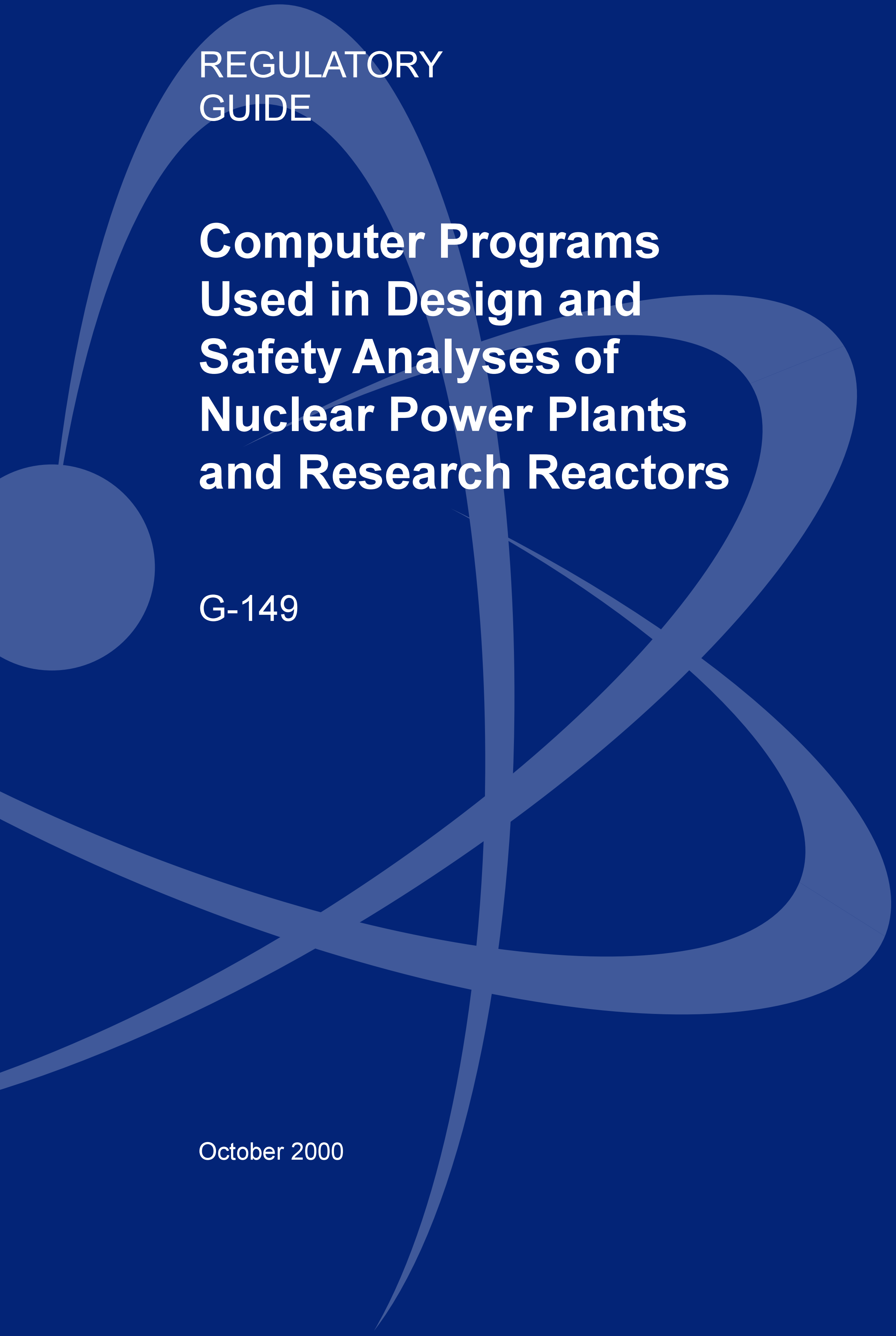
Canadian Nuclear
Safety Commission

Commission canadienne
de sûreté nucléaire



REGULATORY
GUIDE

Computer Programs Used in Design and Safety Analyses of Nuclear Power Plants and Research Reactors

G-149

October 2000

Canada

REGULATORY DOCUMENTS

The Canadian Nuclear Safety Commission (CNSC) operates within a legal framework that includes law and supporting regulatory documents. Law includes such legally enforceable instruments as acts, regulations, licences and orders. Regulatory documents such as policies, standards, guides, notices, procedures and information documents support and provide further information on these legally enforceable instruments. Together, law and regulatory documents form the framework for the regulatory activities of the CNSC.

The main classes of regulatory documents developed by the CNSC are:

Regulatory policy: a document that describes the philosophy, principles and fundamental factors used by the CNSC in its regulatory program.

Regulatory standard: a document that is suitable for use in compliance assessment and describes rules, characteristics or practices which the CNSC accepts as meeting the regulatory requirements.

Regulatory guide: a document that provides guidance or describes characteristics or practices that the CNSC recommends for meeting regulatory requirements or improving administrative effectiveness.

Regulatory notice: a document that provides case-specific guidance or information to alert licensees and others about significant health, safety or compliance issues that should be acted upon in a timely manner.

Regulatory procedure: a document that describes work processes that the CNSC follows to administer the regulatory requirements for which it is responsible.

Document types such as regulatory policies, standards, guides, notices and procedures do not create legally enforceable requirements. They support regulatory requirements found in regulations, licences and other legally enforceable instruments. However, where appropriate, a regulatory document may be made into a legally enforceable requirement by incorporation in a CNSC regulation, a licence or other legally enforceable instrument made pursuant to the *Nuclear Safety and Control Act*.

REGULATORY GUIDE

Computer Programs Used in
Design and Safety Analyses of
Nuclear Power Plants and Research Reactors

G-149

Published by the
Canadian Nuclear Safety Commission
October 2000

Computer Programs Used in Design and Safety Analyses of Nuclear Power Plants and Research Reactors
Regulatory Guide G-149

Published by the Canadian Nuclear Safety Commission

Catalogue number CC173-3/2-149E
ISBN 0-662-29526-9

Ce document est également disponible en français.

Document availability

The document can be viewed on the CNSC website. Copies in English or French may be ordered using the contact information below:

Communications Division
Canadian Nuclear Safety Commission
P.O. Box 1046, Station B
280 Slater Street
Ottawa, Ontario K1P 5S9
CANADA

Telephone: (613) 995-5894 or 1-800-668-5284 (Canada only)
Facsimile: (613) 992-2915
E-mail: info@cnsc-ccsn.gc.ca
Website: www.nuclearsafety.gc.ca

TABLE OF CONTENTS

COMPUTER PROGRAMS USED IN DESIGN AND SAFETY ANALYSES OF NUCLEAR POWER PLANTS AND RESEARCH REACTORS

1.0 PURPOSE

This regulatory guide is intended to provide guidance to licensees involved in the development, maintenance and use of computer programs used in the design and safety analysis of nuclear power plants and research reactors so that a high degree of confidence may be placed in both the programs and the results of their application.

2.0 SCOPE

This guide applies to licensees whose computer programs are used in:

- designing or supporting the design of a nuclear power plant or research reactor
- analyzing operational transients, incidents or accidents.

This guide does not apply to operational control systems software.

For computer programs developed before the effective date of this regulatory guide, the degree of applicability is specified in section 8 of this guide.

3.0 BACKGROUND

3.1 Regulatory framework

The Canadian Nuclear Safety Commission (CNSC) is the federal agency that regulates the use of nuclear energy and materials to protect health, safety, security and the environment, and to respect Canada’s international commitments on the peaceful use of nuclear energy.

The *Nuclear Safety and Control Act* (“the Act”) requires persons or organizations to be licensed by the CNSC for carrying out the activities referred to in Section 26 of the Act, unless otherwise exempted. The associated regulations stipulate prerequisites for CNSC licensing, and the obligations of licensees and workers.

3.2 Licensing process

The CNSC typically applies a phased process to its licensing of nuclear facilities and activities. For major facilities, this process begins with a consideration of the environmental impacts of the proposed project, and proceeds progressively through site preparation, construction, operation, decommissioning and abandonment phases.

The *Nuclear Safety and Control Act* and regulations require licence applicants to provide certain information at each licensing stage. The type and level of detail of this information will vary to accommodate the licensing stage and specific circumstances.

At all licensing stages, applications may incorporate (directly or by reference) new or previously submitted information, in accordance with legislated requirements and the best judgement of the applicant. An application that is submitted at one licensing stage can become a building block for the next stage.

Upon receipt of an application that is complete, the CNSC reviews it to determine whether the applicant is qualified to carry on the proposed activity, and has made adequate provision for the protection of the environment, the health and safety of persons, and the maintenance of national security and measures required to implement international obligations to which Canada has agreed. If satisfied, the CNSC may issue, renew, amend or replace a licence that contains relevant conditions. Typically, this licence will incorporate the applicant's undertakings, and will contain other conditions that the CNSC considers necessary.

3.3 Relevant legislation

The CNSC uses this guide to assess information submitted as part of a licence application. Specifically, the *General Nuclear Safety and Control Regulations*, under paragraph 3(l)(*i*), requires that applicants provide "a description and the results of any test, analysis or calculation performed to substantiate the information included in the application." In addition, section 6 of the *Class I Nuclear Facilities Regulations* requires that the application for a licence to operate a Class 1 nuclear facility contains "(*b*) a description of the systems and equipment at the nuclear facility, including their design and their design operating conditions;" and "(*c*) a final safety analysis report demonstrating the adequacy of the design of the nuclear facility."

It is the licensee's responsibility to ensure that the computer programs used in design and safety analyses of nuclear power plants and research reactors and the output of these programs are reliable and adequate for their intended applications. These objectives can be attained either by:

(a) using the methods of development, maintenance and use of computer programs suggested in this guide, or
(b) applying other methods that are equally, or more, effective.

3.4 Commitment to quality

An organization's commitment to software quality is important, and it is the responsibility of licensees to develop a quality-assurance program for the development, maintenance and use of computer programs.

Adherence to relevant standards referred to in this guide, where appropriate, is one of the important elements in producing and maintaining high-quality computer programs.

4.0 DEVELOPING A COMPUTER PROGRAM

4.1 Developing a computer program in phases

Develop computer programs in phases, using the output of one phase to help set the requirements for one or more of the phases that follow. Typical development phases (see Glossary for definitions) include the following:

- problem definition
- requirements specification
- program design
- coding
- program integration
- validation

4.2 Development phases

Each development phase should consist of the following elements:

- define the input and requirement specifications — the functions to be performed and the expected outputs — based on the preceding phase;
- design, develop or test, as stated in the specifications;
- perform verification and design review, where appropriate, to ensure that the product matches the requirement specifications defined for the phase. If a phase cannot be verified until after more than one phase is complete, each phase should be verified iteratively.

4.3 Guidelines for verification and design review of development phases

With the exception of the validation phase, guidelines for the verification and design review of development phases are provided in sections 4.5 through 4.11 of this guide. Although there are no explicit guidelines for other development activities for these phases, the important elements to be considered in each development phase may be obtained from these guidelines.

4.4 Guidelines for validation

Guidelines for validation are provided in section 4.12 of this guide. Although validation has been described as a development phase, the activities of this phase may require extension beyond development phases to ensure that the computer program is validated for specific applications.

4.5 Overview of verification and design review

Perform verification and design review to show whether or not a program meets its requirements specifications. Specifically, these activities show the following:

- The mathematical equations adequately reflect the phenomena and processes of the physical system.
- The program design correctly reflects the mathematical equations.
- The program accurately reflects the design.
- The program's results satisfy its requirements specifications such that the results mirror the behaviour of the physical system.

Prepare a plan of the activities for verification and design review early during program development, for example, in the requirements specification phase. The plan will typically include the objectives, verification and design review approach, the schedule, and the project organization and management.

Review the verification and design review plan on a set schedule, updating the plan as necessary.

4.6 Verification

Assign verification tasks to the developer of a computer program and independent, qualified personnel.

Place the outputs to be verified under configuration control (refer to section 5 of this guide) before starting verification.

4.7 Verifying the requirements specifications

Demonstrate that:

- the specifications conform to the standards adopted for the program development so that, for example, all required sections are present and each section contains all required information;

- the specifications include the following detail:
  - all required functions;
  - all program input and output requirements, described in enough detail for a program to be designed;
  - identification of development standards;
  - acceptance criteria;
- the specifications are clear and unambiguous.

4.8 Verifying the program design

Demonstrate that:

- the design conforms to the standards adopted for the development of the program so that, for example, all required sections are present and each section contains all the required information;
- the design can be traced to the requirements specifications, so that all requirements are reflected in the design and all design features stem from these requirements;
- the design is complete, showing the required functions and specifying the program inputs and outputs, the operational environment and the processing steps;
- the design is internally consistent;
- the design is clear and unambiguous.

4.9 Verifying the coding

Demonstrate that the source program:

- conforms to programming and language standards adopted for the development of the computer program;
- does not include unnecessarily complex coding;
- does include error-checking capabilities;
- does include logic that is consistent with the design specifications.

4.10 Verifying the program integration

Demonstrate that:

- the integrated program conforms with the requirements of the operating system;
- the integrated program interfaces properly with external files;
- the program links correctly — for example, all modules and subroutines are properly linked;
- the control language is correct;
- the processing and transmission of data between modules are correct.

4.11 Performing design review

Perform a design review in at least the following phases:

- requirements specifications
- program design

Include representatives of the following groups in the design review team:

- developers
- programmers
- end-users
- a quality-assurance group

The design review team should include independent members with well-recognized expertise in review areas, for example, experts in the physical sciences, mathematics, computer science and software programming.

In the design review of the requirement specifications, demonstrate that:

- the technical description of the problem to be solved is correct, complete and consistent with the statement of the problem;
- the solution methods are appropriate, based on state-of-the-art knowledge.

During the design review of the requirement specifications, review the following:

- the scope of the problem, the specifications of the physical system and the identification of the initial and boundary conditions relevant to the phenomena to be modelled;
- knowledge of the physical system;
- identification and understanding of the phenomena;
- the modelling concept and the models of the physical system and phenomena, including assumptions, methods of approximation and simplification;
- the limitations of the models, and what these limitations imply;
- the analytical and numerical solution methods, including the numerical techniques and algorithms;
- the requirement specifications and instructions for coding the computer program.

The review of the computer program design aims to demonstrate that all program requirements are implemented correctly in the design, that all aspects of the design can be traced to the requirements and that the design is feasible, consistent and based on state-of-the-art knowledge.

During the design review of the computer program, review the following:

- program conceptual design
- basic logic
- system flow diagrams
- numerical techniques and algorithms
- operational environment and program interface

In the design review of the computer program, demonstrate that:

- the numerical techniques are appropriate for the types of problems to be solved, as well as for the treatment of boundary or inter-phase conditions and special situations, such as singularities;
- the mathematical equations have been transformed correctly into a numeric scheme of solutions, and the steps in the solution design are in the correct sequence;
- the user options and their restrictions are clearly described.

4.12 Performing validation

Perform validation to determine the accuracy of the computer program's predictions, and to help determine the uncertainties discussed in section 6.2 of this guide.

Develop a validation plan early in the development of a computer program. The validation plan should include:

- the objectives and scope of the validation;
- the validation approach, including the method of validation, requirements to be validated, acceptance criteria for each requirement, and method for evaluation of validation results;
- the basis for the selection of validation cases, and the specifications for each case;
- the validation database;
- the validation procedure, including the validation result reporting procedure.

Review the validation plan on a set schedule, updating the plan as necessary.

Assign program validation tasks to qualified persons, specifically developers and those who are familiar with the program, the sources of the data being used to validate the program and the program's intended use.

Validation tests are meant to determine:

- a program's accuracy;
- how well a program and its conceptual models reflect physical processes or systems;
- a program's capabilities and limits, including the range of parameters to which the program may properly be applied.

During validation, systematically test the performance of conceptual models, empirical correlations and the integrated program. Relevant are tests against analytical solutions, operational data, and data from separate effects and integral effect tests. Comparisons against validated programs may be included.

To perform systematic testing:

- develop a test plan;
- identify and rank important phenomena and parameters;
- identify the models, empirical correlations or components of the program that are to be tested;
- identify suitable existing experimental data, as well as data to be obtained from new experiments;
- assess the program using separate effect and integral effect tests, operational measurements, analytical solutions or the results of one or more validated programs;
- assess the sensitivity to the input options;
- evaluate validation results.

During validation, examine the following key areas when applicable:

- the assumptions made to simplify the physical system;
- the theoretical and experimental bases for the models and empirical correlations, as well as their applicability range;
- the compatibility of the ranges of the parameters, as well as the geometrical and phenomenological similarities between the system being simulated and the experiments;
- the ability of the code to predict behaviour during integrated tests, to show that (1) no unwarranted interaction occurs between various models, and (2) the code accurately simulates how the system's parts interact;
- the sensitivity of individual models and empirical correlations, as well as the integrated code, to variations in key parameters or factors, particularly near the end of their allowable ranges and near-critical values such as singularities or discontinuity;

- potential errors arising from the use of the models and empirical correlations outside their intended ranges;
- the variation in code predictions when using various options of the alternative models and empirical correlations as recommended in the code documentation (refer to section 7 of this guide).

Note all validation results, clearly identifying the program's abilities, limits and ranges of applicability.

During validation, identify improvements that should be made to the program. Identify, as well, any need to enhance the validation data, including experimental data.

Write a report, based on validation results, that shows the correct and appropriate use of the program.

Verify that the validation process has been conducted in accordance with the validation plan, and that the validation results are accurately reported.

5.0 MAINTENANCE

5.1 Configuration management

Establish procedures for computer program maintenance and change control.

Adopt practices that protect program integrity. Clearly define the roles and responsibilities of those who are responsible for a program's integrity, as well as those who maintain and use it.

Maintain a configuration management system to help ensure program integrity and to keep track of a program's versions and components.

Typical components of a program to be configured include:

- source program
- object program
- executable program
- processors and operating systems
- files that control the generating of executable code from source code
- input data files
- documentation
- tools that support program development or maintenance

Configuration management involves:

- using a naming convention to uniquely identify the functional and physical characteristics of each configuration component
- controlling changes to the configuration components
- recording and reporting change processing and implementation status
- verifying compliance with configuration requirements

5.2 Change control

Set up a change control system to control changes to each configuration item. A change control system should include:

- a designated person responsible for all changes to a computer program;
- a change approval group whose members represent developers, users and those who performed the program's verification and validation;
- procedures to:
  - review proposed changes to a configuration item;
  - approve changes;
  - implement changes;
  - document changes and their rationales;
  - verify that a change was implemented correctly;
  - assess the impact of each change on the use of the program and on the quality of its predictions;
  - make any necessary revisions to existing documentation.

6.0 USING COMPUTER PROGRAMS

6.1 Overview

Establish guidelines and procedures for the use of computer programs. These should include guidelines for user prerequisites and the designation of persons responsible for ensuring the correct and appropriate use of computer programs.

Apply a program only to a problem or set of problems for which the program has been validated and designed to solve (only within the range of applicability identified in the verification and validation reports). If the program is used outside its range of validation, the validity of the extrapolation should be justified.

Minimize as far as possible the program's user effects. An illustration of user effects is: if two experienced users, with the same version of a computer program, perform a test or analyze a problem and obtain significantly different results, then the program may be user-dependent. The differences can be attributed to the use of input options, such as those for time step control, nodalization schemes, correlations, coefficients, or parameters.

Report errors and deficiencies in a program or its application to the person who is responsible for the maintenance of the program. Correct the errors and deficiencies, using the change control procedures listed in section 5.2 of this guide.

6.2 Uncertainty analysis

Evaluate a computer program's results to determine the uncertainties in each type of application of the program.

Account for potential uncertainties identified in the development and use of the program, including those due to:

- simplifications and assumptions made to compensate for lack of knowledge or to render a problem solvable, including assumptions that simplify equations, empirical correlations, and physical and system models;
- inadequate knowledge of the problem that the computer program will be used to analyze, such as complex phenomena, scaling effects, and initial and boundary conditions;
- limits to the ability to represent a physical system, such as those presented by nodalization techniques;
- inadequacy of the data used to validate the program, for example, instrumentation errors and test repeatability;
- compensating errors or uncertainties;
- propagation of uncertainties, i.e. the uncertainty of the results increases as the transient progresses.

6.3 User prerequisites

Take steps to ensure that each user of a computer program possesses experience that matches or exceeds the safety profile of the program's application.

Important factors in the knowledge and experience needed by the program's user are:

- good knowledge and understanding of the information in all application documents and the verification and design review reports, as well as familiarity with other information in the design documents;
- sufficient experience in using the program for the intended application, and similar ones, with a good knowledge of program responses to (1) various system and phenomenon models and to (2) assumptions and changes in important parameters;
- extensive knowledge of the problem to be analyzed, for example, knowledge of the physical system and the phenomena being modelled.

6.4 User support

User support should be available to advise the user on the correct use of a computer program and proper modellings or simulations for important or large and complex programs, such as thermal hydraulics system programs or multi-field coupling programs.

Set up a user support team if warranted, and include the following members:

- the developer or those who know the program well;
- those who performed, or who know a great deal about, the program's verification and validation;
- those who have a thorough knowledge of the system to be modelled and the problem to be solved.

6.5 User options

User input options should be avoided as far as practicable.

Write guidelines for each user option, and make them available to users.

6.6 Verifying the application process

Verify that the application process is correct by demonstrating that it satisfies the guidelines for the use of computer programs provided in this section.

If an application process cannot be satisfactorily verified, identify exceptions, and provide a justification for each.

7.0 DOCUMENTATION

7.1 Overview

The documentation of a computer program and its associated components should conform with documentation standards adopted in the development of the program. Verify the program documents to demonstrate that they are complete, consistent, clear and unambiguous.

7.2 Creating development documentation

In the development documentation include:

- problem definition
- program development plan
- requirements specifications
- design specification
- programmer's manual
- verification and design review report

In the problem definition, document in detail the statement of the problem and the rationale and objectives of the program.

In the program development plan, describe the organization, schedule and activities related to the development, design review, verification and validation of the program. Also document procedures for updating the program development plan.

In the requirements specifications, clearly identify all program requirements. In addition, provide the basis for verifying the program design and evaluating the performance of the program through validation.

In the design specification, specify the elements needed for program coding, including the logical structure, information flow, models, numerical solution techniques, discretization method, data structure, supporting software and hardware, and the operating environment.

In the programmer's manual, describe the program flow and structure, the method of translating theory into coding, instructions for maintaining and modifying the computer program and conventions on programming practices, such as variable naming and program commenting.

In the verification and design review report, document verification activities, including the description of the method and results of each verification or design review, the range of applicability and recommendations to users regarding the capability and limits of the computer program. Also include the input data used in each verification.

7.3 Application documentation

Application documentation should include:

- program abstract
- theory manual
- user's manual, including specific user guidelines
- one or more validation reports
- source code
- sample input and output

In the program abstract, provide the following information about the program:

- the name and version of the program and applicable configuration items;
- the program's purpose, capabilities, limitations and operating environment;
- a summary of the problem(s) the program is designed to solve;
- the names of the organization and key individuals responsible for code development, support and maintenance of the program.

In the theory manual, describe the basis of the computer program including:

- the physical systems to be modelled and their models;
- the phenomena to be modelled and their models or empirical correlations;
- mathematical formulations of the problem and solution techniques;
- assumptions and constraints, and what they imply about the limits of the program's capabilities and the range of its applicability;
- references.

In the user's manual include enough information to run the computer program. Provide details on the required input data, techniques for restarting the code, diagnostic information and options available to the user, including:

- basic information on compiling, linking and executing the code;
- detailed description of all input parameters indicating type and format;
- a discussion of pre- and post-processing of the code, including code restarts;
- sample input and output files that show representative problems;
- a description of possible execution error messages and termination messages.

In addition, to minimize the risk of inappropriate use of a program, provide user guidelines on the following topics:

- the correct and proper use of the program for each of its applications;
- the range of applicability;
- the program's capabilities and limitations;
- the options available to users, such as model options or choice of nodalization schemes.

In the validation report, include the following:

- validation plan;
- validation results;
- evaluation of the validation results, including a determination of the computer program's accuracy and range of uncertainty for each test;
- recommendations for improvement to the program;
- specification of further experiments;
- recommendations for the correct and appropriate use of the program for each application, range of applicability, and program capability and limitations.

8.0 EXISTING COMPUTER PROGRAMS

Existing computer programs are those used by the CNSC licensee before the effective date of this document.

Assess and document the extent to which existing programs conform with this guide.

Improve, to the extent practicable, the areas that do not conform with relevant sections of this guide so that there is an increase in the level of confidence in a program consistent with the program's impact on safety.

Provide justification for any non-conformance with the guidelines in this guide.

9.0 PROCUREMENT OF COMPUTER PROGRAMS

Establish procedures for the procurement and maintenance of computer programs. Ensure that all computer programs destined for use by a CNSC licensee, but developed outside a CNSC licensee's organization, meet or exceed the criteria of this guide.

GLOSSARY

coding
The step in which a computer program design is transformed into a sequence of machine-readable instructions suitable for processing by a computer.

computer program
A sequence of machine-readable instructions suitable for processing by a computer; also referred to in this document as a program.

developer
A person who writes the code for a computer program, or a portion of one, and who works on its associated documents.

discretization
A method of approximation of the true mathematical function to be integrated.

problem definition
A detailed description of the problem and the rationale for the decision to develop a computer program.

program design
The elements required for program coding, including the logical structure, information flow, models, numerical solution techniques, discretization method, data structure, supporting software and hardware, operating environment, and other features required to satisfy the requirements specifications.

program integration
The step in which a source program is integrated with other components, such as library routines, system linking specifications, operating system control language, external data libraries and hardware environment to form an integrated, executable program.

requirements specifications
The requirements the program must satisfy and the basis for verifying the program design and then evaluating the program through validation.

source program
A program that must be compiled, assembled or interpreted before it can be executed.

user
A person who uses a computer program.

validation
A process by which the results of a computer program are compared with operational measurements, experimental data, or analytical solutions, to determine the program's accuracy and uncertainty.

verification
The step in which it is determined whether the products of a given phase of the program development cycle fulfil the requirements set during the previous phase.