1/2

CENTER FOR NUCLEAR WASTE REGULATORY ANALYSES
QUALITY ASSURANCE
SURVEILLANCE REPORT

PROJECT NO.:20-5702-154	REPORT NO.:94-02	PAGE 1 OF 2

SURVEILLANCE SCOPE: Review of activities and documentation supporting development of Regulatory Program Database (RPD), Version 1.0.

REFERENCE DOCUMENTS: User's Guide for Regulatory Program Database (RPD), Version 1.0
CQAM, Section 2, Revision 2, Change 6

STARTING DATE: March 24, 1994 ENDING DATE: March 28, 1994

QA REPRESENTATIVE: R. W. Folck

PERSONS CONDUCTING TEST/EXAM/ACTIVITY: R. D. Brient, A. R. DeWispelare, P. C. Mackin

SATISFACTORY FINDINGS:

It should be noted that Version 1.0 of the RPD was released prior to issuance of the current change to the CQAM. However, based on a limited documentation review and a brief interview session, initial development and on-going enhancements of the RPD are generally in compliance with the requirements of the CQAM.

Technical operating procedures are in-place to control RPD contents, updates, and reports. System level testing has been conducted and documented. Version and configuration control is being implemented. A user's manual has been developed and issued.

No unsatisfactory findings were identified but, several recommendations are presented below.

UNSATISFACTORY FINDINGS: None

NONCONFORMANCE REPORT NO.: None

ATTACHMENTS: None

RECOMMENDATIONS/ACTIONS:

The following recommendations are presented as a means to: 1) Enhance maintainability of the computer code, and 2) Ensure the validity of the output from the database. See page 2.

APPROVED: _____
CENTER DIRECTOR OF QUALITY ASSURANCE

DATE: 3/30/94

DISTRIBUTION:
ORIGINAL - CENTER QA DIRECTOR
ORIGINATOR
PRINCIPAL INVESTIGATOR - A. DeWispelare
ELEMENT MANAGER - P. Mackin
B. Sagar W. Patrick
H. Garcia

Recommendations/Actions:

RPD Design

Develop an operator/programmer manual. Aaron DeWispelare has a good start on such a document. The manual may be maintained informally but would contain sufficient information for a competent programmer to perform system maintenance when needed. The document would address:

- Installation instructions for each platform. It is assumed there will be some hardware/software dependencies which would require explanation.
- Data base design. The relationships between tables in the data base should be fully discussed. A graphical presentation may be helpful. Discuss and justify any unnormalized relations (i.e., repeating groups are allowed).
- Module and product interfaces. An interface design description of the code should be discussed at a high level and show the interfaces between Oracle, Topic, Galaxy, and the relationships between the various modules which serve as the "glue" for these packages.

Finally, early in the next phase of RPD development, the software requirements should be reviewed by the development team to determine if the requirements are: feasible and appropriate to implement in software; clearly and properly stated; consistent with each other; and testable. Any requirements identified as having potential problems should be reviewed and changes made as necessary.

Testing

The methodology used during the testing of the RPD is satisfactory. Having the code developers perform module/unit testing is appropriate however, they should document a test plan and the test results in a notebook or in the team's working files. Testing failures at this phase may also be handled informally.

Using WSE&I personnel in the performance system-level testing is an excellent way to obtain objective/realistic testing as well as, user feedback. In addition, the test logs generated during RPD Phase I system-level testing were found to be quite detailed. It is recommended that each system-level test be designed to correspond, on a one-to-one basis, with each of the functional requirements being implemented. This would allow complete validation of the RPD. It is also recommended that the analysis and disposition of system-level testing failures be fully documented.

Configuration Management

Using the UNIX Source Code Configuration Management System is an adequate method of configuration management. A change history should also be maintained as part of the header information of each module. This change history should note what, why, when, and who made a change.

Each module is currently noted with a version number. These version numbers should be updated as appropriate during the development process. However, all modules should revert to a baselined version number, (i.e., 1.0) prior to system-level testing. Any subsequent changes made during and after system-level testing would then be controlled to ensure appropriate evaluation and testing is performed and documented.

Coding

The code reviewed appeared to be structured and clearly written. Coding guidelines, (i.e. best practices), have been found to be useful in the development of uniform and maintainable computer code. Coding standards should address use of key words, indention and spacing, size of code aggregates, use of language constructs and features, (i.e. exception handling), conversions, and levels of complexity. SwRI QAP-17-1, "Software Development Control" contains a set of program design standards which may be used as a guide.