Documentation Guidelines for NPARC Alliance Software Development[*]

Charles E. Towne
NASA Glenn Research Center
Cleveland, Ohio

1 Introduction

This document describes the guidelines to be used when writing documentation during NPARC Alliance software development projects. It was originally written for the NASTD/NPARC/NXAIR code merger effort, but should be applicable to other software development projects as well.

2 Guidelines for External Documentation

This section discusses the external documentation that should be prepared describing the production software. This is the documentation that will be released with the code, in printable and/or electronic form.

Generally there will be two levels of documentation — one for users and one for developers. For relatively simple applications, these may be combined in a single document, but for more complex programs they should be separate. Depending on the application, it may be appropriate to supply more than one user- and/or developer-level document. For example, it may make sense to have a user-level Installation Guide that's separate from the User's Guide. For some types of software, a third level of documentation may be desirable, providing a separate high-level detailed description of the analysis, modeling assumptions, numerical solution procedure, etc.

2.1 User-Level Documentation

This is the documentation that a typical user will refer to most often. It should include all the information that a user, from beginner to expert, needs to successfully apply the program to real-world problems. Ideally, though, it should be organized in such a way that a new user can learn the basics and start running the program relatively quickly, without having to wade through descriptions of advanced and/or rarely-used features.

The user-level documentation should include:

- Where appropriate, a brief description of the analysis, modeling assumptions, numerical solution procedure, etc., especially if a separate detailed "Analysis Description" is not being provided.

[*]This is version 1.0 of this document, released 8 Sep 1997.

- A brief description of important code features and capabilities.
- A description of the mechanics of running the program, including details about how to use any user-level scripts supplied with the program.
- A tutorial, especially for large and/or complex programs, that guides an inexperienced user step-by-step through the process of running a new case.
- Definitions of all input and output variables.
- For interactive programs, details about commands that are used, and/or how to use the GUI interface.
- A discussion of the computational resources required.
- A description of any user-level input and/or output files, including the contents, the associated Fortran unit number, and the file format. Where appropriate, instead of (or in addition to) describing their contents and format, information should be included describing how to create and/or extract information from the files.
- A list of error messages that may be generated by the program, including suggested corrective action.
- Example cases illustrating the proper procedure for running realistic applications.[1]
- Where appropriate, a separate 2–4 page "Quick Guide" to running the program.

2.2 Developer-Level Documentation

This is a reference document that a developer will refer to for detailed information that's needed when modifying or extending the program.

The developer-level documentation should include:

- A description of the overall code structure, including flow charts and subprogram calling trees.
- Details about scripts used to run the program.
- A description of all the files read and/or written by the program, including their contents, the associated Fortran unit number, and the format of the file.
- Definitions of all common block variables.
- Details about each subprogram, including: a list of other subprograms that call or are called by the subprogram; a list of the files read and/or written by the subprogram; definitions of the input, output, and key local variables; and a description of the work being done in the subprogram.

[1]For the NASTD/NPARC/NXAIR merger, input and output files for the example cases will be supplied with the program and/or made available over the WWW.

3 Guidelines for Internal Working Documentation

This section describes the documentation that should be supplied by developers during the course of a software development project. This type of documentation is especially important in large projects, in which several people are involved in writing software and developing the external documentation. The purpose of the internal working documentation is to:

- Establish a detailed history of the software development.
- Serve as reference material for other developers during the course of the project.
- Provide raw material for the development of the external documentation described above.

Whenever a change is made to an existing subprogram, the developer should supply the following information[2]. A suggested format is presented in the Appendix.

- The date and author of the change.
- A brief description summarizing what changes were made, and why.
- A detailed technical description of major changes, to be used in updating the subprogram description in the developer-level documentation.[3]
- Changes that should be made to the user-level documentation, such as definitions of any new input or output variables, a description of any new files that are used, and a list of new error messages that may be generated.
- Changes that should be made to the developer-level documentation for the subprogram, such as a list of additional subprograms that call or are called by the subprogram, and definitions of any new input, output, and/or key local variables.

The same type of documentation is needed when a completely new subprogram is added to the code, except of course that *all* the information needed to describe the subprogram in the developer-level documentation should be supplied.

[2]For the NASTD/NPARC/NXAIR code merger project, this should be included whenever new code is checked into the version control system.

[3]For the NASTD/NPARC/NXAIR merger, this developer-level documentation does not yet exist. Thus, even though this guideline only refers to a description of the *changes* that have been made, supplying a detailed description of the work being done by the subprogram as a whole is perfectly OK, and even encouraged. `:-)`

Appendix — Supplying Internal Working Documentation

The following is a suggested form that code developers may use to supply the information required for the internal working documentation. Where possible, the version control software should not allow code changes to be made without the appropriate documentation.

1 – OVERALL DESCRIPTION

This section provides an overall description of the changes made, and is intended to establish a detailed history of the program's development.

Author: *Author of code*

Date: *Date of change*

Brief summary of changes:
A few lines summarizing the changes made

Detailed technical description of major changes:
For major additions and changes, please include a detailed description of the theory, analysis, etc., that was coded. Reference existing documents as appropriate.

2 – USER-LEVEL INFORMATION

This section will be used to update the user-level documentation.

New/modified keyword input:
List of keywords with definitions

New/modified standard output:
List of changes to standard (i.e., fort.6) output

New/modified input files:
List of files, including a description of the contents and format

New/modified output files:
List of files, including a description of the contents and format

New/modified error messages:
List of messages, including suggested corrective action

3 – DEVELOPER-LEVEL INFORMATION

This section will be used to update the developer-level documentation. The information described below should be included for each subprogram that has been modified or created. Subprogram names (or file names if necessary) should include the relative path name from the "src" directory. Changes to the lists of "called by" routines, input variables, etc., should note those that should be deleted, as well as those to be added.

Subprogram:
Subprogram name, and file name if different from subprogram name

Changes to "called by":
Changes to the list of subprograms that call this subprogram

Changes to "calls":
Changes to the list of subprograms that this subprogram calls

Changes to input variables:
Changes to the list of input variables that this subprogram uses, with their definitions

Changes to output variables:

Changes to the list of output variables that this subprogram computes, with their definitions

Changes to key local variables:

Changes to the list of key local variables in this subprogram, with their definitions

The information described below should be included for each common block that has been modified or created. Include file names should include the relative path name from the "include" directory. Changes to the lists of variables should note those that should be deleted, as well as those to be added.

Common block:

Common block name, and include file name if different from common block name

Changes to common variables:

Changes to the list of variables in this block, with their definitions