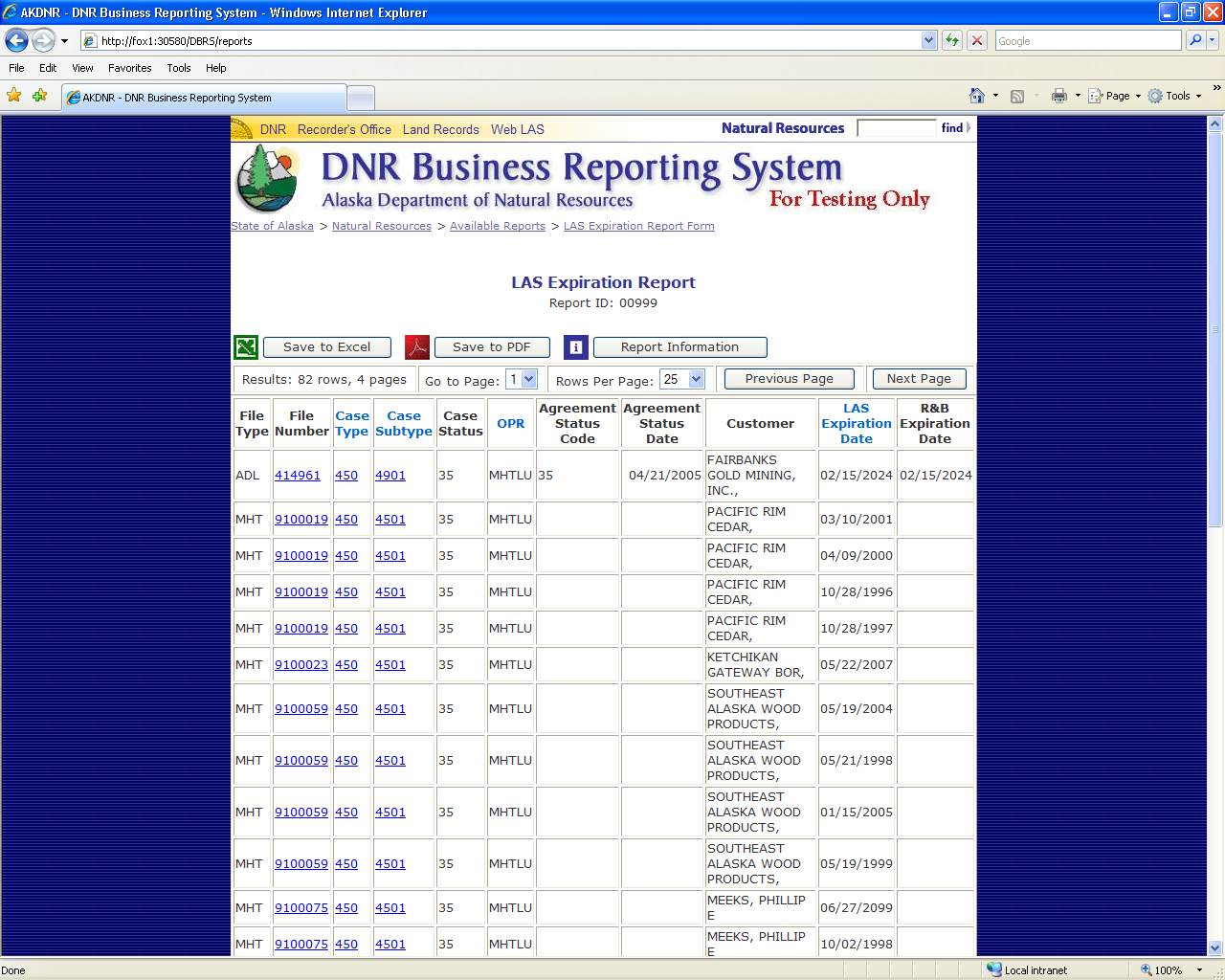
**Figure 3**: DBRS Available Reports page (version 1.0)



**Figure 4**: DBRS Report Parameter page (version 1.0)



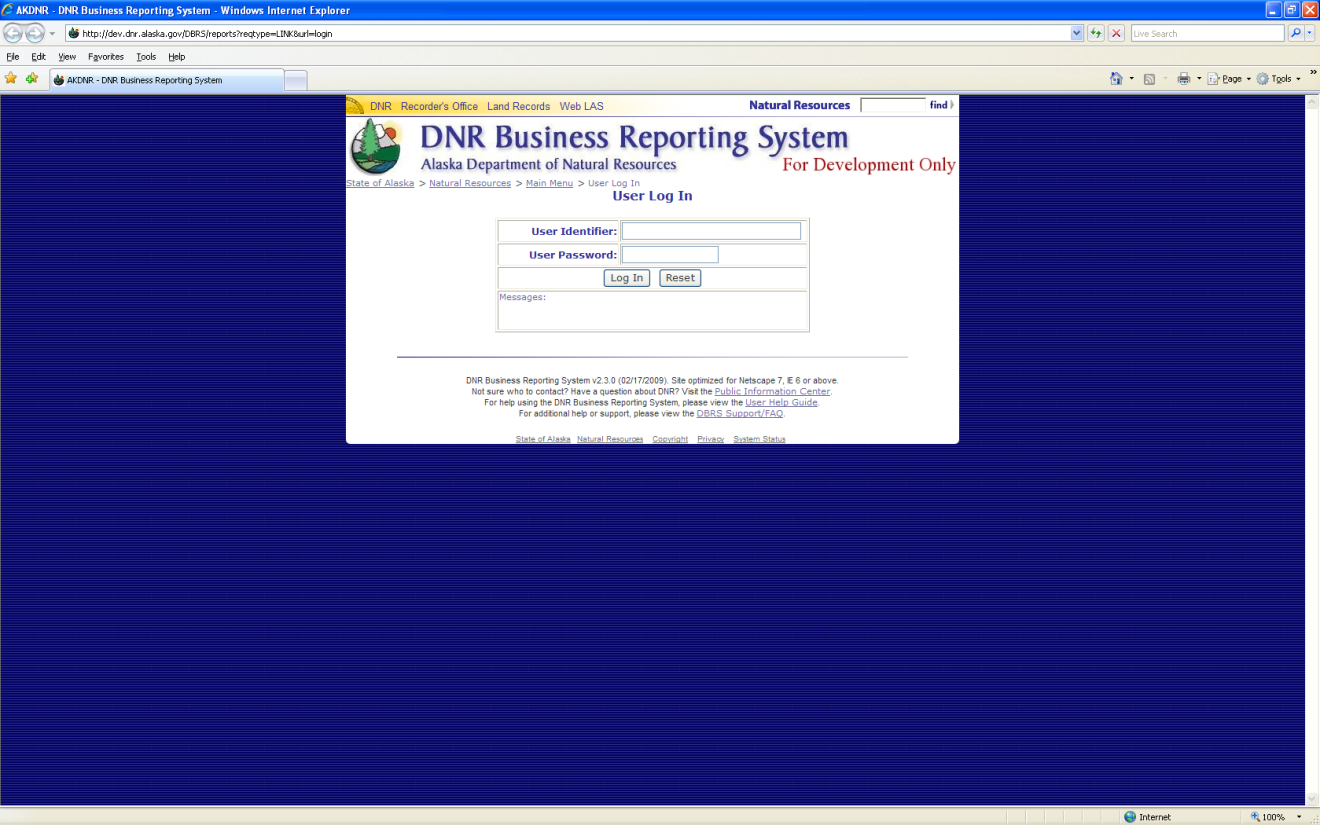
**Figure 5**: DBRS Report Display page (version 1.0)



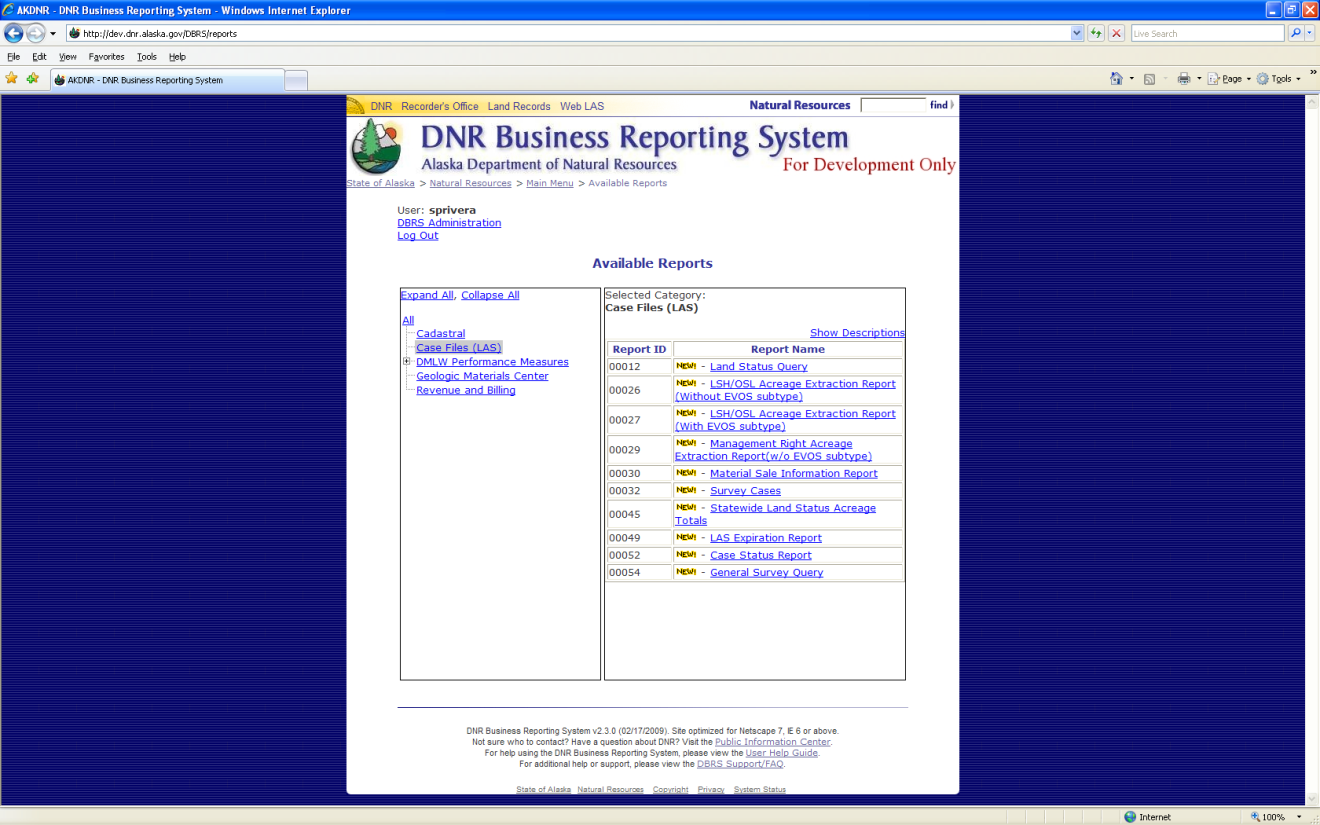
**Figure 6**: DBRS index page (version 2.0)



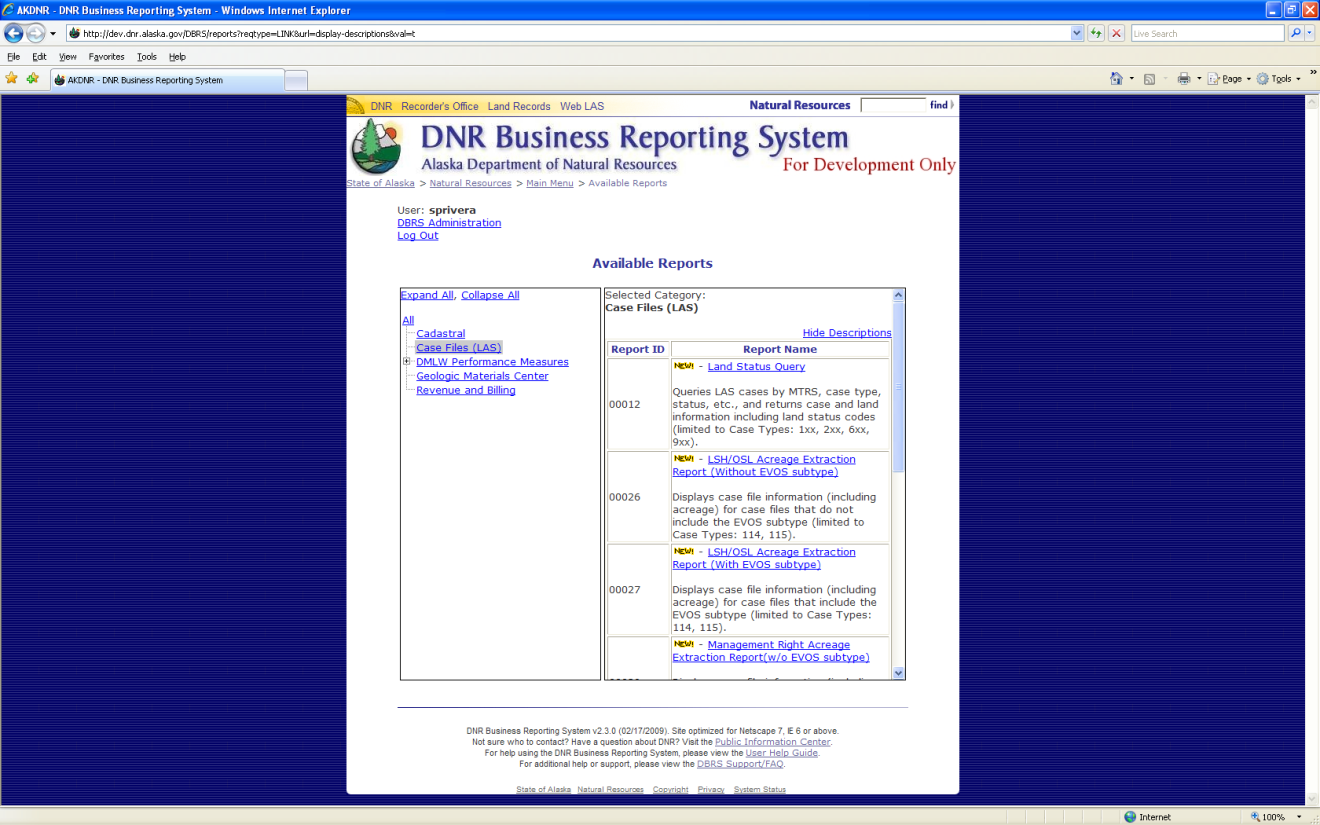
**Figure 7**: DBRS Login page



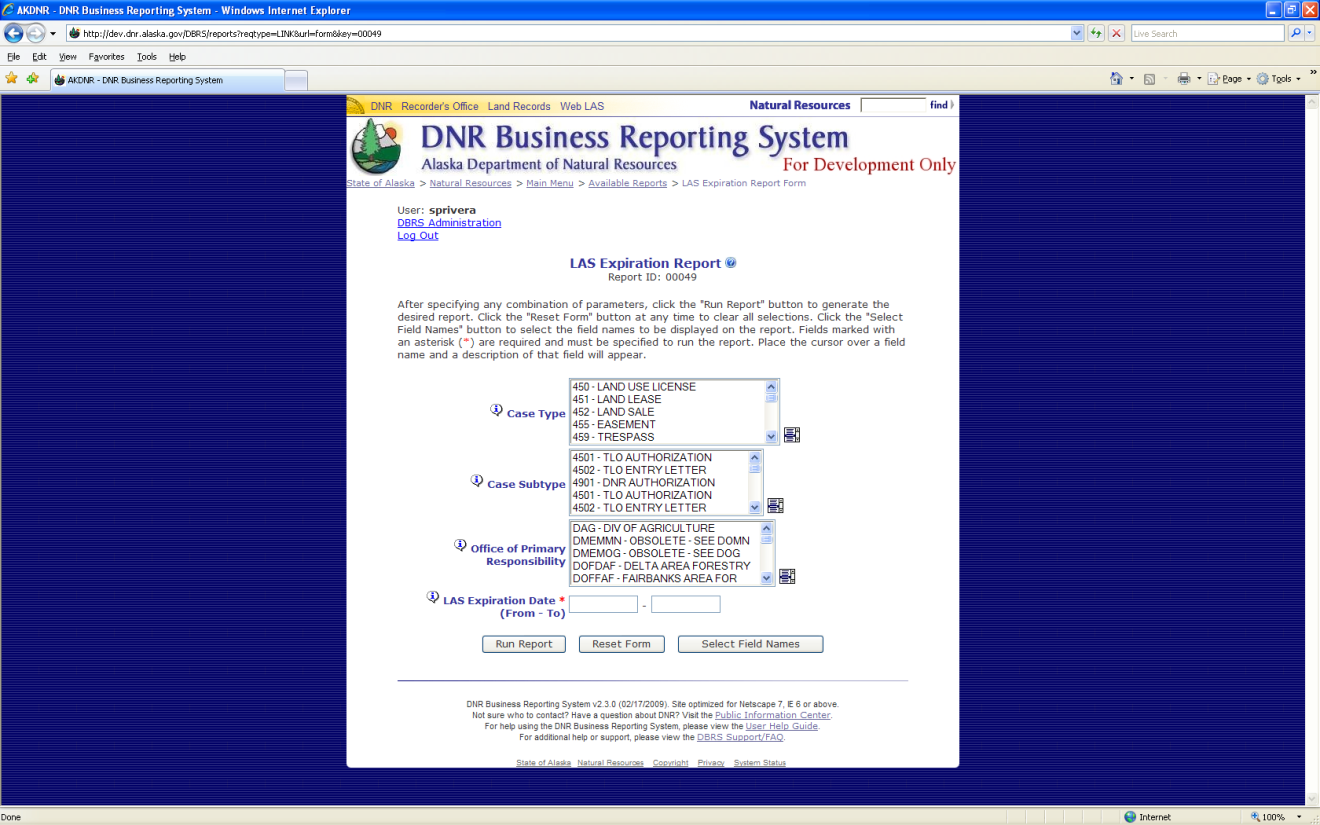
**Figure 8**: DBRS Available Reports page, without descriptions (version 2.0)



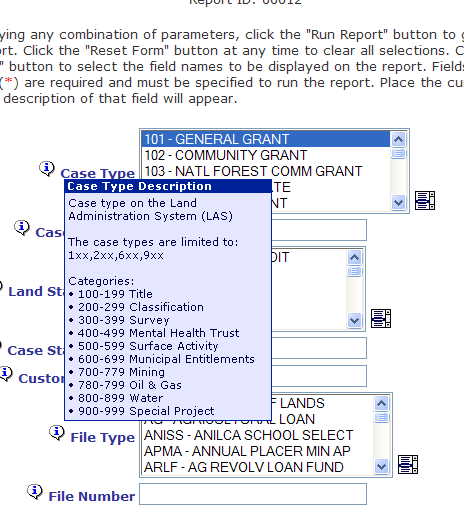
**Figure 9**: DBRS Available Reports page, with descriptions (version 2.0)



**Figure 10**: DBRS Report Parameter page (version 2.0)



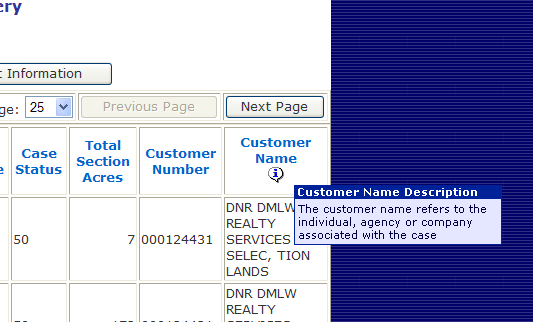
**Figure 11**: Sample report parameter description



**Figure 12**: DBRS Report Display page (version 2.0)



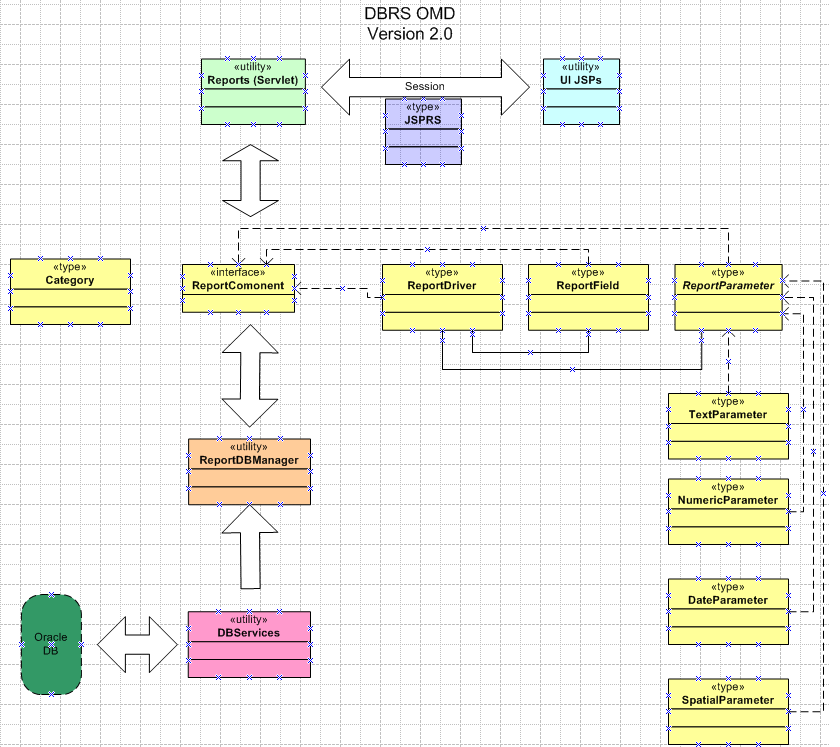
**Figure 13**: Sample report field description



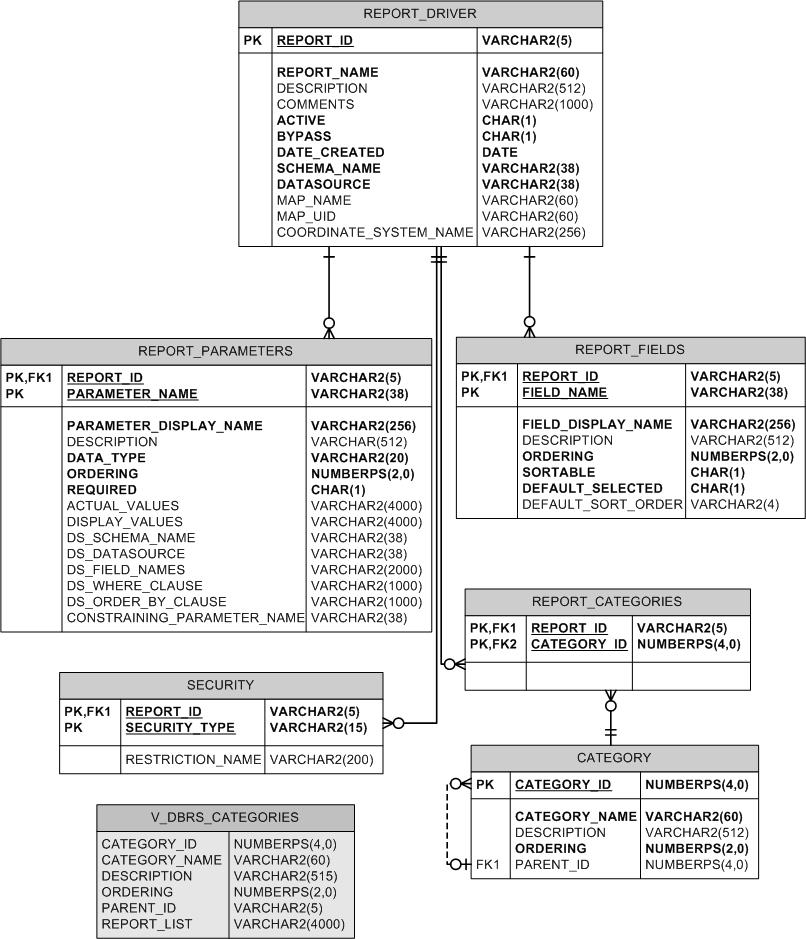
**Figure 14**: “Please Wait” message



**Figure 15**: DBRS Object Model Diagram (version 2.0)



**Figure 16**: DBRS Database Entity-Relationship Diagram (version 2.0)





**Figure 17(A)**: Projected timeline vs. actual timeline (Part A)



**Figure 17(B)**: Projected timeline vs. actual timeline (Part B)

**Appendix B: User Guide**

**Figure 17**: Projected timeline vs. actual timeline

See attached “DNR Business Report System User Guide.”

**Appendix C: Code Listing**

See attached.