| TUGCO NUCLEAR ENGINEERING CPSES | PROCEDURE | REVISION | ISSUE DATE | PAGE |
|---|---|---|---|---|
| CONTROL COPY NO. INFORMATION ONLY | TNE-DC-10 | 1 | 2-9-84 | 1 OF 8 |
| DESIGN VERIFICATION OF COMPUTER PROGRAMS | PREPARED BY Craig L. Hamilton<br>APPROVED BY R D Calder | | | |

1.0 GENERAL

1.1 PURPOSE

This procedure outlines the requirements and acceptable methods for design verification of computer programs in accordance with the general design control requirements of procedure TNE-DC-1.

1.2 SCOPE

This procedure applies to the design verification of computer programs which are originated, developed, modified and/or used by TNE for computational purposes to determine the design, installation, performance, or operation of CLASS I or II equipment, systems, or structures (per procedure TNE-DC-1) at the Comanche Peak Steam Electric Station (CPSES).

1.3 RESPONSIBILITIES

The Supervising Engineers are responsible for implementing the provisions of this procedure and assuring that individuals within their disciplines who perform design verification of computer programs are qualified to do so and are fully aware of the associated responsibilities. Each Supervising Engineer shall be responsible for assuring that any production computer program (per procedure TNE-DC-3) which is used for CLASS I or II computations within his purview shall be design verified according to this procedure.

The Design Verifier shall be responsible for the selection of the verification method(s), as specified in this procedure, adequate completion and documentation of the verification, and the validity of conclusions reached as a direct result of this procedure.

1.4 PERSONNEL QUALIFICATIONS

A Design Verifier shall possess pertinent knowledge or expertise relative to computer science, programming, computer usage at CPSES, and the intended applications of the particular program in question.

In no case shall a computer program be design verified by the author of that program, by an individual who exercised singular control over the pertinent design inputs, theory, or methodology employed in the program, or who has immediate supervisory responsibility for the originator of the computer program.

1.5 DEFINITIONS

1.5.1 Design Verification

The process of reviewing, confirming, or substantiating the design by one or more methods to provide assurance that the design meets the specified design inputs.

1.5.2 Design Verifier

An individual who has been assigned to perform design verification for a particular computer program and who fulfills the responsibilities outlined in this procedure.

1.5.3 Computer Program

A series of instructions or statements in suitable format which define a logical sequence of operations to be performed by a digital computer in solving a particular physical problem or processing certain data.

1.5.4 Benchmark Problem

A physical problem which is well characterized to the extent that the correct solution is certainly known and which is reasonably analogous to a sample problem under investigation.

1.5.5 Sample Problem

A physical problem which is used as a basis for preparing representative input data in order to illustrate proper program usage and to obtain corresponding output data for purposes of testing calculational accuracy.

2.0 PROCEDURE

2.1 GENERAL

Prior to performance of design verification of a computer program, the Design Verifier must become thoroughly familiar with the available program documentation (user manual, program description. utilization instructions, etc.) and with the potential applications and limitations of the program. The functional aspects of the program must be understood sufficiently to allow preparation of proper test input data and to preclude inappropriate use of the program or misinterpretation of test results.

The basic objective of computer program design verification is to execute one or more sample problems as necessary to obtain verifiable results which may be used to demonstrate program validity for typical program application(s) or specific options available within the program. Based upon his review of the available program documentation, the Design Verifier shall determine the types of sample problem(s) which must be executed and the means of verifying the results. The verification of results shall include one or more of the following methods: hand calculations, alternate program calculations, benchmark results, or experimental data.

2.2 RESULTS VERIFICATION METHODS

2.2.1 Hand Calculations

When using hand calculations for comparison, the design verification documentation package must fully describe these calculations and reasons for differences between the results obtained and the sample problem output data. Calculational methods must be justified by references as necessary. Care should be taken to explain potential uncertainties for program applications relative to configurations where similar hand calculations are not feasible for checking purposes.

2.2.2 Alternate Program Calculations

When using this method of design verification, care must be taken to assure that the program being used for alternate calculations has itself been design verified and is being used properly. It is important to make certain that the sample problem is the same in the alternate program as in the program in question. The design verification documentation must adequately describe the comparable sample input data and output data for both programs and explain the differences in results.

2.2.3 Benchmark Results

If a suitable benchmark problem is available, comparison between sample output data and the benchmark results can provide a basis for program design verification. However, as in the case of alternate program calculations, it is the responsibility of the Design Verifier to ensure that the test input data adequately represent the benchmark configuration and to fully describe in the documentation package the input and output data and comparison of results.

2.2.4 Experimental Data

When suitable data are available, it is generally desirable to compare program output data with measured results. Such comparisons, when properly conducted, can yield the most credible evidence of program validity. In using this method of design verification, it is extremely important to ensure that all physical prerequisites are satisfied as necessary for appropriate testing and data acquisition and that the sample problem input data truly represent the experimental configuration.

2.3 DOCUMENTATION

Documentation of design verification of a computer program shall include preparation of a Design Verification Report (Figure 1). Restrictions on program use (e.g. unverified options) shall be noted on the Design Verification Report. The Design Verifier must complete this report and attach necessary supplementary information, including a completed Design Verification Checklist (Figure 2). It is the responsibility of the Design Verifier to determine those items on the checklist which are non-applicable (N/A). Any applicable item for which the design verifier answers "no" must be fully explained in the documentation in order for design verification to be complete.

In addition to the Design Verification Checklist, the supplementary attached information shall include a detailed description of the sample problems, calculations, test input and output data, data comparisons, associated information references, program limitations, etc., as required to adequately document the verification activities and to clearly establish program validity. The calculation portion of the documentation package should be prepared according to the requirements of Section 2.1 of procedure TNE-DC-3 (except a calculation number is not required).

By signature on the Design Verification Report, the Design Verifier signifies that the conclusions stated in the Report accurately reflect any identified errors or restrictions on use applicable to the otherwise verified program including the theoretical description and instructions for use.

2.4 RETENTION AND FILING

When a computer program Design Verification Report is completed, the report and all attachments shall be forwarded to the DCTG for processing per procedure TNE-AD-4.

Supporting information (e.g., information references, published user manuals, bulky computer listings, etc.) which are described in the documentation package but not attached should be uniquely identified as necessary and forwarded to the DCTG with the Report.

FIGURE 1
(TYPICAL)

COMPUTER PROGRAM
DESIGN VERIFICATION REPORT

Page 1 of __

Program Name: Size:

Language: User Manual:

Computer System: Compilation Date:

Program Description:

Method(s) of Verification:

Sample Problem Description:

Summary of Verification Activities/Results:

Conclusions/Restrictions on use:

Prepared By:

(Design Verifier) / / (Date)

Form: TNE-DC-10.1
February 1984

FIGURE 2
(TYPICAL)

DESIGN VERIFICATION CHECKLIST
COMPUTER PROGRAM VERIFICATION
(SHEET 1 OF 2)

Page 2 of ___

Program Name: Size:

Language: User Manual:

Computer System: Compilation Date:

| | Yes | No | N/A | Comment |
|---|---|---|---|---|
| 1. Is user manual available and up to date? | ___ | ___ | ___ | _____ |
| 2. Is source listing available? | ___ | ___ | ___ | _____ |
| 3. Is program widely used in public domain? | ___ | ___ | ___ | _____ |
| 4. Does TNE use program without modification? | ___ | ___ | ___ | _____ |
| 5. Were suitable sample problems executed? | ___ | ___ | ___ | _____ |
| 6. Was nodalization adequate? | ___ | ___ | ___ | _____ |
| 7. Are theory and equations reasonable? | ___ | ___ | ___ | _____ |

Form: TNE-DC-10.2
February 1984

FIGURE 2
(TYPICAL)

DESIGN VERIFICATION CHECKLIST
COMPUTER PROGRAM VERIFICATION
(SHEET 2 OF 2)

Page 3 of ___

Program Name:

Compilation Date:

| | Yes | No | N/A | Comment |
|---|---|---|---|---|
| 8. Are sample problem results acceptable? | ___ | ___ | ___ | _____ |
| 9. Can sensitivity studies be considered as unnecessary? | ___ | ___ | ___ | _____ |
| 10. Is preparation of program input data straightforward, without user judgement? | ___ | ___ | ___ | _____ |
| 11. Is program documentation adequate? | ___ | ___ | ___ | _____ |
| 12. Is documentation of sample problem solution adequate? | ___ | ___ | ___ | _____ |
| 13. Are program limitations easily understood? | ___ | ___ | ___ | _____ |
| 14. Do sample problems adequately exercise program options? | ___ | ___ | ___ | _____ |
| 15. Can program be used properly without special training? | ___ | ___ | ___ | _____ |
| 16. Does program employ appropriate methodology? | ___ | ___ | ___ | _____ |