TERA

NUREG/CR-0807
LA-7812-MS

Informal Report

Quality Assurance for TRAC Development

University of California

282 256

LOS ALAMOS SCIENTIFIC LABORATORY
Post Office Box 1663 Los Alamos, New Mexico 87545

7907030434 K

POOR ORIGINAL

282 257

NUREG/CR-0807
LA-7812-MS
Informal Report

R-4

Quality Assurance for TRAC Development

J. M. Sicilian
R. J. Pryor

Manuscript submitted: April 1979
Date published: May 1979

Prepared for
Division of Water Reactor Safety Research
Office of Nuclear Regulatory Research
US Nuclear Regulatory Commission
Washington, DC 20555

NRC FIN No. A7016

UNITED STATES
DEPARTMENT OF ENERGY
CONTRACT W-7405-ENG. 36

282 258

QUALITY ASSURANCE FOR TRAC DEVELOPMENT

by

J. M. Sicilian and R. J. Pryor

ABSTRACT

The TRAC Computer Program Development Project has been subject to stringent quality assurance procedures since its inception. The three stages of the development project and the quality assurance procedures imposed at each stage are described in this report.

INTRODUCTION

Computer program development is far afield from usual applications of quality assurance. Therefore many standard quality assurance techniques are not applicable to the TRAC development task. It is desirable, however, to assure the quality of TRAC, and a number of steps are being taken toward that end. To understand these steps we must review the goals of the TRAC development project.

TRAC is a generic name for a sequence of computer programs that analyze the events which occur in a light water nuclear reactor system under accident conditions. The codes in this sequence share many models and methods, but address different types of accidents or reactor systems. All codes in the sequence attempt to model processes as accurately as possible, rather than using conservative approximations and are therefore known as "best estimate" programs.

The role of quality assurance in the TRAC project is to provide a framework for the development of TRAC to proceed as free of errors as possible, so that errors will be recognized when they are introduced and can be corrected with a minimum of effort and a reduced likelihood of introducing new errors.

There are three major stages of development inherent in the TRAC project: model and method development, code development, and code assessment. Quality assurance plays a part in each of these stages.

STAGE 1 - MODEL AND METHOD DEVELOPMENT

The modeling of physical processes and the methods of implementing these models are either developed by experts in the relevant fields at LASL or taken from the open literature. This assures an innate high quality. In addition, the models and methods are reviewed both by the Nuclear Regulatory Commission (NRC) staff and by the Advanced Code Review Committee. The models chosen are the most accurate available models compatible with the capability of modern computing systems. The methods of solution used in TRAC are chosen for their speed and accuracy and are thoroughly tested in stand-alone programs before their incorporation into TRAC.

STAGE 2 - CODE DEVELOPMENT

The code development stage of the TRAC project is the conversion of the methods and models developed in stage 1 into a computer program. This stage is both the most likely source of errors, since programming can be tedious and complex, and the stage most amendable to proceedures for quality assurance. Therefore, we have concentrated on providing quality assurance at this stage in the TRAC project.

There are four techniques used during the code development stage to assure the quality of TRAC. They are

a.) standard programming practices,

b.) automated program modification,

c.) program modification acceptance, and

d.) modification reproductibility.

The standard programming practices described below lead to a code structure that minimizes the likelihood of programming errors and simplifies the task of correcting errors that have been identified. TRAC development has proceeded via a top-down structured design since the project's inception. Subprograms and overlays within TRAC are assigned functional definitions. The programming of TRAC is modular in two senses. Individual routines treat

282 260

specific functions and/or specific reactor components. The data used by TRAC are stored in a modular fashion also. Modularity of data is efficient computationally and has a simplifying effect on TRAC programming. Internal consistency checks are used liberally throughout TRAC to identify programming errors. Finally, consistent use of variable names is enforced in TRAC by the use of universally defined COMMON areas through the UPDATE COMDECK convention.

Automated program modification, program modification acceptance, and modification reproducibility are inherent in the process used to produce TRAC. The LASL Central Computing Facility (CCF) is used extensively for the development of TRAC. This facility consists of numerous worker computers (including four CDC-7600 computers used by TRAC developers), a mass data storage system called HYDRA, and user terminals. Master copies of TRAC sources and binary files are stored on the HYDRA mass storage system. These files are backed up by storage on three media: disk, magnetic tape, and the PHOTOSTORE permanent storage machine. HYDRA also provides password protection for these files, thus only selected personnel are permitted to change the master copies of TRAC files. Individuals may access any of these files from their terminals to test modifications as described below.

Modifications to the master TRAC files are controlled by well-defined procedures implemented in a file maintenance program named HORSE.[1] This computer program forms the vehicle of TRAC code development. Modifications are made to TRAC by providing a description of the desired changes in the form of an "update library." An update library is a file which describes changes to be made subprogram by subprogram. HORSE extracts the subprograms to be changed from the master copy of TRAC source files, modifies each as described in the update library, and produces new source files, binary files, and an executable program. HORSE can also execute a series of problems if desired.

1 J. M. Sicilian, "Methods for the Development of Large Computer Codes Under LTSS," Los Alamos Scientific Laboratory report LA-NUREG-6828-MS (June 1977).

282 261

An individual TRAC developer works by creating update libraries and using HORSE to test the modifications. Once he is satisfied with the results he may submit the update for inclusion into the master copy of TRAC. This is accomplished by submitting an update submission form (Fig. 1) to the TRAC program custodian. The custodian tests the "update library" by executing a series of test problems and graphically comparing the results with experimental data. These test problems are designed to exercise TRAC and uncover errors introduced by the update without excessive computational cost. They consist primarily of sections of experiments driven by measured boundary conditions. Should the results of these test problems be acceptable, a new version of the master files is generated and stored on HYDRA. A procedure is followed by the code custodian to ensure that all necessary steps are taken. The procedure is defined in the checklist provided as Fig. 2. Periodically, a more involved procedure is utilized as described in Fig. 3. The permanent code version generated by this procedure is always readily available on PHOTOSTORE.

Modification reproducibility is implemented by periodically storing the TRAC master files and all update libraries on the PHOTOSTORE. Using these, any version of TRAC which was at one time accepted can be regenerated. To facilitate identification of the version of TRAC used for any problem, HORSE inserts identifying information into the TRAC source code, and TRAC prints this information at the head of its output file.

STAGE 3 - CODE ASSESSMENT

Although the quality assurance steps taken in stages 1 and 2 are essential to the development of TRAC, stage 3 is the final assurance that TRAC will provide a valuable tool to reactor safety researchers. Assessment of TRAC is performed by two groups of LASL personnel, as well as by offsite users. The LASL groups are known as the Developmental Assessment Task and the Independent Assessment Task. Developmental assessment provides continuing evaluation of TRAC during the development process by applying the master version of the code described above to many experimental situations. Results of comparisons to experiments are immediately available to the model developers to assist in the improvement of models and methods.

282 262

The Independent Assessment Task uses versions of TRAC that have been released to outside users to study other experiments. Results of these studies are also available to TRAC development personnel.

SUMMARY

In conclusion, we submit that the TRAC development project is implemented as a complex network of interacting processes that make optimum use of modern computer resources to assure the quality of the TRAC computer program. Figure 4 summarizes the process and accentuates the key quality assurance steps taken during TRAC development.

282 263

Update Submission Form

NAME: DATE:

LIX FILE NAME: HYDRA DATASET NAME:

PURPOSE OF THIS UPDATE:

NEW SUBPROGRAMS

NAMES

NEW TRANSFORM ALTER FILES

ALTER FILE NAME	BASE SUBROUTINE	NEW SUBROUTINE

CHANGES TO INPUT DATA:

ALTER FILES

SUBPROGRAM NAME	ALTER FILE NAME	CODE

282 264

Fig. 1.

CHECKLIST PROCEDURE
FOR
CREATING A NEW TEMPORARY VERSION

_______1. Shrink and save update library(s) on Q9LIB,HD under password.

_______2. Run HORSE (in library mode) to retrieve update libraries from Q9LIB, create new source and binary libraries from them, and then run all standard test problems from resulting controllee.

_______3. Check results of standard test problems to determine if update(s) can be accepted.

IF UPDATE IS ACCEPTED

_______4. Save update libraries in current UPD library on Q9LIB. Evict the update libraries from Q9LIB. (Caution - be sure to save update libraries in UPD library before evicting.)

_______5. Save the newly created source and binary libraries (FTRACT and BTRACT) on Q9LIB,HD under password.

_______6. If update(s) changed the input specifications, then update and replace the standard test problem library PTRACT on Q9LIB under password.

_______7. Save a source listing TRAC (version number), a controllee GO (version number), and a symbol table SYM (version number), all created by new update(s) on Q9LIB photostore.

_______8. Replace the compilation listing and standard test problem results in the TRAC-T binders.

_______9. If new version requires modifications to HORSE which are not compatible with the previous version, save the previous versions of HORSE and HORSLIB on Q9LIB,HP under file name HORS (previous version number) and HLIB (previous version number).

______10. If new version requires modifications to GRIT and GRED which are not compatible with the previous version, save the previous versions of GRIT, FGRIT, GRED, and FGRED on Q9LIB,HP under file names GRIT (previous version number), FGRIT (previous version number), GRED (previous version number), and FGRED (previous version number).

Fig. 2.

282 265

CHECKLIST PROCEDURE
FOR
CREATING A NEW PERMANENT VERSION

_______1. Shrink and save update library(s) in Q9LIB,HD under password.

_______2. Run HORSE (in library mode) to retrieve update libraries from Q9LIB, create new source and binary libraries from them, and then run all standard test problems from resulting controllee.

_______3. Check results of standard test problems to determine if update(s) can be accepted.

IF UPDATE IS ACCEPTED

_______4. Replace the source and binary libraries FTRACT and BTRACT with the newly created libraries on Q9LIB,HD.

_______5. Place the update libraries involved in the transition in the current UPD library. Save UPD (new version number -1) on Q9LIB,HD and on Q9LIB,HP.

_______6. When it is certain that the UPD library has been transferred from disk to a photostore cell, evict it from Q9LIB,HD.

_______7. Save the source, binary, and standard test problem libraries (FTRACN, BTRACN, and PTRACN) for the newly created version of Q9LIB,HP under the file names FTRAC (new version number), BTRAC (new version number), and PTRAC (new version number).

_______8. Save a source listing (TRAC), a controllee (GO), and a symbol table (SYMTAB), created by the new version on Q9LIB,HP under the file names TRAC (version number), GO (version number), SYM (version number.

_______9. Empty the TRAC-T binders.

______10. Replace the compilation listing and test problem results for the new version in the TRAC-N binders.

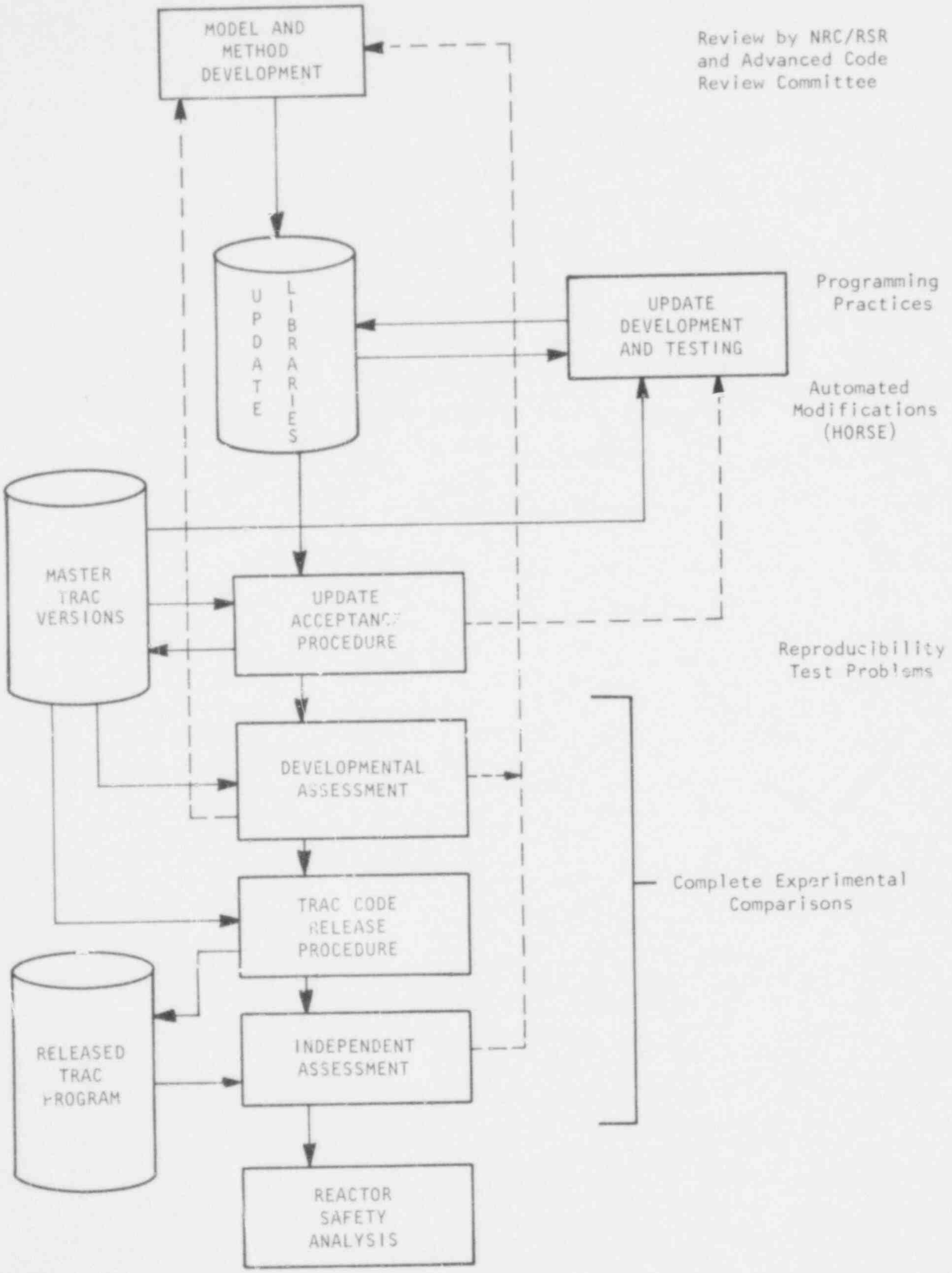
______11. Replace the INDEX listing with an index for the new version.

______12. If the new version requires modifications to HORSE which are not compatible with the previous version, save the previous version of HORSE on Q9LIB,HP under file name HORS (previous version number).

______13. If the new version requires modifications to GRIT and GRED which are not compatible with the previous version, save the previous versions of GRIT and GRED on Q9LIB,HP under file names GRIT (previous version number) and GRED (previous version number).

Fig. 3.

282 266



MODEL AND METHOD DEVELOPMENT
Review by NRC/RSR and Advanced Code Review Committee
UPDATE LIBRARIES
UPDATE DEVELOPMENT AND TESTING
Programming Practices
Automated Modifications (HORSE)
MASTER TRAC VERSIONS
UPDATE ACCEPTANCE PROCEDURE
Reproducibility Test Problems
DEVELOPMENTAL ASSESSMENT
TRAC CODE RELEASE PROCEDURE
Complete Experimental Comparisons
RELEASED TRAC PROGRAM
INDEPENDENT ASSESSMENT
REACTOR SAFETY ANALYSIS

Fig. 4.
TRAC development process.

282 267

DISTRIBUTION

	Copies
Nuclear Regulatory Commission, Bethesda, Maryland, R-4	283
Technical Information Center, Oak Ridge, Tennessee	2
Los Alamos Scientific Laboratory, Los Alamos, New Mexico	50

282 268