(2)

Task/Subtask ID52.1(2)
CDRL Sequence 05504-001
31 July 1993

SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS (STARS) PROGRAM

Cleanroom Engineering Handbook
Volume 1
Cleanroom Engineering Process Introduction and Overview

Contract No. F19628-88-D-0032
Task ID52 – STARS Technology Transfer Demonstration Project for the U.S. Army

Prepared for:

Electronic Systems Center
Air Force Materiel Command, USAF
Hanscom AFB, MA 01731-2816

Prepared by:

IBM Federal Systems Company
800 North Frederick Avenue
Gaithersburg, MD 20879

Task/Subtask ID52.1(2)
CDRL Sequence 05504–001
31 July 1993

SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS (STARS) PROGRAM

Cleanroom Engineering Handbook
Volume 1
Cleanroom Engineering Process Introduction and Overview

Contract No. F19628–88–D–0032

Task ID52 – STARS Technology Transfer Demonstration Project for the U.S. Army

Prepared for:

Electronic Systems Center
Air Force Materiel Command, USAF
Hanscom AFB, MA 01731–2816

Prepared by:

IBM Federal Systems Company
800 North Frederick Avenue
Gaithersburg, MD 20879

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	7/31/93	Initial

4. TITLE AND SUBTITLE

Cleanroom Engineering Handbook: Cleanroom Engineering Process Introduction and Overview

5. FUNDING NUMBERS

F19628-88-C-0032/0010

6. AUTHOR(S)

Ara Kouchakdjian
Richard H. Cobb
Alan R. Hevner
James A. Whittaker

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

IBM Federal Systems Company
800 North Frederick Avenue
Gaithersburg, MD 20879

SET, Inc.
2770 Indian River Blvd.
Vero Beach, FL 32960

8. PERFORMING ORGANIZATION REPORT NUMBER

05504-001
Volume 1

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Electronic Systems Center/ENS
Air Force Materiel Command, USAF
5 Eglin Street, Building 1704
Hanscom Air Force Base, MA 01731-2116

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

N/A

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Cleared for Public Release, Distribution is Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This is one of a series of six engineering handbooks prepared for and used by the engineering staff at Picatinny Arsenal for the STARS technology transfer demonstration. The handbooks define the engineering process and algorithms that will be used in Cleanroom projects. They are designed to provide support to trained engineers using Cleanroom Engineering, not to substitute for training.

This handbook, Volume 1, presents an introduction to the Cleanroom engineering process model and is recommended for all software planning and development participants. It defines the Cleanroom process in light of the project organization which consists of:

- The Specification Team
- The Development Team
- The Certification Team

and the software transformations which are:

- Specification Development - To define the software system to be developed and record the definition in the specification document.
- Software Developement and Certification - To develope and certify software in the target language that runs on the target processor and exhibits the same behavior described in the specification.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Certification, Cleanroom, Cleanroom Engineering, Development, Management, Software Development, Specification	45
	16. PRICE CODE: N/A

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

PREFACE

This series of handbooks is prepared for use by managers and engineers assigned to Cleanroom projects at Picatinny Life Cycle Software Engineering Center.

These handbooks define the engineering process and algorithms that will be used in Cleanroom projects.

This document was developed by the IBM Federal Systems Company, located at 800 North Frederick Avenue, Gaithersburg, MD 20879 and Software Engineering Technology, Inc. located at 2770 Indian River Boulevard, Vero Beach, FL 32960. Questions or comments should be directed to Mr. Paul Arnold at 301-240-7464 (Internet: pga@sei.cmu.edu).

CLEANROOM ENGINEERING PROCESS

TABLE OF CONTENTS

CLEANROOM ENGINEERING PROCESS

SECTION 1: READERS' GUIDE TO THE SET CLEANROOM ENGINEERING

The document is divided into 6 volumes. Each volume has a distinct purpose, as described in the titles below:

Volume 1 - Cleanroom Engineering Process Introduction and Overview
Volume 2 - Organization and Project Formation In The Cleanroom Environment
Volume 3 - Project Execution In The Cleanroom Environment
Volume 4 - Cleanroom Software Specification
Volume 5 - Cleanroom Software Development
Volume 6 - Cleanroom Software Certification

Each of the volumes takes a portion of the Cleanroom process model presented in Volume 1 and presents a concrete prescription for the completion of a particular task. The section that is an exception to this rule is Volume 2, which also discusses activities that precede the commencement of the Cleanroom project, that is, the commencement of the process model presented in Volume 1. The matrix below provides a suggested approach to reading the remainder of the volumes for individuals who have the following roles in a Cleanroom project. Req'd represents required reading, while Opt'l represents optional reading.

	S/W Eng Mgr	Spec. Team Ldr	Spec. Team Engr	Dev. Team Ldr	Dev. Team Engr	Cert. Team Ldr	Cert. Team Engr
Vol 1	Req'd	Req'd	Req'd	Req'd	Req'd	Req'd	Req'd
Vol 2	Req'd	Opt'l		Opt'l		Opt'l	
Vol 3	Req'd	Req'd		Req'd		Req'd	
Vol 4	Req'd	Req'd	Req'd	Req'd	Req'd	Req'd	Req'd
Vol 5	Req'd			Req'd	Req'd		
Vol 6	Req'd					Req'd	Req'd

Each volume will provide instruction on how to complete a particular step in the Cleanroom process. Additionally, templates will be provided for any documentation that is to be created as a part of a process step.

SECTION 2: ENGINEERING PROCESSES: AN INTRODUCTION

2.1 Objective

The objective of this series of handbooks is to guide the specification, development and certification of software by applying an engineering process. The engineering process has been named "Cleanroom Software Engineering" by its principal inventor, Dr. Harlan D. Mills.

As we enter the last decade of the twentieth century, society has been developing software for less than a human generation. In that short time, a vigorous commercial practice for software development has emerged. This practice is based on a growing body of craft practices which produce good results for some practitioners. Software practitioners want to become software engineers so they can efficiently design a product that operates flawlessly in fulfilling its assigned missions just as civil, mechanical, chemical and electrical engineers perform in their respective area of competence. The organizations for which the software practitioners work also desire to form effective software engineering organizations so software solutions can be engineered with to meet requirements within schedule and budget. A transformation from craft to engineering will accomplish these goals. These handbooks are designed to help people and organizations make this transformation.

2.2 Developing An Engineering Practice

It takes more than desire to create an engineering practice.

> **Engineering** is the discipline concerned with putting scientific knowledge to practical use.

In the case of software engineering, practical use is clear, we want to build (that is specify, develop and certify) software that correctly fulfills an assigned mission. In order to develop a software engineering practice, we need to answer the same questions all other engineering disciplines have answered. The questions are:

- What scientific knowledge do we put to use?
- What algorithms and engineering tasks do we use to put the scientific knowledge to use?
- What process do we follow in order to efficiently apply the required algorithms and engineering tasks in order to obtain the desired software?

In the following paragraphs, we will briefly discuss the answers to each of these questions.

2.3 Science Base

A program or a system of programs is a mathematical object. It is a rule for a mathematical function. The specification for a program must define a function that fully and completely describes the behavior that is required for the software to fulfill its assigned mission. Software specification is performed by finding and recording such a function. Software development is performed by developing a correct procedure (rule) for the function defined in the specification. Software certification is performed by designing, conducting and evaluating an experiment to test the hypothesis that the software exhibits the same behavior as the behavior defined by the specification function.

It follows that the science base for software engineering must be derived from the mathematics. Fortunately this effort is very well advanced. The books and papers which define this science base are included in the Cleanroom Reference List included as Section V in this volume.

Specification and development engineers need only work with three classes of functions:

Black Box
State Box
Clear Box

and algebraic processes for expanding

a Black Box from a requirement
a State Box from a parent Black Box
a Clear Box from a parent State Box or a parent Clear Box

or for abstracting

a function from a given procedure (rule).

Certification engineers must define the probability density function that describes how the software will be used by each class of users. A Markov model is used for this purpose. The model is used to generate a representative random sample of use from which to determine sample statistics. These sample statistics can be used to estimate population statistics. Certification engineers make extensive use of statistical decision theory and sampling theory.

2.4 Algorithms, Procedures and Processes

Engineers apply algorithms and perform engineering tasks in applying the science base to develop a solution, in our case, to create software.

> An **algorithm** is a systematic method for solving a well-defined problem.
>
> A **procedure** is a set of heuristics that often rely on engineering judgement for solving a problem.

The distinction being made in the above definitions is that an algorithm defines a set of rules that can be demonstrated to find a solution to a given problem while a procedure is a set of rules that

rely on a large amount of engineering judgement and creativity to find the desired result. In this context algorithms are more powerful than procedures.

The term **engineering task** is used in the balance of this handbook to collectively refer to algorithms and procedures.

In creating software, engineers use engineering tasks to expand and abstract functions, to show that two functions define the same behavior, to define the probability density function for software usage, to create a random, representative sample of software use from a given probability density function and to estimate parameters for the user population.

Fortunately, a large body of useful engineering tasks exist that are useful to the Cleanroom Engineer. Many of these engineering tasks are summarized in these handbooks.

The dictionary defines a **process** as a particular set of steps or operations for doing something.

Many different adjectives are applied to processes to indicate the characteristics of the set of steps being followed. The software field has the waterfall process, the rapid prototyping process, a defined process and many others.

The goal in this volume is to define software engineering process, i.e., a process that can be used to guide the engineering of software solutions.

> An **engineering process** is the set of process steps that an engineer uses to guide his/her creativity in developing a solution to a given problem.

Necessary conditions for defining an engineering process, including a software engineering process, are:

- The control flow among process steps is defined with sequential, iterative, alternation and concurrent operations.
- The data flow into and out of each process step must be specified.
- The definition of each process step must identify the engineering task that is used to perform the process step including any tools that are used to assist the engineer in performing the engineering task.

In short, an engineering process is a program. The program controls parallel processing in order to coordinate the parallel activities of the several engineers working on the project. The program does not execute on a computer. The program executes in real time by each of the engineers and their managers invoking, terminating and coordinating their activities as required by the process. A major task of first line engineering managers is to insure that all of the engineers are assigned to perform engineering tasks in accordance with the engineering process that is being used to

control the project. They do this by making and executing dispatching decisions. Procedures for implementing the dispatching decision in the Cleanroom environment are presented in Volume 3 of this Handbook series.

A generic Cleanroom Engineering process is presented in Section IV. This process is created top-down from the science base using the algorithms and engineering operations developed from the science base. The process is developed by starting with a black box which defines the software development activity in an implementation free format. Subsequently, a state box is designed to specify state data which can be maintained to act as a surrogate for the stimulus histories mapped by the black box function. After that, a clear box is designed from the state box. The clear box was refined. This work is documented in The Cleanroom Engineering Software Development Process, CDRL 7001, The DARPA STARS Program, February 1991. The result is the clear box procedures that is presented in Section IV.

In the Cleanroom engineering process, the actual activities that are invoked are represented by a numbered activity. A numbered activity is either a process, an engineering task. The leaf activities are engineering tasks. Higher level activities are processes. Processes invoke engineering tasks.

A work breakdown structure can be developed from an engineering process model. A work breakdown structure is, in effect, a tree which presents the hierarchial relationship among activities. Activities that are not leaf nodes in the tree are processes. Activities that are leaf nodes in the tree are engineering tasks. Processes are used to organize individuals and information. Engineering tasks use individuals or other agents to effect change to information, by creating new documents (files) or modifying old ones.

It is useful to develop work breakdown structures for differing levels of granularity. For a planning level, less granularity is desirable, while more granularity is required to support dispatching. This is because dispatching decisions require complete visibility to every task that needs to performed on a project. Planning requires a model that enables people to make higher level decisions like the allocation of resources to a project. The ideal planning model abstracts out all the inessential details so the decision maker can concentrate on the essential details. Work breakdown structures that define the levels of granularity required for planning and scheduling are presented in Section IV.

2.5 Cleanroom Engineering

Cleanroom Engineering is the name given to describe the first and, to date, the only software engineering process defined which enables organizations to deploy engineering teams that build software that has a high probability of being able to perform its assigned mission in a flawless manner.

This handbook series provides details of the Cleanroom algorithms, procedures and the Cleanroom Engineering process.

SECTION 3: CLEANROOM ENGINEERING PROCESS CONCEPTS

In this section, several concepts are discussed that assist in understanding the Cleanroom process model which is presented in section IV.

3.1 Project Organization

Developing software presents difficult mental challenges. It has been found that this type of engineering is best performed by a team of engineers. Cleanroom projects are organized using three teams as follows:

Specification The Specification Team is responsible for defining a software system that is capable of fulfilling its assigned mission and recording the definition in the specification document.

Development The Development Team is responsible for developing ,in the specified time cycle, a system of programs in the target language that will execute on the target processor. The system will exhibit exactly the same behavior defined in the specification.

Certification The Certification Team is responsible for conducting measurements and then using the measurements to determine (1) the probability that the software will perform correctly (that is, its behavior will be the same as the behavior defined by the specifications) when it is deployed and (2) that the Development Team is continuing to operate within the organization's quality guidelines.

Each team has a leader who is responsible for

- providing intellectual leadership to the team,
- assigning tasks to team members,
- coordinating team reviews,
- providing status reports which include an accurate assessment of the time remaining for the team to complete its current assignment,
- requesting help when it is required, and
- participating in all of the technical aspects of the team, as would any other team member,

Team leaders are responsible to the Software Engineering Manager (SEM) for the project. The SEM is responsible for the project. The SEM is an active project participant and may be also be one of the team leaders.

The Software Engineering Manager is responsible for

- the successful conclusion of the project in terms of budget, schedule and quality,
- interfacing with the customer,
- the well being of the project staff,
- training and coaching of the project staff,
- providing status reports which include an accurate assessment of the time remaining to complete the project, and
- requesting help when it is required.

3.2 Software Project Context

The projects we are discussing, in this series of handbooks, are software projects. Software projects are usually part of a larger project. The larger project typically has responsibility for developing a system to fulfill some objective. In many such projects, one or more of components may be software. In this series of handbooks the focus is on software projects.

Software projects can take on many different forms. Some typical examples taken from Picatinny are:

COFT The M2/3 A1 Instructional Conduct Of Fire Trainer (I-COFT) is a hardware and software system that provides training for a driver and a gunner in the use of a Bradley Fighting Vehicle (BFV). It simulates the use of a BFV in various combat situations in order to properly develop skills for the crew. The I-COFT's software is written in Fortran and is divided into 38 subsystems, each of which handles some amount of functionality. The system is meant to provide as realistic as possible simulation of the use of the equipment, since simulation is much more cost effective than use of the actual vehicles. The I-COFT is updated as a part of a Software Block Update (SBU)^Xxwhere the software is modified to handle new requirements and correct any existing failures.

MBC The Mortar Ballistic Computer (MBC) is a hardware and software system that assists a fire direction center to provide gun orders in conducting up to three active fire missions for up to three sections of six mortars each. The conduct of the fire mission includes the orchestration of the multiple mortars and the calculation of the ballistics information necessary to fire. The software is currently written in Assembler and DTL, a panel definition language. The

MBC is to be reengineered, with additional functionality, in Ada, to allow for a portable implementation that will function on a variety of hardware platforms.

Experience indicates it is not possible to define a uniform starting point for a software project. A software project starts whenever someone authorizes it to start. This may be an early stage when very little is known or at a late stage when a great deal is known. Therefore, the first duty of the software project team is to determine what is known and what needs to be done to develop a specification for the desired software.

The mission that the software is to fulfill may be very well stated so there is little ambiguity about what the software needs to do. In other cases, the mission assigned to software will not be stated so uncleanly that there is a great amount of ambiguity about what the software is to do.

Whatever the status when a software project is initiated, the first step is to perform sufficient analysis to understand the problem and solution domains, invent the stimuli and responses and the desired behavior for the software. The behavior is documented as a function which defines when each response is produced in terms of stimulus histories (including initial conditions) that have been received by the software.

A specification for a system is not usually a specification for software. Systems are typically composed of people, hardware devices and software components. System stimuli are rarely software stimuli and software responses are rarely system responses since software components are rarely at the boundary of the system. Typically, software components receive their stimuli from other system components and provide the responses to other system components. Therefore, even in cases where there is a very good specification for a system, the specification for each software component that is in the system must be prepared.

3.3 Two Transformations

A typical software project is composed of two major transformations as follows:

Specification Development

To define the software system to be developed and record the definition in the specification document.

Software Development and Certification

To develop and certify software in the target language that runs on the target processor and exhibits the same behavior as the behavior described in the specification.

The first transformation is the responsibility of the Specification Team. The second transformation is the responsibility of the Development Team to develop the software and the Certification Team to certify the software.

The format of the specification volumes and the algorithms and engineering tasks that the Specification Team uses to develop the specification are the subject of Volume 4 of this series. Volumes 5 and 6 define the algorithms and engineering tasks used by the Development Team and Certification Team, respectively.

3.4 Project Spirals

Typically, large projects are decomposed into a series of smaller projects or spirals. Projects are subdivided into spirals for two reasons: (1) to reduce risk by establishing intermediate assessment points so projects can be redirected as required and (2) to provide intermediate deliveries.

Figure 3.1 contains a diagram that represents a graphical model of a software project being developed in spirals. This model defines four quadrants that represent distinct phases of each spiral. The four quadrants of a spiral are:

I Planning for the cycle including risk analysis and objective setting for future cycles.
II Software Specification Preparation including problem and solution domain analysis and the preparation of prototypes.
III Software Development and Certification.
IV Analyzing the results of the cycle including software demonstration, analysis of results and projection of possibilities.

The specific software phases to which the handbook applies are II and III. The process as defined in Section IV of this document is for one spiral.

3.5 Construction Plan

Software projects progress with many activities occurring in parallel. Thus any model of a software project must accommodate the possibility of many activities preceding in parallel. In fact, a project manager wants to organize events to occur in parallel, to minimize cycle time. In the process model, **con** ... **noc**'s are used to delimit the parallel performance of activities.

In a Cleanroom project, the basic flow is to define the specification and then follow the specification with parallel activities as follows:

1. The Development Team develops software increment by increment in accordance with the construction plan.

2. The Certification Team certifies the software for each accumulation of increments in accordance with the construction plan.

Figure 3.2 shows a typical construction plan for a software project. The construction plan decomposes the system level black box which defines overall system behavior into a series of increments which accumulate into the full system. The increments are defined and ordered so

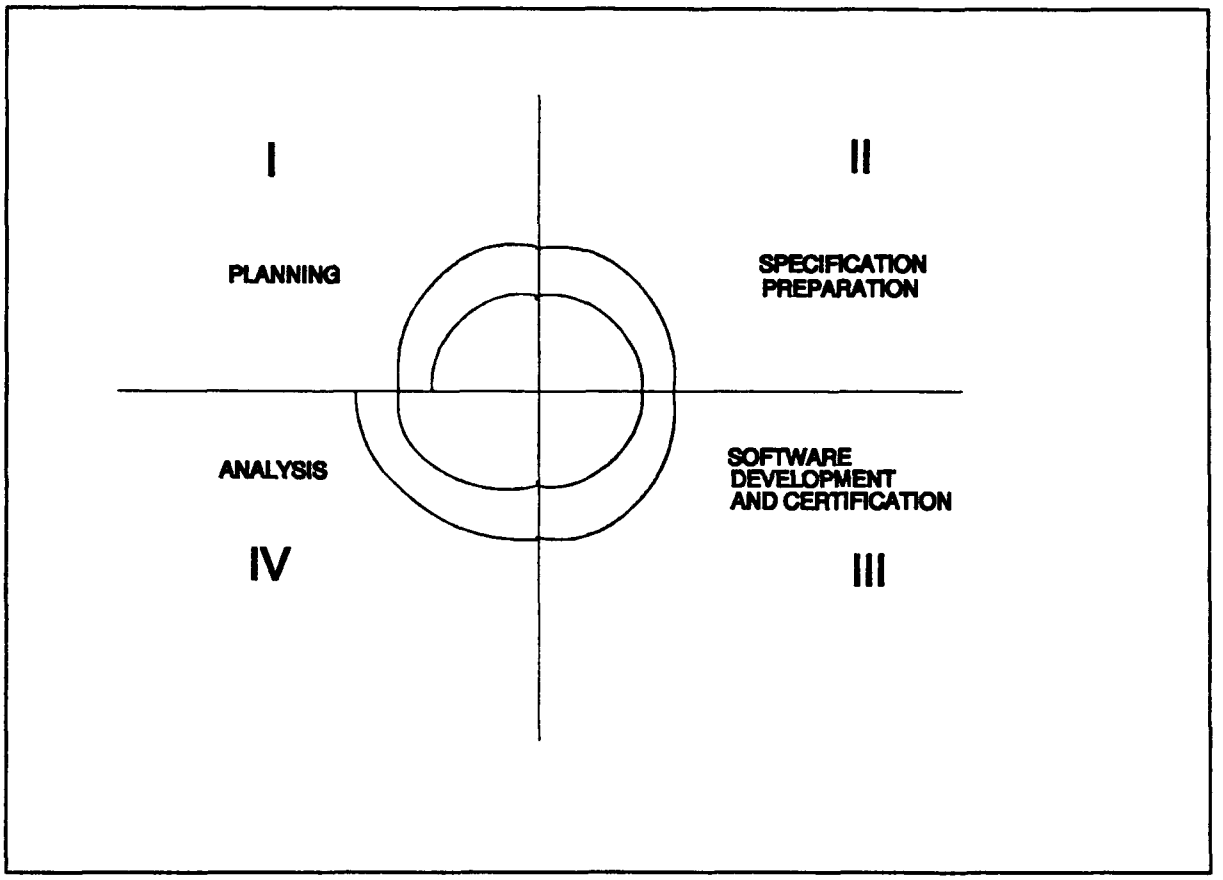

Figure 3.1: Spiral Model

10,000 lines of code. In this way, the Development Team completes an increment every 4 to 6 weeks.

For each increment, the Specification Team decomposes the specification to represent just that increment. The Development Team then develops and verifies the code. When the Development Team is satisfied that the software for the increment is correct they turn the software over to the Certification Team for independent testing.

The Certification Team prepares test cases for the corresponding accumulation of increments. The Development and Certification Teams each do their work based on the same specification.

The Certification Team compiles the software and begins to execute the test cases which have been prepared previously. If the Certification Team encounters any failures, they prepare a failure report. They may batch failure reports, but at some time they turn the reports over to the Development Team. The Development Team modifies the software and prepares an Engineering Change Notice which is sent to the Certification Team. The Certification Team applies the correction to the software. The process continues until the increment is accepted or it is rejected.

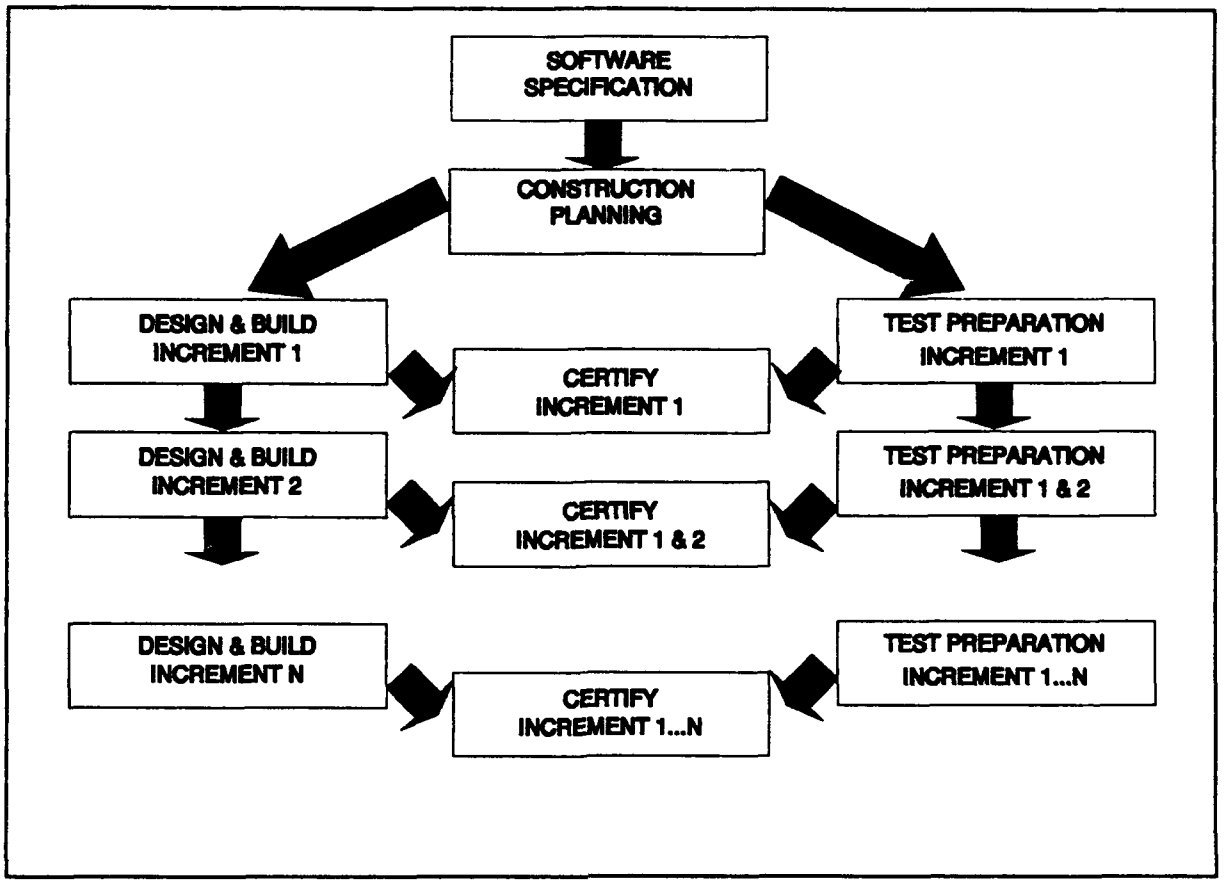

Figure 3.2 Typical Construction Plan

While the testing is progressing the Development Team starts to develop the next increment. The arrangement is more complex but works the same if the software is so large that multiple Development Teams are required.

3.6 State Data Repository

The engineering teams produce documents and files that reflect their work efforts during the course of the project. In Cleanroom, all work is documented in an organized top-down manner. One goal of Cleanroom engineers is to produce a set of documents and files that represent the design trail in such a form that they can be regarded as public literature.

These files and documents at any point in time represent the current state of the project. Therefore, they are referred to collectively as the project state data. All project files are kept in the project State Data Repository (SDR). Private files are not required. All project members and other system stakeholders have access to project files. Write access to the state data repository is controlled to maintain repository integrity. Public availability permits peer audits, which lead to a better product.

The highest level of organization of the project state data is defined in the following table. One important organizing task for each project is to define all the details of the state data repository files.

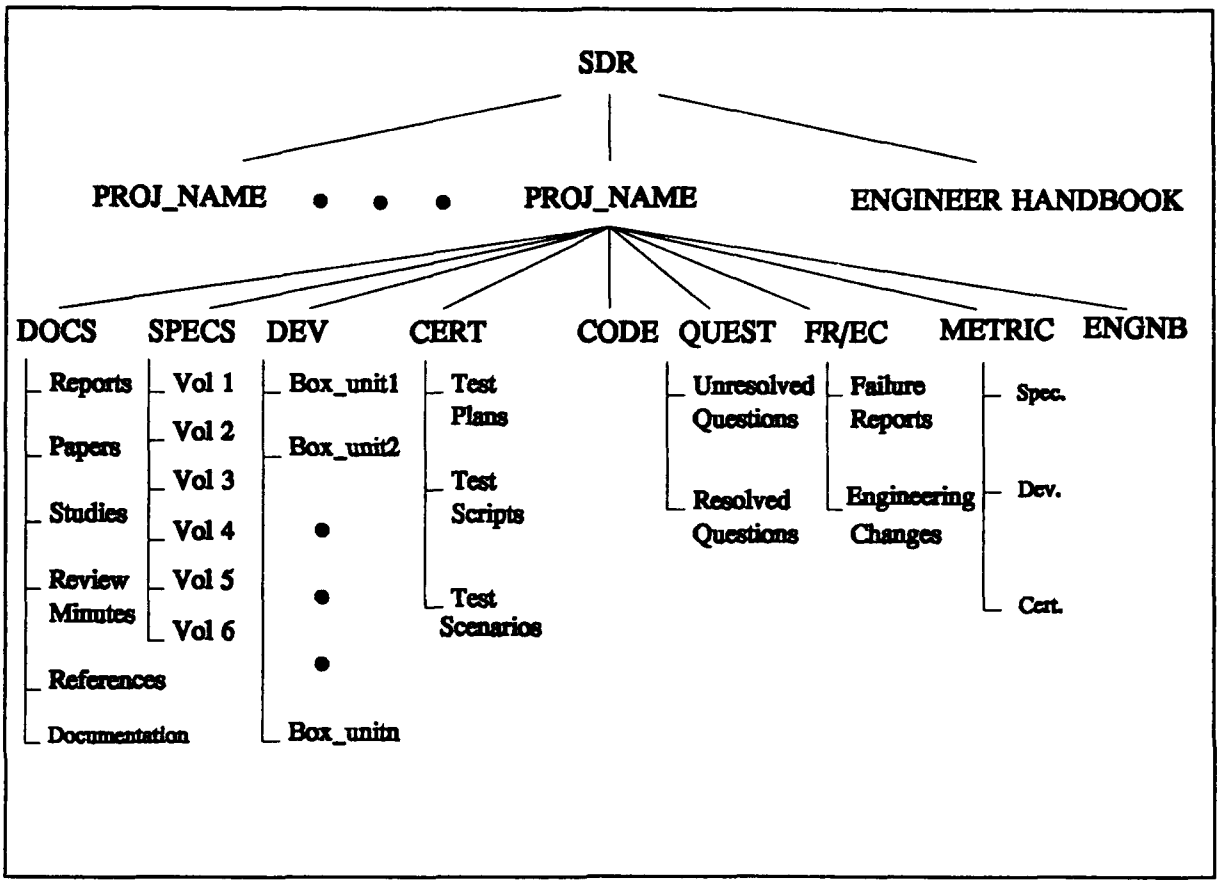

Figure 3.3: SDR Directory for Cleanroom Project Proj_name

3.7 Replanning

Unforseen events occur outside a project which cannot be ignored. For example, a new representative is appointed by the customer to oversee the project. He or she wants something different. Events also go wrong on a project. For example, an earlier design decision which at the time appeared to be a good decision turns out to cause a later problem, so it is necessary to redesign a major part of the system which will delay completion. All events which impact the schedule or require outside approval invoke a replanning effort. In some cases, the result of a replanning effort will be a decision to terminate the project.

There are many stages in a project where it may become necessary to replan. All these possibilities need to be identified in the project process model. A typical decision of a replanning effort is to recycle back to an earlier process step to redo the work from a different perspective. These recycle possibilities must be shown in the project process model.

There are many stages in a project where it may become necessary to replan. All these possibilities need to be identified in the project process model. A typical decision of a replanning effort is to recycle back to an earlier process step to redo the work from a different perspective. These recycle possibilities must be shown in the project process model.

It is the recycling in response to unforseen events that has the most dramatic impact on project completion estimates. Therefore, procedures used to estimate project completion dates must take into account the possibility of replanning and then recycling. The probability of taking different project paths following a replanning effort must be estimated and considered by the procedure used to estimate expected project completion dates. A procedure for doing this is discussed in Volume 3.

SECTION 4: A GENERIC CLEANROOM ENGINEERING PROCESS

A generic Cleanroom process model for use by software engineers and managers at Picatinny is defined in Figure 4.1. This generic process model serves as a basis for developing project process models. The specific project process model is contained in the project Software Development Plan. This activity is discussed in Volume 2.

The generic Cleanroom process model has the following features:

- Provides for modeling a project conducted in **spirals** to reduce risk or to facilitate evolutionary development and sequential delivery.
- Provides for developing and recording specifications in **cycles**.
- Provides for developing software in **increments**.
- Provides for certifying software in **accumulations** of increments.
- Changes the numbering scheme for tasks to facilitate use on a project. The scheme used is:
    - A control process is denoted by a P. A process is invoked by using a **run** statement.
    - An engineering task is denoted by a S, D or C depending on which engineering team is responsible for the task. Additionally, M denotes that management is responsible for the task. All tasks are **do** ... **until** task Completion Conditions achieved **od**;
    - Processes are numbered with a nomenclature that permits complete traceability of each process to all of its parents. For example the task D5.j.3.2 is the second of the tasks performed by the Development Team in carrying out process D5.j.3 which is the third control structure in developing increment j.

Managers, when preparing project planning and master project schedules only take into account processes and subprocesses for specification cycles, development increments and certification accumulations.

Team leaders are responsible for dispatching tasks within a subprocess for cycles, increments and accumulations. In this way, the model used by project managers and planners is at the right level

of aggregation for useful planning and team leaders have all the information they require to perform task dispatching and monitor team performance.

This generic process model must be tailored for each individual project to which it is applied. Of course some projects will be similar so there will be opportunities for reuse. Project models are placed in the organization's process library to promote reuse of appropriate portions of the model.

The process model can be used to directly develop two work breakdown structures for Cleanroom projects. The first is a work breakdown structure that should be sufficient for planning. The second work breakdown structure is used to support dispatching decisions made by team leaders as they assign work to team members to develop the product in accordance with the plan. This same structure is also used to guide the collection of data to support the calculation of metrics.

The two generic Work Breakdown Structures are presented in Figures 4.2 and 4.3.

This model must be tailored for each project to which it is going to be applied. For this purpose, a soft copy of the model is provided with the handbook. Three types of tailoring are required as follows:

- The first is to parameterize the model for the anticipated number of specification cycles and the expected number of increments.

- The second is to change the processes in the model to reflect the exact nature of the project.

- The third is to include in the model the external reviews that are scheduled for the project.

The tailored model is included in the project software development plan. It is used by the Software Engineering Manager and team leaders to guide the project to a successful conclusion. During project replanning activities it is likely that the process model will need to be refined.

The generic Cleanroom process model serves to drive the organization of the remainder of the engineering handbook volumes.

```
proc P0: Cleanroom Process for Spiral s
[This process conti 'ls one project spiral.]
   do [P0: Cleanroom process for Spiral s]
      [P1 results in the organization of staff and resources for performing the spiral
      (project).]
      run  P1: Spiral (Project) Invocation;
      con
         [P2 provides information management services for the project/cycle, where
         questions and issues, as well as externally received stimuli, are handled and
         stored.]
         run  P2: Information Management;
         [P3 results in modified or published spiral (project) plans, spiral (project) reviews
         or the submission of a spiral (project) status report.]
         run  P3: Program Management;
         do
            [P4 results in specifications for the software to produced in this spiral
            (project).]
            run P4: Specification Preparation;
            [P5 and P6 result in software certified to be in conformance with its
            specifications.]
            while
               project spiral not terminated and plan still has more increments to be
               certified
            do
               con
                  [P5 results in the development of the software for the spiral (project).]
                  run  P5: Software Development;
                  [P6 results in the certification of the software for the spiral (project).]
                  run  P6: Software Certification;
                  run  P7: Specification Configuration Management;
               noc;
            od;
         od;
      noc;
      [P8 results in the preparation of all of the project deliverables to all of the project
      stakeholders]
      run P8: Product (Project) Releases;
   od;
corp;
```

Figure 4.1(a) Generic Cleanroom Process Model **proc** P0: Cleanroom Process for Spiral s

```
proc P1: Spiral (Project) Invocation
   [This process results in the organization of the staff and resources required to perform
   the spiral (project).]
   do [P1: Spiral (Project) Invocation]
      con
         M1.1:   Allocate and organize initial project staff;
         M1.2:   Allocate and organize other project resources;
         M1.3:   Agree on project charter;
         M1.4:   Tailor software development plan for project/spiral;
         M1.5:   Initialize metrics collection activities;
         M1.6:   Resolve all questions and issues;
         M1.7:   Archive external information received;
      noc;
   until
      Completion Conditions achieved for tasks M1.1 through M1.7
   od;
corp;
```

Figure 4.1(b) Generic Cleanroom Process Model **proc** P1: Spiral (Project) Invocation

```
proc P2: Information Management
   [This process provides information management services for the project/cycle]
   do [P2: Information Management]
      con
         [2.1-2.4 address the resolution of questions for each of the teams, while 2.5-2.8
         address the transition into state data of externally received information.]
         S2.1:   Resolve questions and issues;
         D2.2:   Resolve questions and issues;
         C2.3:   Resolve questions and issues;
         M2.4:   Resolve questions and issues;
         S2.5:   Archive external information;
         D2.6:   Archive external information;
         C2.7:   Archive external information;
         M2.8:   Archive external information;
      noc;
   until
      P8 complete and all questions and issues resolved
   od;
corp;
```

Figure 4.1(c) Generic Cleanroom Process Model **proc** P2: Information Management

```
proc P3: Program Management
   [This process results in modified or published spiral (project) plans, spiral (project)
   reviews or the submission of a spiral (project) status report.]
   do [P3: Program Management]
      for  each project time slice
      do
         con
            if Review Scheduled or Major Problem?
            then
               case
                  review is
               part (review name 1)
                  run P3.1.k:   Initiate Project Review k, k=review name 1;
               part (review name 2)
                  run P3.1.k:   Initiate Project Review k, k=review name 2;
                .
                [one part needs to be created for each planned review]
                .
               else [Major Problem]
                  run P3.1.k:   Initiate Project Review k, k=Major Problem;
               esac;
            fi;
            if Status Scheduled?
            then
               run P3.2.t:   Initiate Prepare and Submit Status Report t, t=1,...;
            fi;
            if project team needs to be enhanced or changed
            then
               M3.3:   Organize project staff as required;
            fi;
         noc;
      od;
   until
      Completion Conditions for all initiated processes are achieved
   od;
corp;
```

Figure 4.1(d) Generic Cleanroom Process Model **proc** P3: Program Management

```
proc P3.1.k:  Initiate Project Review k, k=1,...
   do
      M3.1.k.1:  Prepare Review;
      M3.1.k.2:  Conduct Review;
      M3.1.k.3:  Issue Directives to Implement Review Findings;
   until
      Completion Conditions achieved for M3.1.k.3
   od;
corp;
```

Figure 4.1(e) Generic Cleanroom Process Model **proc** P3.1.k: Initiate Project Review k, k=1,...

```
proc P3.2.t:  Initiate Prepare and Submit Status Report t, t=1,...
   do
      con
         M3.2.t.1:  Prepare and Submit Report t;
         M3.2.t.2:  Monitor Metrics Collection;
      noc;
   until
      Completion Conditions achieved for combination of M3.2.t.1 and M3.2.t.2
   od;
corp;
```

Figure 4.1(f) Generic Cleanroom Process Model **proc** P3.2.t: Initiate Project Review k, k=1,...

```
proc P4: Specification Preparation
   [This process results in specifications for the software to produced in this spiral
   (project).]
   [P4: Specification Preparation]
   while
      (project spiral not terminated and plan still has more cycles to be completed) or not
      (M4.i.5.2 management decision was specification suitable to initiate development)
      [M4.i.5.2 results in one of the following decisions: Specifications ready, another
      specification cycle is necessary, stop specifications and replan.]
   do
      [P4.i handles the investigations for (including prototyping), preparation of, and
      appraisal of the specifications, resulting in the specifications for a project/spiral
      when all cycles are complete.]
      run  P4.i: Software Specification for Cycle i; [where i is the next cycle in the
           sequence i= 1...i...m according to the project plan]
      if M4.i.5.2 management decision was specification problems-replan project or (this
         is last cycle in plan and not M4.i.5.2 management decision was specifications
         suitable to initiate development)
         [M4.i.5.2 results in one of the following decisions: Specifications ready, another
         specification cycle is necessary, stop specifications and replan.]
      then
         [PCS1: is a common service procedure which results in an updated Software
         Development Plan or a decision to terminate current project/spiral.]
         run  PCS1: Replan project/spiral;
         if MCS1.4 management decision indicates need to terminate project spiral
         then
            terminate current spiral
         fi;
      fi;
   od;
corp;
```

Figure 4.1(g) Generic Cleanroom Process Model **proc** P4: Specification Preparation

```
proc P4.i: Software Specification for Cycle i
   [P4 handles the investigations for (including prototyping), preparation of, and appraisal
   of the specifications, resulting in the specifications for a project/spiral.]
   do [P4.i: Software Specification for Cycle i]
      run P4.i.1:    Prepare Plan for Cycle i;
      for
         Allocated time period
      do
         con
            [P4.i.2 studies the problem domain to guide system design]
            run P4.i.2:    Problem Domain Analysis-Cycle i;
            [P4.i.3 studies the solution domain to guide system design including searching
            for reuse opportunities]
            run P4.i.3:    Solution Domain Analysis-Cycle i;
            [P4.i.4 a complete preparation cycle through the six volume specification]
            run P4.i.4:    Prepare Cycle i Specifications;
         noc;
      od;
      run P4.i.5:    Appraise Cycle i Specifications;
   od;
corp;
```

Figure 4.1(h) Generic Cleanroom Process Model **proc** P4.i: Software Specification for Cycle i

```
proc P4.i.1:   Prepare Plan for Cycle i
   do [P4.i.1: Prepare Plan for Cycle i]
      con
         M4.i.1.1:   Establish Objectives For Cycle i;
         M4.i.1.2:   Allocate Time Period To Cycle i;
         M4.i.1.3:   Prepare Plan For Cycle i;
      noc;
   until
      Completion Conditions achieved for combination of M4.i.1.1, M4.i.1.2 and M4.i.1.3
   od;
corp;
```

Figure 4.1(i) Generic Cleanroom Process Model **proc** P4.i.1: Prepare Plan for Cycle i

```
proc P4.i.2:  Problem Domain Analysis-Cycle i
  do  [P4.i.2:  Problem Domain Analysis-Cycle i]
    con
      for each analysis method in plan
      do
        case [select the appropriate analysis method]
          analysis method is
        part  Information Collection [collect and document required information]
          S4.i.2.1:    Information Collection;
        part Reverse Engineering    [develop black box function for each system]
          S4.i.2.2:    Assemble Information for System(s)
          for   each system in need of reverse engineering
          do
            case
              existing system type is
            part Manual
              S4.i.2.3:    Manual System Reverse Engineering;
            part Automated
              S4.i.2.4:    Automated System Reverse Engineering;
            part Hybrid
              S4.i.2.5:    Hybrid System Reverse Engineering;
            esac;
          od;
        part Black Box Models   [develop black box function model for system to be
                                 built, document in Specification volumes 2 and 3]
          S4.i.2.6:    Develop and Analyze Black Box Model;
        part Markov Usage Models [develop Markov usage model for system to be
                                  built, document in Specification volume 5]
          S4.i.2.7:    Develop and Analyze Markov Usage Model;
        esac;
      od;
    noc;
  until
    sufficient Completion Conditions achieved for all selected analysis methods and
    (time period up or sufficient understanding has been obtained)
  od;
corp;
```

Figure 4.1(j) Generic Cleanroom Process Model **proc** P4.1.2: Problem Domain Analysis-Cycle i

```
proc P4.i.3:  Solution Domain Analysis-Cycle i
  do  [P4.i.3:  Solution Domain Analysis-Cycle i]
    con
      for each analysis method in plan
      do
        case [select the appropriate analysis method]
          analysis method is
        part Information Collection [collect and document required information]
          S4.i.3.1:    Information Collection;
        part Reuse Analysis   [consider reuse options effect on system behavior]
          do
            S4.i.3.2:    Browse Reuse Repositories To Find Match For Needs;
            S4.i.3.3:    Prepare Cost/Benefit Analysis;
            S4.i.3.4:    Select Potential Reuse Modules for Integration into System;
          until
            Completion Conditions achieved for combination of S4.i.3.2-S4.i.3.4
          od;
        part Black Box Models   [document unspecified objects as black box fox
                                 functions]
          S4.i.3.5:    Determine Black Box Behavior Of Other Solution Domain
                       Objects;
        part Prototyping     [design, develop, conduct and appraise prototype
                              experiments to appraise solution possibilities]
          do
            S4.i.3.6:    Develop prototype software using Cleanroom practices;
            S4.i.3.7:    Conduct and appraise prototype experiment
            S4.i.3.8:    Store prototype components in project reuse repository;
          until
            Completion Conditions achieved for combination of S4.i.3.7 and S4.i.3.8
          od;
        esac;
      od;
    noc;
  until
    sufficient Completion Conditions achieved for all selected analysis methods and
    (time period up or sufficient understanding has been obtained)
  od;
corp;
```

Figure 4.1(k) Generic Cleanroom Process Model **proc** P4.1.3: Solution Domain Analysis-Cycle i

```
proc P4.i.4:  Prepare Cycle i Specifications
  do  [P4.i.4:  Prepare Cycle i Specifications]
    con
      S4.i.4.1:   Record Cycle i Results In Mission Volume;
      S4.i.4.2:   Record Cycle i Results In User's Reference Manual Volume;
      S4.i.4.3:   Record Cycle i Results In Black Box Function Volume;
      S4.i.4.4:   Record Cycle i Results In Mission Validation Volume;
      S4.i.4.5:   Record Cycle i Results In Usage Profile Volume;
      S4.i.4.6:   Record Cycle i Results In Construction Plan Volume;
    noc;
  until
    Completion Conditions achieved for S4.i.4.1 through S4.i.4.6
  od;
corp;
```

Figure 4.1(l) Generic Cleanroom Process Model **proc** P4.1.4: Prepare Cycle i Specifications

```
proc P4.i.5:  Appraise Cycle i Specifications
  do  [P4.i.5:  Appraise Cycle i Specifications]
    M4.i.5.1:  Review and Evaluate Cycle i specifications;
    M4.i.5.2:  Management Decision: (1) Specifications Suitable To Initiate
               Development or (2) Specification Problems-Replan Project or (3)
               Continue Specification Effort with cycle i+1;
  until
    Completion Conditions achieved for M4.i.5.2
  od;
corp;
```

Figure 4.1(m) Generic Cleanroom Process Model **proc** P4.1.5: Appraise Cycle i Specifications

```
proc P5: Software Development
   [This process results in the development of the software for the spiral (project).]
   con [Several increments will be in various stages of development]
      do [P5: Software Development]
         [P5.j results in the development of the software (design and code) and
         certification plan (sampling plan, test scenarios and expected results) for
         increment j.]
         run  P5.j: Software and Certification Development for Increment j; [where j is
                 the next increment in the sequence j=1...j...n according to the project plan]
         if D5.j.3.5 team decision indicates the need for specification revision
         [D5.j.3.5 results in one of the following decisions: increment complete -ready to
         be certified, design problems - replan, specification problems - revisions needed]
         then
         [PCS2 is a common services procedure which results in an updated specification]
            run PCS2: Update Specifications;
         fi;
         if MCS2.4 management decision is revised specifications require replanning or
            D5.j.3.5 team decision indicates the need for replanning
            [MCS2.4 results in a decision that: revised specs can be used without
            replanning or revised specs result in need to replan and D5.j.3.5 results in one
            of the following decisions: increment complete - ready to be certified, design
            problems - replan, specification problems - revisions needed]
         then
            [PCS1: is a common service procedure which results in an updated Software
            Development Plan or a decision to terminate current project/spiral.]
            run  PCS1: Replan project/spiral;
            if MCS1.4 management decision indicates need to terminate project spiral
            then
               terminate current cycle
            fi;
         fi;
      od;
   noc;
corp;
```

Figure 4.1(n) Generic Cleanroom Process Model **proc** P5: Software Development

```
proc P5.j: Software Development and Certification Preparation
   [For each P5.j, the specification is tailored, then the software is designed to the code by
   the Development Team (P5.j.3), while the Certification Team does the work necessary
   to prepare for certification of the increment (P5.j.2).]
   do    [P5.j: Software and Certification Planning]
      run P5.j.1:    Tailor specification to increment/accumulation j;
      con
         run P5.j.2:    Prepare for Certification of Accumulation j;
         run P5.j.3:    Increment j Development;
      noc;
   until
      Completion Conditions achieved for P5.j.2 and P5.j.3 or P5.j.3 team decision
      indicates the need for replanning or specification revisions
   od;
corp;
```

Figure 4.1(o) Generic Cleanroom Process Model **proc** P5.j: Software Development and Certification Preparation

```
proc P5.j.1:   Tailor specification to increment/accumulation j
   do [P5.j.1: Tailor specification to increment/accumulation j]
      con
         S5.j.1.1:    Tailor Black Box function to increment/accumulation j;
         S5.j.1.2:    Tailor Usage Profile to increment/accumulation j;
      noc;
   until
      Completion Conditions achieved for S5.j.1.1 and S5.j.1.2
   od;
corp;
```

Figure 4.1(p) Generic Cleanroom Process Model **proc** P5.j.1: Tailor specifications to increment/accumulation j

```
proc P5.j.2: Prepare for Certification of Accumulation j
   [This process results in test plan and test scenarios for accumulation j]
   [P5.j.2:  Prepare for Certification of Accumulation j]
   con
      do
         do
            C5.j.2.1:    Prepare test plan;
         until
            Completion Conditions for C5.j.2.1 achieved
         od;
         do
            con
               C5.j.2.2:    Prepare test scenarios or test case generator for accumulation j;
               C5.j.2.3:    Determine expected results for test cases;
            noc;
         until
            Completion Conditions for C5.j.2.2 and C5.j.2.3 achieved
         od;
      od;
      C5.j.2.4:    Increase Understanding of Problem and Solution Domains;
   noc;
corp;
```

Figure 4.1(q) Generic Cleanroom Process Model **proc** P5.j.2: Prepare for Certification of Accumulation j

```
proc P5.j.3: Increment j Development
   [Process results in verified software for increment.]
   do [P5.j.3: Increment Development]
      con
         for   all design objects currently available to be worked on
         do
            D5.j.3.1:   Select a box design object from pick list; [Items appear on
                        individual developers pick list based on (1) Completion Conditions
                        achieved for parent design object, (2) design object previously
                        assigned to team member making selection and (3) design object is
                        not yet fully refined]
            D5.j.3.2:   run P5.j.3.2 Refine and verify box;
            D5.j.3.3:   Update pick list; [To reflect results of D5.j.3.2]
         od;
         D5.j.3.4:   Increase Understanding of Problem and Solution Domains;
         D5.j.3.5:   Team Decision if one is required: (1) Increment complete, verified
                     and ready for certification, (2) Design problems-project/spiral should
                     be replanned or (3) Specification problems-specifications should be
                     revised;
      noc;
   until
      D5.j.3.5 team decision indicates increment complete or project should be replanned
      or specifications should be revised
   od;
corp
```

Figure 4.1(r) Generic Cleanroom Process Model **proc** P5.j.3: Increment Development

```
proc P5.j.3.2: Refine and Verify Box
   [Process results in a refined and verified box.]
   do [P5.j.3.2: Refine and Verify Box
      D5.j.3.2.1: Refine, verify and team review design object;
      if team review passes
      then
         D5.j.3.2.2: Sign completion Conditions;
      else
         D5.j.3.2.3: Team Decision: (1) Continue refinement of design object, (2) Change
                     some prior design decision which requires pruning of design
                     hierarchy;
      fi;
   until
      Completion Conditions signed by full team or team review concludes that a prior
      design decision be modified which requires pruning of design hierarchy and
      updating of pick list
   od;
corp;
```

Figure 4.1(s) Generic Cleanroom Process Model **proc** P5.j.3: Increment Development

```
proc P6: Software Certification
   [P6 certifies the software and makes the decision as to whether the software should be
   accepted or rejected.]
   con [Several increments may be in various stages of certification]
      do   [P6: Software Certification]
         if Completion Conditions achieved for P5.j and P6.j-1
         then
            [P6.j addresses the actual certification of the developed software for an
            increment, using the software and certification design from P5.j.]
            run  P6.j: Software Certification for Increment j; [where j is the next
                 increment in the sequence j=1...j...n according to the project plan]
            if M6.j.5 management decision is certification quality not satisfactory-
               replanning required
            [M6.j.5 results in one of the following decisions: certification complete-
            quality satisfactory, certification complete-quality not satisfactory-Replanning
            is required, or certification should continue]
            then
               run  PCS1: Replan project/spiral;
               if MCS1.4 management decision indicates need to terminate project cycle
               then
                  terminate current cycle
               fi;
            fi;
         fi;
      od;
   noc;
corp;
```

Figure 4.1(u) Generic Cleanroom Process Model **proc** P6: Software Certification

```
proc P6.j: Software Certification for Increment j
   [P6.j certifies the code, and makes the decision as to whether an increment will be
   accepted or rejected.]
   do   [P6.j: Software Certification for Increment j]
      C6.j.1:  Build Accumulation j;
      if no pre-certification failures
      then
         while
            certification plan requires more tests and sufficient failures have not been
            observed to make it desirable to terminate testing and wait for corrections
         do
            C6.j.2:  Perform Certification Tests for Accumulation j;
         od;
      fi;
      if at least one failure observed and observed failures are considered to be
         correctable
      then
         C6.j.3:  Prepare failure report(s);
         D6.j.4:  Correct failure, verify correction and prepare ECN;
      fi;
      M6.j.5:  Management Decision: (1) certification complete-accumulation quality
               satisfactory, (2) certification complete-quality not satisfactory-replanning Is
               required or (3) certification should continue;
      if not Management Decision is certification should continue
      then
         C6.j.6:  Prepare Certification Report;
      fi;
   until
      Completion Conditions achieved for C6.j.6
   od;
corp;
```

Figure 4.1(v) Generic Cleanroom Process Model **proc** P6.j: Software Certification for Increment i

```
proc P7: Specification Configuration Management
   [This process results in the specification being kept current with all changes required
   due to normal project findings.]
   do
      con
         S7.1 Consult with Development and Certification Teams About Specification
              Issues;
         S7.2 Update Specification as required;
         S7.3 Increase Understanding of Problem and Solution Domains;
      noc;
   until
      project spiral terminated and Completion Conditions for combination of S7.1 and
      S7.2 achieved
   od;
corp;
```

Figure 4.1(w) Generic Cleanroom Process Model **proc** P7: Specification Configuration Management

```
proc P8: Product (Project) Releases
   [P8 results in the preparation of all of the project deliverables to all of the project
   stakeholders]
   do [P8: Product (Project) Releases]
      con
         S8.1:  Prepare all required user documentation;
         S8.2:  Specification Team prepare final project releases according to the plan;
         D8.3:  Development Team prepare final project releases according to the plan;
         C8.4:  Certification Team prepare final project releases according to the plan;
      noc;
   until
      Completion Conditions achieved for S8.1, S8.2, D8.3 and C8.4
   od;
corp;
```

Figure 4.1(x) Generic Cleanroom Process Model **proc** P8: Product (Project) Releases

```
proc PCS1: Replan project/spiral
   [This process results in an updated Software Development Plan  or a decision to
   terminate current project/spiral.]
   do
      MCS1.1    Appraise Situation and Plan Investigation;
      MCS1.2    Conduct Investigation;
      MCS1.3    Analyze results;
      if analysis results in decision to continue
         then
            MCS1.4    Revise Software Development Plan in Accordance with Decision
      fi;
   until
      Completion Conditions achieved for MCS1.3 and MCS1.4 if the revise plan was
      taken
   od;
corp;
```

Figure 4.1(y) Generic Cleanroom Process Model **proc** PCS1: Replan project/spiral

```
proc PCS2: Update Specifications
   [This process results in a major update to the specification]
   do [PCS2: Update Specification]
      if Question or Issue or stimulus from outside project or error discovery causes a
         specification change
      then
         do
            con
               SCS2.1:    Increase Understanding of Problem and Solution Domains;
               SCS2.2:    Update Specification;
            noc;
            SCS2.3:    Publish Change Sheets;
            MCS2.4:    Management Decision: (1) OK continue current plan with revised
                       specifications or (2) Revised specifications require replanning
         until
            Completion Conditions for MCS2.4 achieved
         od;
      fi;
   od;
corp;
```

Figure 4.1(z) Generic Cleanroom Process Model **proc** PCS2: Update Specifications

P0: Cleanroom Process for Spiral s
- P1: Spiral (Project) Invocation
- P2: Information Management
- P3: Program Management
  - P3.1.k: Initiate Project Review k, k=1,...
  - P3.2.t: Initiate Prepare and Submit Status Report t, t=1,...
- P4: Specification Preparation
  - .
  - .
  - P4.i: Software Specification for Cycle i
    - P4.i.1: Prepare Plan for Cycle i
    - P4.i.2: Problem Domain Analysis-Cycle i
    - P4.i.3: Solution Domain Analysis-Cycle i
    - P4.i.4: Prepare Cycle i Specifications
    - P4.i.5: Appraise Cycle i Specifications
  - PCS1: Replan project/spiral
  - .
  - .
  - .
- P5: Software Development
  - .
  - .
  - .
  - P5.j: Software Development and Certification Preparation
    - P5.j.1: Tailor specification to increment/accumulation j
    - P5.j.2: Prepare for Certification of Accumulation j
    - P5.j.3: Increment j Development
  - PCS1: Replan project/spiral
  - PCS2: Update Specifications
  - .
  - .
  - .
- P6: Software Certification
  - .
  - .
  - .
  - P6.j: Software Certification for Increment j
  - PCS1: Replan project/spiral
  - .
  - .
- P7: Specification Configuration Management
- P8: Product (Project) Releases

Figure 4.2 Work Breakdown Structure for Project Planning

P0: Cleanroom Process for Spiral s

P1: Spiral (Project) Invocation

M1.1: Allocate and organize initial project staff
M1.2: Allocate and organize other project resources
M1.3: Agree on project charter
M1.4: Tailor software development plan for project/spiral to Project/Spiral Plan
M1.5: Initialize metrics collection activities
M1.6: Resolve all questions and issues
M1.7: Archive external information received

P2: Information Management

S2.1: Resolve questions and issues
D2.2: Resolve questions and issues
C2.3: Resolve questions and issues
M2.4: Resolve questions and issues
S2.5: Archive external information
D2.6: Archive external information
C2.7: Archive external information
M2.8: Archive external information

P3: Program Management

P3.1.k: Initiate Project Review k, k=1,...

M3.1.k.1: Prepare Review
M3.1.k.2: Conduct Review
M3.1.k.3: Issue Directives to Implement Review Findings

P3.2.t: Initiate Prepare and Submit Report t, t=1,...

M3.2.t.1: Prepare and Submit Report t
M3.2.t.2: Monitor Metrics Collection

M3.3: Organize project staff as required

Figure 4.3 Work Breakdown Structure for Dispatching (1 of 4)

P4: Specification Preparation

P4.i: Software Specification for Cycle i

P4.i.1: Prepare Plan for Cycle i

M4.i.1.1: Establish Objectives For Cycle i
M4.i.1.2: Allocate Time Period To Cycle i
M4.i.1.3: Prepare Plan For Cycle i

P4.i.2: Problem Domain Analysis-Cycle i

S4.i.2.1: Information Collection
S4.i.2.2: Assemble Information for System(s)
S4.i.2.3: Manual System Reverse Engineering
S4.i.2.4: Automated System Reverse Engineering
S4.i.2.5: Hybrid System Reverse Engineering
S4.i.2.6: Develop and Analyze Black Box Model
S4.i.2.7: Develop and Analyze Markov Usage Model

P4.i.3: Solution Domain Analysis-Cycle i

S4.i.3.1: Information Collection
S4.i.3.2: Browse Reuse Repositories To Find Match For Needs
S4.i.3.3: Prepare Cost/Benefit Analysis
S4.i.3.4: Select Potential Reuse Modules for Integration into System
S4.i.3.5: Determine Black Box Behavior Of Other Solution Domain Objects
S4.i.3.6: Develop prototype software using Cleanroom practices
S4.i.3.7: Conduct and appraise prototype experiment
S4.i.3.8: Store prototype components in project reuse repository

P4.i.4: Prepare Cycle i Specifications

S4.i.4.1: Record Cycle i Results In Mission Volume;
S4.i.4.2: Record Cycle i Results In User's Reference Manual Volume;
S4.i.4.3: Record Cycle i Results In Black Box Function Volume;
S4.i.4.4: Record Cycle i Results In Mission Validation Volume;
S4.i.4.5: Record Cycle i Results In Usage Profile Volume;
S4.i.4.6: Record Cycle i Results In Construction Plan Volume;

P4.i.5: Appraise Cycle i Specifications

M4.i.5.1: Review and Evaluate Cycle i specifications;
M4.i.5.2: Management Decision: (1) Specifications Suitable To Initiate Development or (2) Specification Problems-Replan Project or (3) Continue Specification Effort with cycle i+1;

PCS1: Replan project/spiral

MCS1.1 Appraise Situation and Plan Investigation
MCS1.2 Conduct Investigation
MCS1.3 Analyze results
MCS1.4 Revise Software Development Plan in Accordance with Decision

Figure 4.3 Work Breakdown Structure for Dispatching (2 of 4)

P5: Software Development

P5.j: Software Development and Certification Preparation

P5.j.1: Tailor specification to increment/accumulation j

S5.j.1.1: Tailor Black Box function to increment/accumulation j

S5.j.1.2: Tailor Usage Profile to increment/accumulation j

P5.j.2: Prepare for Certification of Accumulation j

C5.j.2.1: Prepare test plan

C5.j.2.2: Prepare test scenarios or test case generator for accumulation j

C5.j.2.3: Determine expected results for test cases

C5.j.2.4: Increase Understanding of Problem and Solution Domains

P5.j.3: Increment j Development

D5.j.3.1: Select a box design object from pick list

D5.j.3.2: Refine and Verify Box

D5.j.3.2.1: Refine, verify and team review design object

D5.j.3.2.2: Sign Completion Conditions

D5.j.3.2.3: Team Decision: (1) Continue refinement of design object, (2) Change some prior design decision which requires pruning of design hierarchy

D5.j.3.3: Update pick list

D5.j.3.4: Increase Understanding of Problem and Solution Domains

D5.j.3.5: Team Decision if one is required: (1) Increment complete, verified and ready for certification, (2) Design problems-project/spiral should be replanned or (3) Specification problems-specifications should be revised

PCS1: Replan project/spiral

MCS1.1 Appraise Situation and Plan Investigation

MCS1.2 Conduct Investigation

MCS1.3 Analyze results

MCS1.4 Revise Software Development Plan in Accordance with Decision

PCS2: Update Specifications

SCS2.1: Increase Understanding of Problem and Solution Domains

SCS2.2: Update Specification

SCS2.3: Publish Change Sheets

MCS2.4: Management Decision: (1) OK continue current plan with revised specifications or (2) Revised specifications require replanning

Figure 4.3 Work Breakdown Structure for Dispatching (3 of 4)

P6: Software Certification

P6.j: Software Certification for Increment j

C6.j.1: Build Accumulation j

C6.j.2: Perform Certification Tests for Accumulation j

C6.j.3: Prepare failure report(s)

D6.j.4: Correct failure, verify correction and prepare ECN

M6.j.5: Management Decision: (1) certification complete-accumulation quality satisfactory, (2) certification complete-quality not satisfactory-replanning is required or (3) certification should continue

C6.j.6: Prepare Certification Report

P[illegible]1: Replan project/spiral

MCS1.1 Appraise Situation and Plan Investigation

MCS1.2 Conduct Investigation

MCS1.3 Analyze results

MCS1.4 Revise Software Development Plan in Accordance with Decision

P7: Specification Configuration Management

S7.1 Consult with Development and Certification Teams About Specification Issues

S7.2 Update Specification as required

S7.3 Increase Understanding of Problem and Solution Domains

P8: Product (Project) Releases

S8.1: Prepare all required user documentation

S8.2: Specification Team prepare final project releases according to the plan

D8.3: Development Team prepare final project releases according to the plan

C8.4: Certification Team prepare final project releases according to the plan

Figure 4.3 Work Breakdown Structure for Dispatching (4 of 4)

SECTION 5: CLEANROOM READING LIST

For additional information on Cleanroom and aspects of the Cleanroom process, the following references are suggested:

Cobb, R.H., H.D. Mills and A. Kouchakdjian, The Cleanroom Engineering Software Development Process Manual, SET, 1991.

Cobb, R.H. and H.D. Mills, "Engineering Software Under Statistical Quality Control," IEEE Software, November 1990.

Green, S.E., A. Kouchakdjian and V.R. Basili, The Cleanroom Case Study in the Software Engineering Laboratory: An Experiment in Formal Methods, Software Engineering Laboratory, NASA Goddard Space Flight Center, 1989.

Linger, R.C. and H.D. Mills and B.I. Witt, Structured Programming: Theory and Practice, Addison-Wesley Publishing Company, Inc., Reading, MA, 1979.

Linger, R.C. and H.D. Mills, "A Case Study in Cleanroom Software Engineering: The IBM COBOL Structuring Facility," Proceeding of COMPSAC '88, IEEE 1988.

Mills, H.D., R.C. Linger and A. Hevner, Principles of Information Systems Analysis and Design, Academic Press, Inc. New York, 1986.

Mills, H.D., M. Dyer and R.C. Linger, "Cleanroom Software Engineering," IEEE Software, September 1987.

Mills, H.D., "Stepwise Refinement and Verification in Box-Structured Systems," IEEE Computer, June 1988.

Mills, H.D. and J.H. Poore, "Bringing Software Under Statistical Quality Control," Quality Progress, November 1988.

Whittaker, J., **Markov Chain Techniques for Software Testing and Reliability Analysis**, Ph.D. Dissertation, Department of Computer Science, University of Tennessee, Knoxville, 1992.