Solid Waste Projection Model: Database (Version 1.4)


C. Blackburn
T. Cillan

September 1993

Prepared for
the U.S. Department of Energy
under Contract DE-AC06-76RLO 1830

Pacific Northwest Laboratory
Richland, Washington 99352
Executive Summary

The Solid Waste Projection Model (SWPM) system is an analytical tool developed by Pacific Northwest Laboratory (PNL) for Westinghouse Hanford Company (WHC). The SWPM system provides a modeling and analysis environment that supports decisions in the process of evaluating various solid waste management alternatives.

This document, one of a series describing the SWPM system, contains detailed information regarding the software and data structures utilized in developing the SWPM Version 1.4 Database. This document is intended for use by experienced database specialists and supports database maintenance, utility development, and database enhancement.

Those interested in using the SWPM database should refer to the SWPM Database User’s Guide. This document is available from the PNL Task M Project Manager (D. L. Stiles, 509-372-4358), the PNL Task L Project Manager (L. L. Armacost, 509-372-4304), the WHC Restoration Projects Section Manager (509-372-1443), or the WHC Waste Characterization Manager (509-372-1193).
Contents

Executive Summary ................................................................. iii

1.0 Introduction ........................................................................... 1.1
   1.1 Solid Waste Projection Model Database System Goal ................. 1.2
   1.2 Documents ......................................................................... 1.4

2.0 Database Hardware and Software Requirements ......................... 2.1
   2.1 Hardware Requirements .................................................... 2.1
      2.1.1 Computer type ....................................................... 2.1
      2.1.2 Memory ............................................................... 2.1
      2.1.3 Disk Space ......................................................... 2.1
      2.1.4 Network Access ................................................. 2.1
   2.2 Software Requirements .................................................... 2.1
   2.3 Paradox Default Settings .................................................. 2.2
   2.4 Required Network Configuration ........................................ 2.2

3.0 Database Files and Table Structures ....................................... 3.1
   3.1 File Types ........................................................................ 3.1
   3.2 File Naming and Location Conventions .................................. 3.2
      3.2.1 Report Files, Tables, Procedures ................................ 3.2
      3.2.2 Data Tables ......................................................... 3.3
      3.2.3 Queries .............................................................. 3.3
      3.2.4 PAL Scripts ......................................................... 3.4
      3.2.5 Procedure Libraries and Related Files .......................... 3.4
      3.2.6 Ad Hoc Files ....................................................... 3.5
   3.3 Directory Structure .......................................................... 3.5
   3.4 Database Structure .......................................................... 3.9
   3.5 Data Dictionary ............................................................... 3.11

4.0 Application Startup Scripts ................................................... 4.1
   4.1 Version Selection Script - SWPM.SC .................................. 4.1
   4.2 Application Execution Script - STARTSWP.SC ....................... 4.1
5.0 Database Processes and Menus .......................................................... 5.1
  5.1 Process Dictionary .............................................................................. 5.1
  5.2 Menu System ....................................................................................... 5.9
     5.2.1 PAL Scripts and Procedures .......................................................... 5.9
     5.2.2 Tables and Related Objects .......................................................... 5.9
     5.2.3 Adding an Item to a Menu ............................................................. 5.9

6.0 Forecast Data Entry ............................................................ 6.1
  6.1 Procedures and Scripts ............................................................. 6.1
  6.2 Table Names .............................................................................. 6.4
  6.3 Implementation ............................................................................. 6.4

7.0 Data Entry Notepad Functions .................................................. 7.1
  7.1 Procedures and Scripts ............................................................. 7.1
  7.2 Notepad Tables and Related Objects ............................................... 7.2
  7.3 Implementation ............................................................................. 7.3
  7.4 Automatically Captured Contextual Information .................................. 7.4

8.0 Reporting Functions ............................................................... 8.1
  8.1 Descriptions and Locations of Files ............................................... 8.1
     8.1.1 PAL Scripts and Procedures ....................................................... 8.1
     8.1.2 Tables and Related Objects ........................................................ 8.2
  8.2 Adding a Report to the Report Manager .......................................... 8.2
     8.2.1 Defining a new Report Manager Record ........................................ 8.2
     8.2.2 Other Preparation ...................................................................... 8.3
     8.2.3 Documenting a New Report ........................................................ 8.4
  8.3 Execution of the Report Manager .................................................. 8.4

9.0 Upload and Download Processing ............................................. 9.1
  9.1 Descriptions and Locations of Files ............................................... 9.1
     9.1.1 DOS Batch Files ..................................................................... 9.1
     9.1.2 Log Files .............................................................................. 9.2
     9.1.3 Quality Assurance Query Scripts ................................................. 9.2
9.2 Process Descriptions ........................................ 9.2
  9.2.1 Upload Processing ..................................... 9.2
  9.2.2 Download Processing ................................... 9.3
  9.2.3 Upload and Download Logging ............................. 9.3

9.3 Quality Control of Uploaded Data ................................ 9.4
  9.3.1 Verification of Waste Generator Abbreviation Usage .......... 9.4
  9.3.2 Verification of Waste Class Abbreviation Usage .................. 9.4
  9.3.3 Detection of New Waste Generator Names ..................... 9.4
  9.3.4 Detection of Existing Waste Generator Names Now Absent From Upload . 9.5

9.4 Installation of Uploaded Data .................................. 9.5
  9.4.1 Installation When All Data Entry is Local ...................... 9.5
  9.4.2 Installation When Data Entry Conducted Concurrently .......... 9.5

9.5 Preparation for the Next Upload ................................ 9.6

10.0 Reference Data Library Generator ................................... 10.1
  10.1 Scenario Management ....................................... 10.1
  10.2 Forecast Data Preparation .................................... 10.1
  10.3 RDL File Generation Process ................................ 10.1
  10.4 RDL File Generator Operation ................................. 10.1

11.0 Procedure Libraries ............................................ 11.1
  11.1 ADMIN Library .......................................... 11.1
  11.2 DETOOLS Library ......................................... 11.1
  11.3 FEAEDIT Library ......................................... 11.1
  11.4 FOREDE Library .......................................... 11.3
  11.5 RDL Generator Library ...................................... 11.12
  11.6 UTIL Library ............................................ 11.13

12.0 Development and Administration Tools ............................... 12.1
  12.1 Status Accounting ......................................... 12.1
    12.1.1 Procedures and Scripts ................................ 12.1
    12.1.2 Tables and Related Objects ............................... 12.1
    12.1.3 Implementation ........................................ 12.2
    12.1.4 Operation .......................................... 12.2
    12.1.5 Maintenance ........................................ 12.2
12.2 Table/Field Documenter ..................................... 12.2
12.2.1 Procedures and Scripts ..................................... 12.3
12.2.2 Tables and Related Objects .................................. 12.3
12.2.3 Implementation ........................................... 12.3
12.2.4 Operation ............................................... 12.3
12.2.5 Maintenance .............................................. 12.4

12.3 Report Development Tools .................................... 12.4
12.3.1 Menu Options ............................................. 12.4
12.3.2 Procedures and Scripts ..................................... 12.5
12.3.3 Implementation ........................................... 12.6

12.4 Password Protection ........................................ 12.6
12.4.1 Levels of Security ......................................... 12.6
12.4.2 Procedures and Scripts ..................................... 12.7
12.4.3 Implementation ........................................... 12.7

12.5 Miscellaneous Tools .......................................... 12.7

13.0 Maintaining Treatment, Storage, and Disposal Data ....................... 13.1
13.1 The Oper Table ............................................... 13.2
13.2 The Dist_F Table ............................................. 13.2
13.3 The Operflow Table ........................................... 13.2
13.4 The Opercap Table ............................................ 13.3
13.5 The WG_TSD Table ............................................ 13.3
13.6 The WSTSTRM Table .......................................... 13.3
Figures

1.1 The Solid Waste Management System Representation ..................................... 1.2
1.2 The SWPM System ......................................................................................... 1.3
1.3 Primary Processes Performed by the SWPM Database ....................................... 1.5
3.1 SWPM Database Directory Structure ................................................................ 3.6
3.2 SWPM Forecast Data Entity Relationship Diagram ......................................... 3.10
4.1 Listing of SWPM.SC .................................................................................... 4.3
4.2 Listing of STARTSWP.SC ........................................................................... 4.5
5.1 SWPM Menu Structure ................................................................................... 5.2
9.1 Listing of SWPMUPLD.BAT ......................................................................... 9.7
9.2 Listing of SWPMDNLD.BAT .......................................................................... 9.10
9.3 Listing of SWPMLOG.BAT .......................................................................... 9.15
12.1 Report Development Tools Menu ................................................................. 12.9
12.2 Page Setup Menu ......................................................................................... 12.9
12.3 Report Parameters Screen ............................................................................ 12.9

Tables

3.1 SWPMDB Data Tables and Fields .................................................................. 3.12
1.0 Introduction

The Solid Waste Projection Model (SWPM) system is an analysis tool developed for Westinghouse Hanford Company’s (WHC’s) Restoration Projects Section by Pacific Northwest Laboratory (PNL)* to assist analysis of complex waste management issues. The SWPM system provides the ability to develop projections of solid waste volumes and characteristics and evaluate alternative waste treatment and disposal strategies.

A generic representation of the system modeled by the SWPM is shown in Figure 1.1. Waste is received from waste generating facilities and is distributed to various operations. Operations are defined as either "treatments" or "disposals" and are linked to other operations to represent a given waste management scheme. Each operation has an associated storage option to track waste volumes that arrive in excess of defined capacity.

The SWPM system consists of three modules: the database, the user interface, and the model. Figure 1.2 shows how the components of the system interact and the organization responsible for each component.

The SWPM database, which is described in this technical reference manual, performs three important functions for the SWPM system. First, the database provides secure storage for waste volume forecasts and for system characterizations. Second, it produces the Reference Data Library for input to the model. Last, it uses the information to generate data reports for performing studies without the model. Information such as physical attributes of wastes or waste packaging can be summarized using standard data reports. The database reports can also be used to verify or review the information stored in the system.

The SWPM database uses Paradox database software. Data are entered and maintained through menus and custom data entry screens. There are two distinct types of data in the database. As shown in Figure 1.1, volumes and characteristics that describe the waste from the waste generators, as well as the cost, capacity, and flow instructions for the treatment, storage, and disposal (TSD) operations, are maintained. As illustrated in Figure 1.2, waste generator information is maintained by WHC, while TSD data are maintained by PNL.

The database, with its menu structure and data entry screens, provides the ability to assemble and maintain many different waste volume forecasts and any number of TSD alternatives, and to make large-scale changes to this information.

---

(a) The Pacific Northwest Laboratory is operated by the Battelle Memorial Institute for the U.S. Department of Energy under Contract DE-AC06-76RLO 1830.
Figure 1.1. The Solid Waste Management System Representation

1.1 Solid Waste Projection Model Database System Goal

The goal of the SWPM database is to provide the environment for collecting and maintaining the data required for input to the SWPM system. In addition, the SWPM database provides forecast data summary reports. The primary processes performed by the SWPM database include the following:

- maintain waste generator data
- maintain TSD data
- maintain model-variable dictionary data
Figure 1.2. The SWPM System
• maintain feature codes and descriptions

• produce forecast data summary reports

• produce the reference data library files.

Figure 1.3 shows the primary processes performed by the SWPM database in a data flow diagram. This figure represents the flow of information between the processes, and the external sources and destinations for data.

1.2 Documents

This Database Technical Reference Manual describes Version 1.4 of the SWPM database, discusses the hardware and software requirements necessary to operate the database, and defines the directory and the data structure in which the information is stored. Five additional documents, which provide instructions in the use, maintenance, and application of the SWPM system, complete the documentation set. These documents are the System Overview, Database User’s Guide, the Model User’s Guide, the Model Technical Reference Manual, and the Configuration Management and Administration Plan. The following is a brief description of each of the six manuals:

• System Overview - provides an overview of the SWPM system and an assessment of potential applications

• Database User’s Guide - provides instructions for data entry, maintenance, and reporting

• Model User’s Guide - provides instructions for model operation and execution

• Model Technical Reference Manual - describes model software and applications

• SWPM Database (version 1.4) Technical Reference Manual - describes database software, utilities, and structure

• Configuration Management and Administration Plan - provides instructions for long-term system administration and maintenance.

This Database Technical Reference Manual is provided as a reference guide for a knowledgeable database specialist. Chapter 2 of this manual addresses hardware and software requirements and the structure of the SWPM database. The remaining chapters detail the essential operations and functions performed by the database and the specific routines, tables, and scripts that must be accessed and maintained to provide continuing database support.
Figure 1.3. Primary Processes Performed by the SWPM Database
It is important to note that this manual does not contain any user help information, nor does it attempt to assist the analyst in reporting data or in utilizing SWPM data for analyses. The *Database User's Guide* and the *System Overview* will provide the information required for these purposes.
2.0 Database Hardware and Software Requirements

This chapter provides the database specialist with guidance on hardware and software needed to set up and operate the SWPM database application.

2.1 Hardware Requirements

2.1.1 Computer type

IBM or compatible with a 20 megahertz 386 processor or better. Lesser processors will work although performance will probably be unacceptable.

2.1.2 Memory

Two to four megabytes (mb) of RAM is recommended. If Paradox is executed as a Windows 3.1 task, eight mb is recommended.

2.1.3 Disk Space

When using the SWPM database, some report processes will require 30 mb of temporary work space on disk, the location of which is defined by the Paradox setup. Usually this space must exist on a local (non-network) hard drive. Approximately 20 mb of permanent storage space will also be used if a local copy of the database is made by downloading the network database.

2.1.4 Network Access

Access to the network is required, although a user may create a local copy of the database and use a stand-alone version of Paradox. Section 2.5 contains information about network connections needed for access to SWPM data.

2.2 Software Requirements

The SWPM database application runs under DOS Version 3.3 or higher. The application requires Version 3.5 of the Paradox relational database management system. Paradox controls the storage and retrieval of information on the computer and provides facilities used by the database developer to create menus, forms, and reports. Certain changes to default parameter settings for the Paradox installation are required. See Section 2.4.

Access to the file server containing the network Paradox software installation is required in order to use the network-resident SWPM database. This requirement applies to owners of stand-alone Paradox copies as well, since multi-user access of network data must be controlled through a file kept
with the network software. If a user does not own an individual copy of Paradox, he or she must have approval to use the Paradox software installed under network license. See Section 2.4 for information about the server connections required to access SWPM data, application software, and network Paradox.

2.3 Paradox Default Settings

Default settings are modified by playing the CUSTOM script and choosing items to set from the menus presented. This script can be found in the directory containing the network Paradox software or in the directory containing a stand-alone Paradox installation.

The directions below show the progression of menu selections needed for each parameter and assume that the user is starting from the top level menu. Menu selections are shown in bold. Other text must be typed in.

Set default length of page to 60 lines -

Reports LengthOfPage 60

Send form feeds at end of every report page -

Reports FormFeed FormFeeds

Treat blanks as zeros for calculating purposes -

Defaults Blank=Zero Yes

Set the private directory to a local hard disk with at least 30 mb of available space. If certain Paradox files have been copied to a local hard drive to enhance performance or a stand-alone copy of Paradox is in use, this directory can be the same as the one containing the Paradox software -

Net SetPrivate "private directory name"

When complete, select Do_It! or press F2. Choose HardDisk if you are working from a stand-alone Paradox installation or NetWork if you are running network software. In the latter case, override the suggested directory and save the configuration to the private directory designated in the instructions above.

2.4 Required Network Configuration

All SWPM users who wish to access the data and application software on the network need to connect to the network server as follows:
USE J: \"server name + share name"

The server and share name combination is available from the database administrator.

The use of "J:" for the drive designation is recommended. The application will run with other drive letters, but problems with shared use of the database may result. Also, the drive letter indicates to the application whether the network reference database is in use. If a local copy is used (or a network drive designated as something besides "J:"), standard reports will be labeled as "Unofficial." This is done on the assumption that the reports may be generated from a local, uncontrolled copy of the database. See Chapter 12 for technical notes on the SWPM start-up script, where the drive designation is examined.

Users must also connect to the server on which the network Paradox software is installed. This includes users who have their own copy of Paradox or who have made a local copy of the network Paradox software for performance improvement. Enter the following:

USE K: \"server name + share name"

The server and share names are available from the database administrator.

The use of a drive letter other than "K" may hinder the ability of Paradox to manage multiuser access to the SWPM database.

The SWPM application software assumes that the user's current working directory is J:\SWPMDB92. This can be set from the Paradox menu by selecting Tools More Directory and then entering the directory name. In addition, the default working directory can be changed to this name by playing the CUSTOM script and designating J:\SWPMDB92 as the initial working directory from the Defaults menu. A third method to establish the working directory is to include the directory change command in INIT.SC, a script that Paradox runs immediately after it is first executed. This script should be located in the private directory for network users and, for stand-alone users, in the directory that contains the Paradox software.
3.0 Database Files and Table Structures

This chapter provides the database specialist with a description of file types used in the SWPM database system and details the structure of the database.

3.1 File Types

Several types of files are used throughout the SWPM database. File types are identified by the file extension, a one- to three-character suffix appended to the file name. Listed below are descriptions of the types of files in use:

- **.BAT** DOS batch file - Batch files are used to execute a series of DOS commands.
- **.DB** Paradox database table - These are the files in which data are stored.
- **.Fnn** Paradox form - Forms are used to view and maintain data. They have the same file name as the related database table. Forms have a file name extension of either ".F" (for the default form) or ".Fnn", where 'nn' is a form number from 1 to 14. As a result of this convention, there can be no more than 15 forms for each table.
- **.LIB** Paradox library - Libraries contain compiled versions of scripts. Each library may have one or more compiled scripts.
- **.PX** Paradox primary index - If a table has key fields, a primary index file with the same name as the table is created and maintained automatically.
- **.Rnn** Paradox report specification - Report specifications are used to display data to a printer, screen, or file. Each has the same file name as the related database table. Reports have a file name extension of either ".R" (for the default report) or ".Rnn", where 'nn' is a report number from 1 to 14. As a result of this convention, the number of reports associated with one table is limited to 15.
- **.RPF** Forecast data file produced by the Reference Data Library (RDL) process - These files contain data extracted from the database, reformatted to a form acceptable for input to the model, and stored as ASCII text.
- **.RPS** Similar to .RPF files, except contain scenario data.
- **.RPT** Report text files - SWPM database reports are routinely written to this type of file as well as sent to the printer. Report text files are readable by a text editor and can be printed using the DOS PRINT or COPY command.
.SET Paradox image settings - These control numeric precision, column widths, table lengths, and other table or field defaults. They have the same file name as their corresponding data table. There is a maximum of one setting file for each data table.

.SC Paradox script - A script is similar to a program or subroutine. Script files contain either lines of Paradox Application Language (PAL) code or saved queries.

.TXT Plain text files - This type of file is created by certain data conversion processes, which may later import these text files into Paradox data tables. Plain text files are readable by any text editor.

.VPL Paradox validity check - Validity check files contain information used to set maximum, minimum, and default values for a field. They also determine if the value for a field must exist elsewhere in the database before it may be used. They have the same file name as their corresponding data table.

3.2 File Naming and Location Conventions

3.2.1 Report Files, Tables, Procedures

Report scripts: PAL\Cnnn.SC, where 'nnn' is report ID number assigned when the report is added to the Report Manager table

Report tables: WORK\Cnnn.DB

Report specifications: WORK\Cnnn.Rxx, where 'xx' is the Paradox report number

Report queries: Q\Cnnn_x.SC, where 'x' indicates order of use

Note: Some reports use common queries. In such cases, the query is named for the first report to be implemented using the query. Other queries may have compound names indicating applicability to more than one report process.

Report text files: R\Cnnn.RPT

Example: For report C312, the following file names apply -

PAL\C312.SC - script to create the report data
WORK\C312.DB - table in which report data are placed
WORK\C312.R - file containing Paradox report specification
Q\C312_2.SC - one of the queries used to extract data
3.2.2 Data Tables

Forecast data: D\'name'.DB, where 'name' is the entity to which the data applies

*Example:* D\FOREVOL.DB - table containing forecast volumes

Temporary data entry: WORK\E'name'.DB, where 'name' is defined as for forecast data above

*Example:* WORK\EFOREVOL.DB - table to which forevol data are entered for validation prior to being automatically added to d\FOREVOL.DB

Lookup, validation data: V\$'name'.DB, where 'name' is the class of item to be validated using the contents of this table

*Example:* V\$WCLASS.DB - table used to validate waste class usage throughout the application; also controls order of appearance on forms and reports

Other data: 'dir\'name'.db, where 'dir' is the directory containing the table named by 'name'

*Examples:* ACTION\ACTION.DB - table containing action item list for project team members

D\NOTEPAD.DB - table containing notes about items and issues identified during data entry

3.2.3 Queries

General purpose: Q\'name'.SC - where name usually relates to the process within which the query is used

*Examples:* Q\C310_4.SC - the fourth query used in the process to create report C310

Q\VOLSUM1.SC - a query used in creating volume summary information

Up/Download QA: UPLOAD\'name'.SC - a query used to perform quality assurance testing on data uploaded to the network
Example: UPLOAD\QA1.SC - a query used to detect missing WG forecasts in the uploaded FOREVOL table

3.2.4 PAL Scripts

General purpose: PAL\'name'.SC - where 'name' relates to the process performed by the script

Examples: PAL\HCD.SC - the script that defines actions to be taken while entering HCD data

PAL\C309.SC - the script that performs data manipulation required to produce report C309

Library procedures: UTIL\'name'.SC, PAL\'name'.SC, or 'dir'\'name'.SC - where 'name' is usually similar to the function defined by the script and 'dir' is a special purpose directory other than the UTIL or PAL directories

Examples: UTIL\RPTMGR.SC - the script that defines the Report Manager procedure, RPTMGR(), that is compiled and stored in library LIB\UTIL.LIB

PAL\FEAEDIT.SC - the script containing several procedure definitions that are compiled and written to library LIB\FEAEDIT.LIB

MENU\MENU.SC - the script defining the table-driven menu procedure that is compiled and written to LIB\UTIL.LIB

Tools: TOOLS\'name'.SC - where 'name' is the name of a development software tool created for some specific task

Example: TOOLS\RPTSUM.SC - the script that will create report totals for one or more columns during the development of a report

3.2.5 Procedure Libraries and Related Files

Libraries: LIB\'name'.LIB - where 'name' is the name assigned to the library of procedures

Examples: LIB\UTIL.LIB - a library containing a number of miscellaneous procedures

3.4
LIB\FOREDE.LIB - a library containing data entry procedures for most of the forecast data tables

Procedure source: See Section 3.2.4 above for conventions and examples

Compile scripts: LIB\name\'.SC - where 'name' is the name of the library to which procedures compiled by this script will be written

Example: LIB\UTIL.SC - a script that deletes the existing LIB\UTIL.LIB, compiles all utility procedures, and writes them into a newly created LIB\UTIL.LIB

3.2.6 Ad Hoc Files

All tables, queries, and related files are placed in subdirectories of ADHOC. These subdirectories are named with the log number assigned when the request is received. The database administrator keeps a manual log of all ad hoc requests. Explanatory notes for ad hoc tasks are kept in the ad hoc log book.

3.3 Directory Structure

The SWPM database directory structure is designed to meet a number of objectives:

• to separate data and software
• to avoid, as much as possible, hard coding disk drive designations and application directory names in programs, so that the application is relatively portable to other network drives or workstations
• to provide a coherent, easily understood structure that will simplify maintenance, administration and further development
• to facilitate uploading to and downloading from a network reference database using sound quality assurance (QA) procedures.

The general structure shown in Figure 3.1 was adapted from a column by Brian J. Smith in the June, 1990 issue of "Data Based Advisor" magazine. Certain additions were made to meet SWPM requirements and to further extend the method of disk space organization advocated in the column.
SWPMDB92

<table>
<thead>
<tr>
<th>SWPMDB92</th>
<th>ADHOC</th>
<th>Ad hoc query data and other support files</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D</td>
<td>Forecast data</td>
</tr>
<tr>
<td></td>
<td>DOC</td>
<td>Figures and tables for documentation</td>
</tr>
<tr>
<td></td>
<td>FIGURES</td>
<td>Forms for record keeping and requests</td>
</tr>
<tr>
<td></td>
<td>FORMS</td>
<td>Technical documentation</td>
</tr>
<tr>
<td></td>
<td>TECH</td>
<td>User documentation</td>
</tr>
<tr>
<td></td>
<td>USER</td>
<td>Correspondence and misc. documents</td>
</tr>
<tr>
<td></td>
<td>MISC</td>
<td>Tables containing field and table</td>
</tr>
<tr>
<td></td>
<td>TABLES</td>
<td>descriptions</td>
</tr>
<tr>
<td></td>
<td>DTSD</td>
<td>Treatment, storage and disposal data</td>
</tr>
<tr>
<td></td>
<td>LIB</td>
<td>Procedure libraries</td>
</tr>
<tr>
<td></td>
<td>MENU</td>
<td>Menu table</td>
</tr>
<tr>
<td></td>
<td>MVD</td>
<td>Model variable dictionary</td>
</tr>
<tr>
<td></td>
<td>PAL</td>
<td>Paradox Application Language scripts</td>
</tr>
<tr>
<td></td>
<td>Q</td>
<td>Queries</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>Report text files and report table</td>
</tr>
<tr>
<td></td>
<td>RDL</td>
<td>RDL output files</td>
</tr>
<tr>
<td></td>
<td>SS</td>
<td>Spreadsheets exported from tables</td>
</tr>
<tr>
<td></td>
<td>STATACC</td>
<td>Status accounting for change requests</td>
</tr>
<tr>
<td></td>
<td>TOOLS</td>
<td>Developer's software tools</td>
</tr>
<tr>
<td></td>
<td>UPLOAD</td>
<td>Tables uploaded from a local database</td>
</tr>
<tr>
<td></td>
<td>UTIL</td>
<td>Utility procedure scripts</td>
</tr>
<tr>
<td></td>
<td>V</td>
<td>Data validation and lookup tables</td>
</tr>
<tr>
<td></td>
<td>WORK</td>
<td>Work tables and report specification files</td>
</tr>
</tbody>
</table>

Figure 3.1. SWPM Database Directory Structure
The following is a description of the directories and their contents.

\SWPMDB92

The SWPMDB92 directory contains the application startup script and, occasionally, other files. No permanent data or program storage is to be done in this directory.

\SWPMDB92\ADHOC

Tables, scripts and other files used to respond to ad hoc requests for data are kept here. Subdirectories beneath this level are created as needed with names that correspond to ad hoc query log numbers assigned to each request.

\SWPMDB92\D

The D directory contains data tables and related files reflecting the FY 1992 forecast.

\SWPMDB92\DOC

The SWPMDB92 DOC directory should only contain subdirectories.

\SWPMDB92\DOC\FIGURES

Figures and diagrams appearing in the SWPM database documentation are stored here. Files in this subdirectory with ".WPG" as file name extensions were created with DrawPerfect. Diagrams created with WordPerfect use ".DOC" for extensions.

\SWPMDB92\DOC\FORMS

Forms for report requests, logging ad hoc queries, and other purposes are stored in this directory.

\SWPMDB92\DOC\MISC

Memos, DSIs, and other documents are kept in this directory.

\SWPMDB92\DOC\TABLES

This directory contains tables, forms, reports, and scripts for collecting information about tables and fields used in the database. Table 3.1 of this technical reference manual is created from the Paradox tables and other objects found here.

\SWPMDB92\DOC\TECH

This directory contains technical documentation describing processes, database structures, and other matters of interest to the database designer, administrator, and programmer.

\SWPMDB92\DOC\USER

Database user documentation files are stored in this directory.

\SWPMDB92\DTSD

Paradox tables and related files containing TSD data are kept in this directory.

3.7
The SWPMDB92\LIB directory contains procedure libraries and the necessary scripts to create and compile the libraries. The contents of the libraries are defined when certain PAL scripts are executed and written to the libraries as compiled programs. (Each individual procedure is documented in Chapter 11.)

This directory holds the menu table and related files and scripts that define the menu structure and functions available in SWPMDB92. Using this feature, menu maintenance generally involves making simple changes to a table rather than modifying program code. (See Chapter 5 for more information about the menu system.)

The SWPMDB92\MVD directory contains tables and related files that hold the model variable dictionary.

Scripts of PAL code are kept here. These scripts are for purposes that include data manipulation prior to reporting, source code for compiled procedures, and other application support.

This directory contains query scripts used in SWPM database processing. Generally, queries are called by PAL scripts stored in \SWPMDB92\PAL or by library procedures.

All report text files are stored here. Most processes that generate reports in this application store them as plain text in this directory in addition to or instead of sending the reports to the printer. Also, data and other files supporting the Report Manager function are stored in this directory.

Files generated by the Reference Data Library (RDL) process are stored here.

Spreadsheet files created during processing are placed in this directory.

Tables and related files that track status of change requests are located here.

PAL scripts and procedure source code for administrative and developmental software tools are located in this directory.
The SWPM database structure is an evolution of earlier designs. It is based on requirements posed by the sponsor, the needs of the model, the ability of the waste generator organizations to provide the requested data, and limitations imposed by database management systems. The basic forecast structure is the 19 data tables shown on the entity-relationship (E-R) diagram in Figure 3.2. This diagram depicts the types of data in the system (entities), the attributes of the entities, and the relationship between the types of data. The E-R diagram presented here displays only the major forecast tables. To avoid clutter, some minor tables are only documented in the data dictionary. The following conventions are used on the diagram:

- Boxes represent entities. For instance, WG (Waste Generator) is the name of an entity on the E-R diagram. Entities on the diagram become tables in the implemented database.

- Names within a box are the attributes of an entity. WG Abbrev is an attribute of the WG entity. Attributes define the columns (also referred to as fields) of a database table.
Figure 3.2: SWPM Process Data Entity Relationship Diagram
• Bullets in front of attribute names indicate that the field is the key for that table. When more than one field is marked, the designated fields form a compound key. Keys are unique within each table and establish the default order in which rows are displayed in the Paradox workspace.

• Heavily shaded boxes represent tables containing forecast information entered by users into the database. The more lightly shaded boxes are system tables used to enforce data validation rules and provide supporting information for various data manipulation and reporting processes.

• Numbers in brackets following each entity name are the approximate number of rows of data to be found in that table of the database. There is no enforcement of row limitations in the design. Paradox table sizes are limited only by disk space.

• The lines between entities indicate their relationships. Each line is marked with the appropriate one-to-many (1:M) or many-to-one (M:1) symbol. For example, the WG and FORECAST entities have a relationship where there can be multiple rows in the FORECAST table for each row in the WG table. The connecting lines also show that each FORECAST row relates to one or more rows in the FOREVOL table. In the database implementation, the relationships are established in queries, multi-table reports and forms, and within procedures programmed in PAL.

• Dollar signs are used in table names where the table provides a look-up function. Users do not enter data to these tables.

3.5 Data Dictionary

The data dictionary in Table 3.1 provides detailed information about the database tables and fields. It lists all tables, the row structure of each table, and includes descriptions of each table and field. All tables in Figure 3.2 are shown here, along with supporting tables that are excluded from the figure for clarity.
<table>
<thead>
<tr>
<th>Table: DOC\TABLES\TAB</th>
<th>This table contains the list of all tables in the SWPM database.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 TABLE NAME</td>
<td>A30</td>
</tr>
<tr>
<td>2 DESCRIPTION</td>
<td>A160</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table: DOC\TABLES\TABFLD</th>
<th>This table contains the list of fields in each table of the SWPM database.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 TABLE NAME</td>
<td>A30*</td>
</tr>
<tr>
<td>2 FIELD NUMBER</td>
<td>S*</td>
</tr>
<tr>
<td>3 FIELD NAME</td>
<td>A25</td>
</tr>
<tr>
<td>4 FIELD TYPE</td>
<td>A5</td>
</tr>
<tr>
<td>5 DESCRIPTION</td>
<td>A160</td>
</tr>
<tr>
<td>6 LOWVALUE</td>
<td>A20</td>
</tr>
<tr>
<td>7 HIGHVALUE</td>
<td>A20</td>
</tr>
<tr>
<td>8 DEFAULT</td>
<td>A20</td>
</tr>
<tr>
<td>9 TABLELOOKUP</td>
<td>A20</td>
</tr>
<tr>
<td>10 PICTURE</td>
<td>A20</td>
</tr>
<tr>
<td>11 REQUIRED</td>
<td>A3</td>
</tr>
</tbody>
</table>
### Table 3.1. (contd)

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV: The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2</td>
<td>WCLASS ABBREV: The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>3</td>
<td>PCD ABBREV: The abbreviation for a physical content descriptor.</td>
</tr>
<tr>
<td>4</td>
<td>CONT ABBREV: The abbreviation for a container name.</td>
</tr>
<tr>
<td>5</td>
<td>LENGTH: Length of box.</td>
</tr>
<tr>
<td>6</td>
<td>WIDTH: Width of box.</td>
</tr>
<tr>
<td>7</td>
<td>HEIGHT: Height of box.</td>
</tr>
<tr>
<td>8</td>
<td>DESCRIPTION: Description of the box.</td>
</tr>
<tr>
<td>9</td>
<td>NOTE DATE: The system supplied date at the time the note was taken.</td>
</tr>
<tr>
<td>10</td>
<td>USER NAME: Paradox user name - automatically recorded when the note is made.</td>
</tr>
</tbody>
</table>

### Table: D\CONT

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV: The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2</td>
<td>SCENARIO ABBREV: The abbreviation for a scenario name.</td>
</tr>
<tr>
<td>3</td>
<td>WCLASS ABBREV: The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>4</td>
<td>PCD ABBREV: The abbreviation for a physical content descriptor.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>CONT ABBREV</td>
<td>A14* The abbreviation for a container name.</td>
</tr>
<tr>
<td>6</td>
<td>PERCENT</td>
<td>N Decimal fraction of waste class/pcd combination packaged in this container type.</td>
</tr>
</tbody>
</table>

Table: D\CONTOTH

This table contains descriptions of "Other" containers.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV</td>
<td>A14 The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2</td>
<td>WCLASS ABBREV</td>
<td>A14 The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>3</td>
<td>PCD ABBREV</td>
<td>A14 The abbreviation for a physical content descriptor.</td>
</tr>
<tr>
<td>4</td>
<td>CONT ABBREV</td>
<td>A14 The abbreviation for a container name.</td>
</tr>
<tr>
<td>5</td>
<td>DESCRIPTION</td>
<td>A160 Description of the container listed by the waste generator as &quot;Other.&quot;</td>
</tr>
<tr>
<td>6</td>
<td>NOTE DATE</td>
<td>D System date - automatically recorded when the note is made.</td>
</tr>
<tr>
<td>7</td>
<td>USER NAME</td>
<td>A20 Paradox user name - automatically recorded when the note is made.</td>
</tr>
</tbody>
</table>

Table: D\DATACHG

This table contains data change requests to support configuration management.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CHANGE ID</td>
<td>N* The unique ID assigned to a change request.</td>
</tr>
<tr>
<td>2</td>
<td>REQUESTED BY</td>
<td>A30 The name or initials of the person requesting the data change.</td>
</tr>
<tr>
<td>3</td>
<td>REQUEST DATE</td>
<td>D The date on which the change request was made.</td>
</tr>
<tr>
<td>4</td>
<td>TITLE</td>
<td>A30 The title by which this change is to be referred.</td>
</tr>
</tbody>
</table>
### Table 3.1. (contd)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 ACTION REQUESTED</td>
<td>A1</td>
<td>The nature of the change - (A)dd, (C)hange or (D)elete.</td>
</tr>
<tr>
<td>6 DESCRIPTION</td>
<td>A255</td>
<td>A brief description of the requested change.</td>
</tr>
<tr>
<td>7 APPROVED BY</td>
<td>A30</td>
<td>The name or initials of the person approving the change.</td>
</tr>
<tr>
<td>8 APPROVED DATE</td>
<td>D</td>
<td>The date on which the approval was given.</td>
</tr>
<tr>
<td>9 ASSIGNED TO</td>
<td>A30</td>
<td>The person assigned to make the change.</td>
</tr>
<tr>
<td>10 ACTION TAKEN</td>
<td>A255</td>
<td>The description of the action taken to effect the change.</td>
</tr>
<tr>
<td>11 ACTION DATE</td>
<td>D</td>
<td>The date on which the action was completed.</td>
</tr>
<tr>
<td>12 REVIEWED BY</td>
<td>A30</td>
<td>The name or initials of the person who reviewed the change action.</td>
</tr>
<tr>
<td>13 REVIEW DATE</td>
<td>D</td>
<td>The date on which the change was reviewed.</td>
</tr>
<tr>
<td>14 REFERENCE</td>
<td>A30</td>
<td>A reference to a document or other authority on which the change was based.</td>
</tr>
<tr>
<td>15 USER ID</td>
<td>A15</td>
<td>The Paradox user ID of the person entering the change record.</td>
</tr>
<tr>
<td>16 CREATION DATE</td>
<td>D</td>
<td>The system date on which the change record was entered.</td>
</tr>
</tbody>
</table>

### Table: D\FORECAST

This table identifies all available volume forecasts in the database.

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 WG ABBREV</td>
<td>A14*</td>
<td>The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2 SCENARIO ABBREV</td>
<td>A14*</td>
<td>The abbreviation for a scenario name.</td>
</tr>
</tbody>
</table>

3.15
Table 3.1. (contd)

Table: D\FOREVOL
This table contains forecast volumes by year and waste class for each waste generator. Data are entered into this table in cubic feet.

1. W G ABBREV A14* The abbreviation for a waste generator name.
2. SCENARIO ABBREV A14* The abbreviation for a scenario name.
3. WCLASS ABBREV A14* The abbreviation for a waste class name.
4. YR A12* The year forecasted.
5. VOL N The volume of waste forecasted.

Table: D\HCD
This table holds the hazardous content percentages for the physical contents of each waste class produced by waste generators.

1. W G ABBREV A14* The abbreviation for a waste generator name.
2. SCENARIO ABBREV A14* The abbreviation for a scenario name.
3. WCLASS ABBREV A14* The abbreviation for a waste class name.
4. PCD ABBREV A14* The abbreviation for a physical content descriptor.
5. HCD ABBREV A14* The abbreviation for a hazardous content descriptor.
6. PERCENT N Decimal fraction of a waste class/pcd combination represented by this hazardous content descriptor.
### Table 3.1. (contd)

Table: D\HCDOTH

This table contains descriptions of "Other" HCD types.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 WG ABBREV</td>
<td>A14</td>
<td>The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2 WCLASS ABBREV</td>
<td>A14</td>
<td>The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>3 PCD ABBREV</td>
<td>A14</td>
<td>The abbreviation for a physical content descriptor.</td>
</tr>
<tr>
<td>4 HCD ABBREV</td>
<td>A14</td>
<td>The abbreviation for a hazardous content descriptor.</td>
</tr>
<tr>
<td>5 DESCRIPTION</td>
<td>A160</td>
<td>Description of the HCD listed by the waste generator as &quot;Other.&quot;</td>
</tr>
<tr>
<td>6 NOTE DATE</td>
<td>D</td>
<td>The system supplied date at the time the note was taken.</td>
</tr>
<tr>
<td>7 USER NAME</td>
<td>A20</td>
<td>The Paradox user name of the person taking the note.</td>
</tr>
</tbody>
</table>

Table: D\NOTEPAD

This table contains notes made using the SWPM database application notepad function.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 TABLE NAME</td>
<td>A20</td>
<td>The table name currently active (view or edit mode) when the note was taken.</td>
</tr>
<tr>
<td>2 FIELD NAME</td>
<td>A20</td>
<td>The field name in which the cursor was positioned when the note was taken.</td>
</tr>
<tr>
<td>3 FIELD CONTENT</td>
<td>A80</td>
<td>The content of the current field.</td>
</tr>
<tr>
<td>4 NOTES</td>
<td>A160</td>
<td>The text of the note.</td>
</tr>
<tr>
<td>5 NOTE DATE</td>
<td>D</td>
<td>The system supplied date at the time the note was taken.</td>
</tr>
<tr>
<td>6 USER NAME</td>
<td>A20</td>
<td>The Paradox user name of the person taking the note.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

Table: D\NOTERES
This table contains notes originally placed in D\NOTE PAD and removed after each item was resolved.

1 TABLE NAME A20 The table name currently active (view or edit mode) when the note was taken.
2 FIELD NAME A20 The field name in which the cursor was positioned when the note was taken.
3 FIELD CONTENT A80 The content of the current field.
4 NOTES A160 The text of the note.
5 NOTE DATE D The system supplied date at the time the note was taken.
6 USER NAME A20 The Paradox user name of the person taking the note.
7 RESOLUTION DATE D The date on which action was taken or completed that resolved the notepad item.
8 RESOLVING ACTION A160 A description of the action taken.
9 POSTED BY A20 The name of the user posting the resolution to the notepad item.

Table: D\PCD
This table holds the volume percentages of physical contents of each waste class produced by waste generators.

1 WG ABBREV A14* The abbreviation for a waste generator name.
2 SCENARIO ABBREV A14* The abbreviation for a scenario name.
3 PCD ABBREV A14* The abbreviation for a physical content descriptor.
4 WCLASS ABBREV A14* The abbreviation for a waste class name.
### Table 3.1. (contd)

| 5 PERCENT | N | Decimal fraction of a waste class represented by this physical content descriptor. |

**Table: D\PCDDOTH**

This table contains descriptions of "Other" PCD types.

| 1 WG ABBREV | A14 | The abbreviation for a waste generator name. |
| 2 WCLASS ABBREV | A14 | The abbreviation for a waste class name. |
| 3 PCD ABBREV | A14 | The abbreviation for a physical content descriptor. |
| 4 DESCRIPTION | A160 | The description of the PCD type that the waste generator specified as "Other." |
| 5 NOTE DATE | D | The system supplied date at the time the note was taken. |
| 6 USER NAME | A20 | The Paradox user name of the person taking the note. |

**Table: D\RAD**

This table contains the radionuclides that were reported by the waste generators for specific waste classes.

| 1 WG ABBREV | A14* | The abbreviation for a waste generator name. |
| 2 SCENARIO ABBREV | A14* | The abbreviation for a scenario name. |
| 3 WCLASS ABBREV | A14* | The abbreviation for a waste class name. |
| 4 RADIONUCLIDE | A14* | The element symbol and atomic number of the radionuclide, e.g., Pu 240. |
| 5 CONCENTRATION | N | Concentration of the radionuclide in the waste volume. |
### Table 3.1. (contd)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>EXPONENT</td>
<td>N</td>
</tr>
<tr>
<td>7</td>
<td>UNIT</td>
<td>A14</td>
</tr>
<tr>
<td>8</td>
<td>PCT</td>
<td>N</td>
</tr>
</tbody>
</table>

**Table: D\SCENARIO**

This table contains the waste generator scenarios defined in the database.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SCENARIO ABBREV</td>
<td>A14*</td>
</tr>
<tr>
<td>2</td>
<td>SCENARIO NAME</td>
<td>A32</td>
</tr>
<tr>
<td>3</td>
<td>SCENARIO BASE YR</td>
<td>A4</td>
</tr>
<tr>
<td>4</td>
<td>SCENARIO DESCRIPTION</td>
<td>A150</td>
</tr>
</tbody>
</table>

**Table: D\SOFTCHG**

This table contains software change requests to support configuration management.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CHANGE ID</td>
<td>N*</td>
</tr>
<tr>
<td>2</td>
<td>REQUESTED BY</td>
<td>A30</td>
</tr>
<tr>
<td>3</td>
<td>REQUEST DATE</td>
<td>D</td>
</tr>
<tr>
<td>4</td>
<td>TITLE</td>
<td>A30</td>
</tr>
<tr>
<td>5</td>
<td>PRIORITY</td>
<td>A1</td>
</tr>
<tr>
<td>6</td>
<td>DESCRIPTION</td>
<td>A255</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)  

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>APPROVED BY</td>
<td>A30</td>
<td>The name or initials of the person approving the change.</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>APPROVED DATE</td>
<td>D</td>
<td>The date on which the approval was given.</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ASSIGNED TO</td>
<td>A30</td>
<td>The person assigned to make the change.</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>ACTION TAKEN</td>
<td>A255</td>
<td>The description of the action taken to effect the change.</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ACTION DATE</td>
<td>D</td>
<td>The date on which the action was completed.</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>REVIEWED BY</td>
<td>A30</td>
<td>The name or initials of the person who reviewed the change action.</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>REVIEW DATE</td>
<td>D</td>
<td>The date on which the change was reviewed.</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>REFERENCE</td>
<td>A30</td>
<td>A reference to a document or other authority on which the change was based.</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>USER ID</td>
<td>A15</td>
<td>The Paradox user ID of the person entering the change record.</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>CREATION DATE</td>
<td>D</td>
<td>The system-supplied date at the time the note was taken.</td>
<td></td>
</tr>
</tbody>
</table>

Table: D\WG  
This table contains the list of all valid waste generators, their full names, classification, and group affiliation.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV</td>
<td>A14*</td>
<td>The abbreviation for a waste generator name.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>WG NAME</td>
<td>A30</td>
<td>The full waste generator name.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>WGCLASS ABBREV</td>
<td>A14</td>
<td>The abbreviation for the class to which this waste generator belongs.</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>RECEIVED DATE</td>
<td>D</td>
<td>The date on which the forecast was received from the waste generator.</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>COMPLETED DATE</td>
<td>D</td>
<td>The date when data entry for the forecast was finished.</td>
<td></td>
</tr>
</tbody>
</table>

3.21
6 WG GROUP NAME A16  The name of the WG group to which this waste generator belongs.
7 TSD WG ABBREV A14  An alternate WG name to be used during RDL processing.
8 WG CONSOL A14  An alternate name on which split waste generators can be consolidated.

Table: D\WG_TSD
This table contains the list of all valid waste generators that can appear in volume data to be passed to the RDL generator.

1 WG ABBREV A14*  The abbreviation for a waste generator name.
2 WG NAME A30  The full waste generator name.
3 WGCLASS ABBREV A14  The abbreviation for the class to which this waste generator belongs.
4 RECEIVED DATE D  Not used.
5 COMPLETED DATE D  Not used.
6 WG GROUP NAME A16  The name of the WG group to which this waste generator belongs.
7 SOURCE FORECAST A2  Not used.

Table: MENU\MENUOPT
This table contains data that determine the menu structure of the SWPM database application.

1 MENU ABBREV A10*  The name of the menu group to which this menu option belongs.
2 MENU OPT # N*  A number that establishes the order in which this item appears in its group.
3 MENU OPT A15  The name for this option that will appear as a menu selection.
### Table 3.1. (contd)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>MENU DESC</td>
<td>A70</td>
</tr>
<tr>
<td>5</td>
<td>MENU PROC</td>
<td>A80</td>
</tr>
<tr>
<td>6</td>
<td>MENU REPORT ORDER</td>
<td>N</td>
</tr>
</tbody>
</table>

---

#### Table: MVD\SUBSDICT

This table contains the definitions of model subscripts. Subscripts are parameters that are passed to a variable.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SUBS_ABBREV</td>
<td>A12*</td>
</tr>
<tr>
<td>2</td>
<td>SUBS_NAME</td>
<td>A30</td>
</tr>
<tr>
<td>3</td>
<td>SUBS_SEQ</td>
<td>N</td>
</tr>
<tr>
<td>4</td>
<td>SUBS_TYPE</td>
<td>A10</td>
</tr>
<tr>
<td>5</td>
<td>SUBS_UNITS</td>
<td>A10</td>
</tr>
<tr>
<td>6</td>
<td>SUBS_MIN_VALUE</td>
<td>N</td>
</tr>
<tr>
<td>7</td>
<td>SUBS_MAX_VALUE</td>
<td>N</td>
</tr>
<tr>
<td>8</td>
<td>SUBS_COMMENT</td>
<td>A255</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

**Table: MVD\VALDICT**

This table contains the values that a subscript may have. This is only used for subscripts with discrete (as opposed to continuous) values.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SUBS_ABBREV</td>
<td>A12* The abbreviation of subscript name.</td>
</tr>
<tr>
<td>2</td>
<td>SUBS_VALUE_SHORT</td>
<td>A12* An abbreviated value that the subscript may contain.</td>
</tr>
<tr>
<td>3</td>
<td>SUBS_VALUE_LONG</td>
<td>A40 The full length value that the subscript may contain.</td>
</tr>
<tr>
<td>4</td>
<td>SUBS_VALUE_DESC</td>
<td>A255 A description of this possible subscript value.</td>
</tr>
</tbody>
</table>

**Table: MVD\VARDICT**

This table contains the definitions of variables used by the model. Variables represent a type of information, such as waste volumes or abbreviation lists.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VAR_ABBREV</td>
<td>A20* The abbreviation for a variable name.</td>
</tr>
<tr>
<td>2</td>
<td>VAR_NAME</td>
<td>A50 The full name of the variable.</td>
</tr>
<tr>
<td>3</td>
<td>VAR_TYPE</td>
<td>A10 The data type for the specified variable.</td>
</tr>
<tr>
<td>4</td>
<td>RPF_TYPE</td>
<td>A10 The data type of the field that should be passed to the reference data library files.</td>
</tr>
<tr>
<td>5</td>
<td>VAR_RPF_USE</td>
<td>A1 A flag indicating the type of variable (waste generator or treatment).</td>
</tr>
<tr>
<td>6</td>
<td>VAR_DICT_USE</td>
<td>A1 A flag indicating whether the variable should be output to the RDL. (Some variables are for comment only).</td>
</tr>
<tr>
<td>7</td>
<td>VAR_PARAMS</td>
<td>A40 Parameters associated with this variable.</td>
</tr>
<tr>
<td>8</td>
<td>VAR_SPARSE_ARRAY</td>
<td>A1 A flag indicating if the variable is sparsely populated.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th></th>
<th>FIELD NAME</th>
<th>CHARACTER FIELD SIZE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>VAR_DEFAULT_VALUE</td>
<td>A25</td>
<td>The default value for the variable.</td>
</tr>
<tr>
<td>10</td>
<td>VAR_COMMENT</td>
<td>A255</td>
<td>The description of the variable.</td>
</tr>
<tr>
<td>11</td>
<td>VAR_UPDATE_DATE</td>
<td>D</td>
<td>The date when this variable was last updated.</td>
</tr>
<tr>
<td>12</td>
<td>VAR_UPDATE_WHO</td>
<td>A3</td>
<td>The initials of the person who performed the last update.</td>
</tr>
<tr>
<td>13</td>
<td>VAR_DELETE_DATE</td>
<td>D</td>
<td>The date when this variable was deleted.</td>
</tr>
<tr>
<td>14</td>
<td>VAR_DELETE_WHO</td>
<td>A3</td>
<td>The initials of the person who deleted this variable.</td>
</tr>
<tr>
<td>15</td>
<td>VAR_DISP_SEQ</td>
<td>N</td>
<td>The order in which this variable appears in reports.</td>
</tr>
</tbody>
</table>

Table: MVD\VARSUBS
This table is used to link variables and subscripts in many-to-many relationships.

<table>
<thead>
<tr>
<th></th>
<th>FIELD NAME</th>
<th>CHARACTER FIELD SIZE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VAR_ABBREV</td>
<td>A20*</td>
<td>The abbreviation for a variable name.</td>
</tr>
<tr>
<td>2</td>
<td>VAR_SUBS_SEQ</td>
<td>N*</td>
<td>The sequence number assigned to the variable-subscript relationship to reflect the order of the subscript list.</td>
</tr>
<tr>
<td>3</td>
<td>SUBS_ABBREV</td>
<td>A12</td>
<td>The abbreviation of the subscript name.</td>
</tr>
</tbody>
</table>

Table: R\REPORT
This table holds a list of all database reports available from the report manager.

<table>
<thead>
<tr>
<th></th>
<th>FIELD NAME</th>
<th>CHARACTER FIELD SIZE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RPT ID</td>
<td>A5*</td>
<td>The identifier assigned to this report format.</td>
</tr>
<tr>
<td>2</td>
<td>RPT ABBREV</td>
<td>A8</td>
<td>The table name from which the report will be generated.</td>
</tr>
<tr>
<td>3</td>
<td>REPORT NUMBER</td>
<td>A2</td>
<td>The report number given to this report specification (R, 1, 2, 3, etc.).</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
</table>
| 4 | RPT NAME  
A45  | The title of this report - a condensed version of the text of the 1st title line. |
| 5 | RPT SCRIPT  
A8  | The script or procedure name to be invoked to create the report.               |
| 6 | COMMENTS  
A65  | Comments about the report content or process.                                 |
| 7 | FORMAT NOTES  
A65  | Notes concerning the format of the report.                                   |
| 8 | DATE MODIFIED  
D  | The last date on which the report format was modified - manually maintained.  |
| 9 | DATE LAST RUN  
D  | The last date on which the report was run - automatically maintained.         |
| 10 | RUN TIME  
A10  | The time in minutes and tenths of minutes that the report process took when last run. |
| 11 | TABLE LOCATION  
A16  | The subdirectory in which the report table named in Rpt Abbrev is found - default is WORK\. |
| 12 | RPT FILE SIZE  
N  | The amount of space in bytes that DOS has allocated to the file containing the report output. |

Table: V\$CONT  
This lookup table holds the information about each valid container type.

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
</table>
| 1 | CONT ABBREV  
A14*  | The abbreviation for a container name.                                      |
| 2 | CONT NAME  
A32  | The full container name.                                                    |
| 3 | CONT REPORT ORDER  
N  | The order in which data for this container type are to appear on reports.    |
| 4 | CONT FORM ORDER  
N  | The order in which data for this container are to appear or be entered on forms. |
| 5 | CONT GROUP ABBREV  
A14  | The name of the container group to which this container belongs.             |

3.26
<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>CONT DESCRIPTION</td>
<td>A255</td>
<td>The description of this container type.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table: V\$HCD

This lookup table contains information about each valid hazardous content descriptor.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HCD ABBREV</td>
<td>A14*</td>
<td>The abbreviation for a hazardous content descriptor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>HCD NAME</td>
<td>A32</td>
<td>The full name of the hazardous content descriptor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>HCD REPORT ORDER</td>
<td>N</td>
<td>The order in which the data for this HCD are to appear on reports.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>HCD FORM ORDER</td>
<td>N</td>
<td>The order in which the data for this HCD are to appear or be entered on forms.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>HCD GROUP ABBREV</td>
<td>A14</td>
<td>The name of the HCD group to which this HCD belongs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>HCD DESCRIPTION</td>
<td>A255</td>
<td>The description of this HCD.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table: V\$PCD

This lookup table contains information about each valid physical content descriptor.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PCD ABBREV</td>
<td>A14*</td>
<td>The abbreviation for a physical content descriptor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>PCD NAME</td>
<td>A32</td>
<td>The full name of the physical content descriptor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PCD REPORT ORDER</td>
<td>N</td>
<td>The order in which data for this PCD appear on reports.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PCD FORM ORDER</td>
<td>N</td>
<td>The order in which data for this PCD appear or are entered on forms.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>PCD GROUP ABBREV</td>
<td>A14</td>
<td>The name of the PCD group to which this PCD belongs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>PCD DESCRIPTION</td>
<td>A255</td>
<td>The description of this PCD.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th>Table: V$RAD</th>
<th>This lookup table contains a list of valid radionuclide names.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 RADIONUCLIDE</td>
<td>A14* The element symbol and atomic number of the radionuclide, e.g., Pu 240.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table: V$RADUNIT</th>
<th>This lookup table contains a list of valid radionuclide units of measure.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 UNIT</td>
<td>A14* Unit of measure of the concentration, such as ci/g (curies per gram) or &quot;Not specified.&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table: V$TSDWC</th>
<th>This lookup table contains information about valid treatment, storage, and disposal (TSD) waste classes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 TSD WC ABBREV</td>
<td>A12* The abbreviation for a TSD waste class name.</td>
</tr>
<tr>
<td>2 TSD WC NAME</td>
<td>A32 The full name of the TSD waste class.</td>
</tr>
<tr>
<td>3 TSD WC FORM ORDER</td>
<td>N The order in which data about this TSD waste class appear or are entered on forms.</td>
</tr>
<tr>
<td>4 TSD WC REPORT ORDER</td>
<td>N The order in which data about this TSD waste class appear on reports.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

Table: V\$VALCHK
This table contains the list of validity checks that may be defined for Paradox table columns and the report order for documentation.

<table>
<thead>
<tr>
<th>VALCHK TYPE</th>
<th>ORDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>A15*</td>
<td>N</td>
</tr>
<tr>
<td>The type of validity check.</td>
<td>The order in which the validity check appears on reports generated for documentation.</td>
</tr>
</tbody>
</table>

Table: V\$WCAT
This lookup table contains information about valid waste categories.

<table>
<thead>
<tr>
<th>WCAT ABBREV</th>
<th>WCAT NAME</th>
<th>WCAT FORM ORDER</th>
<th>WCAT REPORT ORDER</th>
<th>ALT REPORT ORDER</th>
<th>TYPE</th>
<th>WCAT GROUP</th>
</tr>
</thead>
<tbody>
<tr>
<td>A14*</td>
<td>A32</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>A15</td>
<td>A14</td>
</tr>
<tr>
<td>The abbreviation for a waste category.</td>
<td>The full name of the waste category.</td>
<td>The order in which data for this waste category appear or are entered on forms.</td>
<td>The order in which data for this waste category appear on reports. Categories of a similar handling type will appear together with this ordering.</td>
<td>The order in which data for this waste category appear on reports. Categories of similar composition will appear together with this ordering.</td>
<td>The type of handling to which this waste category is subject - remote or contact.</td>
<td>The waste group to which this waste category belongs.</td>
</tr>
</tbody>
</table>

3.29
<table>
<thead>
<tr>
<th>Table: V$WCATGRP</th>
<th>This lookup table defines the available groups of waste categories.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 WCAT GROUP</td>
<td>A14* The abbreviated name of the waste group - established by the nature of the waste without regard to level or handling requirement.</td>
</tr>
<tr>
<td>2 REPORT ORDER</td>
<td>N The order relative to other waste groups that this group appears on reports.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table: V$WCLASS</th>
<th>This lookup table contains information about valid waste classes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 WCLASS ABBREV</td>
<td>A14* The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>2 WCLASS NAME</td>
<td>A32 The full name of the waste class.</td>
</tr>
<tr>
<td>3 WCAT ABBREV</td>
<td>A14 The abbreviation for a waste category.</td>
</tr>
<tr>
<td>4 TSD WCLASS ABBREV</td>
<td>A14 The abbreviation for the equivalent TSD waste class.</td>
</tr>
<tr>
<td>5 WCLASS REPORT ORDER</td>
<td>N The order in which data about this waste class appear on reports.</td>
</tr>
<tr>
<td>6 WCLASS FORM ORDER</td>
<td>N The order in which data about this waste class appear or are entered on forms.</td>
</tr>
<tr>
<td>7 ALT REPORT ORDER</td>
<td>N Reserved for future use.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table: V$WGCLASS</th>
<th>This lookup table contains information about valid waste generator classes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 WGCLASS ABBREV</td>
<td>A14* The abbreviation for a waste generator class.</td>
</tr>
<tr>
<td>2 WGCLASS NAME</td>
<td>A24 The full name for the waste generator class.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>WGCLASS REPORT ORDER</td>
</tr>
<tr>
<td>4</td>
<td>WGCLASS FORM ORDER</td>
</tr>
<tr>
<td>5</td>
<td>WGCLASS DESCRIPTION</td>
</tr>
</tbody>
</table>

Table: V\$WGGROUP

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG GROUP NAME</td>
</tr>
<tr>
<td>2</td>
<td>WG GROUP REPORT ORDER</td>
</tr>
</tbody>
</table>

Table: V\$WSTSTRM

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WASTE CLASS</td>
</tr>
<tr>
<td>2</td>
<td>CONTAINER</td>
</tr>
<tr>
<td>3</td>
<td>WASTE STREAM</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCLASS ABBREV</td>
<td>The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>PCD ABBREV</td>
<td>The abbreviation for a physical content descriptor.</td>
</tr>
<tr>
<td>55G DRUM</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>ODD DRUM</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>4X4X8 BOX</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>&lt; 128 BOX</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>128-500 BOX</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>501-1000 BOX</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>&gt; 1000 BOX</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>STANDARD BOX</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>TRUPACT II</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>PLASTIC WRAP</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>DUMP TRUCK</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>SEALED EQUIP</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
<tr>
<td>OTHER</td>
<td>Fraction of the PCD in this container type.</td>
</tr>
</tbody>
</table>

This table receives all container data entry prior to data validation checking and passing of data to D\CONT for permanent storage.

Table: WORK\ECONT
<table>
<thead>
<tr>
<th>Table: WORK\FOREVOL</th>
<th>Table 3.1. (contd)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 WCLASS ABBREV</td>
<td>This table receives all volume data entry prior to data validation checking and passing of data to D\FOREVOL for permanent storage.</td>
</tr>
<tr>
<td>2 HELD</td>
<td>A14 The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>20 2010</td>
<td>N Volume forecast for the year 2010.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

<table>
<thead>
<tr>
<th>Year</th>
<th>Volume Forecast</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011</td>
<td>N Volume forecast for the year 2011.</td>
</tr>
<tr>
<td>2012</td>
<td>N Volume forecast for the year 2012.</td>
</tr>
<tr>
<td>2014</td>
<td>N Volume forecast for the year 2014.</td>
</tr>
<tr>
<td>2015</td>
<td>N Volume forecast for the year 2015.</td>
</tr>
<tr>
<td>2021</td>
<td>N Volume forecast for the year 2021.</td>
</tr>
<tr>
<td>2022</td>
<td>N Volume forecast for the year 2022.</td>
</tr>
</tbody>
</table>

Table: WORK\EHCD

This table receives all HCD data entry prior to data validation checking and passing of data to D\HCD for permanent storage.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WCLASS ABBREV</td>
<td>A14 The abbreviation for a waste class name.</td>
</tr>
<tr>
<td>PCD ABBREV</td>
<td>A14 The abbreviation for a physical content descriptor.</td>
</tr>
<tr>
<td>FLAMMABLE</td>
<td>N Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>OXIDIZER</td>
<td>N Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>COMBUSTIBLE</td>
<td>N Fraction of the PCD contaminated with this HCD.</td>
</tr>
</tbody>
</table>

3.34
<table>
<thead>
<tr>
<th>6</th>
<th>ACIDS</th>
<th>N</th>
<th>Fraction of the PCD contaminated with this HCD.</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>ALKALINES</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>8</td>
<td>WATER REACTIVE</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>9</td>
<td>EXPLOSIVE</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>10</td>
<td>CYANIDE</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>11</td>
<td>TC METAL</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>12</td>
<td>TC PESTICIDE</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>13</td>
<td>TC ORGANIC</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>14</td>
<td>SPENT SOLVENT</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>15</td>
<td>P &amp; U LISTED</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>16</td>
<td>PCB &lt; 50 PPM</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>17</td>
<td>PCB &gt;=50,&lt; 500</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>18</td>
<td>PCB &gt;= 500 PPM</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
<tr>
<td>19</td>
<td>OTHER</td>
<td>N</td>
<td>Fraction of the PCD contaminated with this HCD.</td>
</tr>
</tbody>
</table>
Table 3.1. (contd)

Table: WORK\EPCD

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PCD ABBREV</td>
</tr>
<tr>
<td>1</td>
<td>A14 The abbreviation for a physical content descriptor.</td>
</tr>
<tr>
<td>2</td>
<td>CH LLW I</td>
</tr>
<tr>
<td>2</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>3</td>
<td>CH LLW III</td>
</tr>
<tr>
<td>3</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>4</td>
<td>CH LLW GTCIII</td>
</tr>
<tr>
<td>4</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>5</td>
<td>RH LLW I</td>
</tr>
<tr>
<td>5</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>6</td>
<td>RH LLW III</td>
</tr>
<tr>
<td>6</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>7</td>
<td>RH LLW GTCIII</td>
</tr>
<tr>
<td>7</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>8</td>
<td>CH LLMW I</td>
</tr>
<tr>
<td>8</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>9</td>
<td>CH LLMW III</td>
</tr>
<tr>
<td>9</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>10</td>
<td>CH LLMW GTCIII</td>
</tr>
<tr>
<td>10</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>11</td>
<td>RH LLMW I</td>
</tr>
<tr>
<td>11</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>12</td>
<td>RH LLMW III</td>
</tr>
<tr>
<td>12</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
<tr>
<td>13</td>
<td>RH LLMW GTCIII</td>
</tr>
<tr>
<td>13</td>
<td>N The fraction of the waste class made up of this PCD.</td>
</tr>
</tbody>
</table>

This table receives all PCD data entry prior to data validation checking and passing of data to D\PCD for permanent storage.
### Table 3.1. (contd)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>HAZ</td>
<td>N</td>
</tr>
<tr>
<td>15</td>
<td>CH_TRU</td>
<td>N</td>
</tr>
<tr>
<td>16</td>
<td>RH_TRU</td>
<td>N</td>
</tr>
<tr>
<td>17</td>
<td>CH_TRUM</td>
<td>N</td>
</tr>
<tr>
<td>18</td>
<td>RH_TRUM</td>
<td>N</td>
</tr>
</tbody>
</table>

---

**Table: WORK\ERAD**

This table receives all radionuclide data entry prior to data validation checking and passing of data to D\RAD for permanent storage.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WCLASS ABBREV</td>
<td>A14</td>
</tr>
<tr>
<td>2</td>
<td>RADIONUCLIDE</td>
<td>A14</td>
</tr>
<tr>
<td>3</td>
<td>CONCENTRATION</td>
<td>N</td>
</tr>
<tr>
<td>4</td>
<td>EXPONENT</td>
<td>N</td>
</tr>
<tr>
<td>5</td>
<td>UNIT</td>
<td>A14</td>
</tr>
<tr>
<td>6</td>
<td>PCT</td>
<td>N</td>
</tr>
</tbody>
</table>
### Table 3.1 (contd)

**Table: WORK\FORECONT**

- **1.** WG ABBREV: A14 The abbreviation for a waste generator name.
- **2.** WCLASS ABBREV: A14 The abbreviation for the waste class name.
- **3.** YR: A12 The year to which the volume in this row applies.
- **4.** PCD ABBREV: A14 The abbreviation for the physical content descriptor characterizing this waste volume.
- **5.** CONT ABBREV: A14 The abbreviation for the container type in which this waste volume is shipped.
- **6.** VOL: N The volume of waste to be shipped this year, in cubic meters.

**Table: WORK\FOREHCD**

- **1.** WG ABBREV: A14 The abbreviation for a waste generator name.
- **2.** WCLASS ABBREV: A14 The abbreviation for the waste class name.
- **3.** YR: A12 The year to which the volume in this row applies.
- **4.** PCD ABBREV: A14 The abbreviation for the physical content descriptor characterizing this waste volume.
- **5.** HCD ABBREV: A14 The abbreviation for the hazardous content descriptor characterizing this waste volume.
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>VOL</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>The volume of waste to be shipped this year, in cubic meters.</td>
</tr>
</tbody>
</table>

**Table: WORK\FOREPCD**

This table contains the volumes reported by each WG for waste class/PCD type combinations, 1 row per year.

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2</td>
<td>SCENARIO ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The scenario to which this forecasted volume applies - initially &quot;BASELINE&quot;.</td>
</tr>
<tr>
<td>3</td>
<td>WCLASS ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The abbreviation for the waste class name.</td>
</tr>
<tr>
<td>4</td>
<td>YR</td>
</tr>
<tr>
<td></td>
<td>A12</td>
</tr>
<tr>
<td></td>
<td>The year to which the volume in this row applies.</td>
</tr>
<tr>
<td>5</td>
<td>PCD ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The abbreviation for the physical content descriptor characterizing this waste volume.</td>
</tr>
<tr>
<td>6</td>
<td>VOL</td>
</tr>
<tr>
<td></td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>The volume of waste to be shipped this year, in cubic meters.</td>
</tr>
</tbody>
</table>

**Table: WORK\FVCONV**

This table receives forecast volume data after it is converted from cubic feet to cubic meters.

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The abbreviation for a waste generator name.</td>
</tr>
<tr>
<td>2</td>
<td>SCENARIO ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The scenario to which this forecasted volume applies - initially &quot;BASELINE&quot;.</td>
</tr>
<tr>
<td>3</td>
<td>WCLASS ABBREV</td>
</tr>
<tr>
<td></td>
<td>A14</td>
</tr>
<tr>
<td></td>
<td>The waste stream name to which this volume belongs.</td>
</tr>
<tr>
<td>4</td>
<td>YR</td>
</tr>
<tr>
<td></td>
<td>A12</td>
</tr>
<tr>
<td></td>
<td>The year to which the volume in this row applies.</td>
</tr>
</tbody>
</table>
Table 3.1 (contd)

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>WGCLASS ABBREV A14</td>
</tr>
<tr>
<td>6</td>
<td>VOL N</td>
</tr>
</tbody>
</table>

Table: WORK\FVCONVX

This is a cross-tabulated version of work\FVCONV.

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WG ABBREV A14</td>
</tr>
<tr>
<td>2</td>
<td>SCENARIO ABBREV A14</td>
</tr>
<tr>
<td>3</td>
<td>WCLASS ABBREV A14</td>
</tr>
<tr>
<td>4</td>
<td>WGCLASS ABBREV A14</td>
</tr>
</tbody>
</table>

3.40
<table>
<thead>
<tr>
<th>Year</th>
<th>Volume Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td>Volume in cubic meters for 2012.</td>
</tr>
<tr>
<td>2013</td>
<td>Volume in cubic meters for 2013.</td>
</tr>
<tr>
<td>2015</td>
<td>Volume in cubic meters for 2015.</td>
</tr>
<tr>
<td>2021</td>
<td>Volume in cubic meters for 2021.</td>
</tr>
<tr>
<td>2022</td>
<td>Volume in cubic meters for 2022.</td>
</tr>
<tr>
<td>HELD</td>
<td>Volume forecast for held waste.</td>
</tr>
<tr>
<td>TOTAL</td>
<td>Total volume for the 30-year forecast, including HELD.</td>
</tr>
</tbody>
</table>
4.0 Application Startup Scripts

There are a number of events that occur during the initialization of the application. These events are controlled by two Paradox Application Language (PAL) scripts, the first of which is played by the user from the standard Paradox menu within any available SWPM database application directory. The second script is specific to the particular application version selected. This chapter describes in detail the actions performed by these scripts and includes listings of the PAL code that they contain.

4.1 Version Selection Script - SWPM.SC

When the user is ready to begin using a SWPM database application, he or she will have executed Paradox and established a SWPM database directory as the current working directory for the Paradox session. In each version of the database, the only script in the application will be SWPM.SC. This allows easy execution of the script to be played, since it will always be the default selection, even when the user has INIT.SC or INSTANT.SC scripts in a private directory.

Once executed, SWPM.SC presents a menu of database versions. In response to a selection, the appropriate application directory is established as current and that version's copy of STARTSWP.SC is played. See Figure 4.1 for a listing of SWPM.SC

4.2 Application Execution Script - STARTSWP.SC

This script is located in the PAL\subdirectory of the application directory that contains the selected version of the database. It performs a number of actions, some of which are specific to the particular version chosen. The actions taken are summarized below. (See Figure 4.2 for a complete listing of PAL\STARTSWP.SC for Version 1.4 of the database application.)

- All table images in the workspace are removed. All variables and procedures currently in memory are released.

- The AUTOLIB variable is defined with the names of the required procedure libraries. If the current user is a database administrator, then the AUTOLIB value includes an extra library of procedures used in development and administrative activities.

- The user's password is obtained and tested against a commonly used but protected database table. If the password does not grant access, the user is given two more attempts to enter a valid password before the script terminates.

- The notepad hotkey is established as [Ctrl-N]. This allows the notepad to be invoked when any application database table is being viewed while the application itself is not running. When the application does have control and the user is editing a table, [Ctrl-N] also invokes the notepad
although the means of implementing this feature is somewhat different. See Chapter 7 for more information about the data entry notepad.

- A set of global variables are defined. Most of these are names of tables and are intended to allow the changing of underlying table names. The renaming of tables should be done with caution however. Paradox query scripts do not support the use of variables to define table names. Each occurrence in a query of a table name to be changed must be located and modified accordingly.

- The location of the selected database application is checked. By convention, if the database is located on logical drive J, then the application assumes that it is using the official network version that is subject to standard configuration management practices. If the current directory is located anywhere with a drive designation other than J, then use of a local copy is assumed. These assumptions are used to establish values for two variables, one of which displays a location message on menu screens and the other which serves as a label to appear on reports as a source indication.

- Finally, the procedure MENUEXE() is executed to invoke the table-based menuing system used within the application.
;************************************************************************************************
; Procedure name: SWPM.SC
;
; Author: Cary Blackburn - based on work by Frank Carr
;
; (c) 1992 by Pacific Northwest Laboratory
;
; Purpose: Provide SWPM startup menu to request database version
;          user wants via menu. Then set the appropriate default
directory and run the SWPM menu procedure.
;
; Parameters: None
;
; History:
;
; 05/07/91 - created by Cary Blackburn
; 12/29/91 - added baseline selection
; 06/15/92 - added FY92 selection - with not-ready msg
; 06/15/92 - activated FY92 selection
;************************************************************************************************

While True

Showmenu
   "1) FY89(v1.1)" : "FY-1989 version of SWPMDB",
   "2) FY90(v1.2)" : "FY-1990 version of SWPMDB",
   "3) FY91(v1.3)" : "FY-1991 version of SWPMDB",
   "4) FY92(v1.4)" : "FY-1992 version of SWPMDB",
   "5) Baseline" : "SWPM Baseline Database",
   "Quit" : "Quit without going into a SWPM database"

To choice

Switch

Case choice = "1) FY89(v1.1)"

; setdir "\\SWPMDB"
; play "SCRIPTS\STSWPM"
beep
message "Sorry, FY89 database has been archived. (ENTER to continue)"
I = getchar()
release vars I

Figure 4.1. Listing of SWPM.SC
Case choice = "2) FY90(v1.2)" :
    setdir "\\SWPMDB90"
    play "PAL\\STARTSWP"
    beep
    message "Sorry, FY90 database has been archived. (ENTER to continue)"
    I = getchar()
    release vars I
    clear

Case choice = "3) FY91(v1.3)" :
    setdir "\\SWPMDB91"
    Play "PAL\\STARTSWP"
    clear

Case choice = "4) FY92(v1.4)" :
    setdir "\\SWPMDB92"
    Play "PAL\\STARTSWP"
    clear

Case choice = "5) Baseline" :
    setdir "\\SWPMBL"
    Play "PAL\\STARTSWP"
    clear

Case choice = "Quit" or choice = "Esc" :
    ; message "Current directory is " + directory() + ". (ENTER to continue)"
    message "Current directory is " + directory()
    sleep 1500
    ; I = getchar()
    ; release vars I

Return
Endswitch
Endwhile

Figure 4.1. (cont)
; Procedure name: STARTSWP.SC
; Author: Cary Blackburn
; (c) 1992 by Pacific Northwest Laboratory
; Purpose: Initializes SWPMDB Version 1.4 (FY-1992) Application
; Parameters: None
; History:
; 06/25/92 - modified by Cary Blackburn for the FY92 version of the database
; 01/05/93 - modified by Cary Blackburn to prevent 4.0 users from attempting to use the application
; 07/16/93 - modified by Joyce Jamison to remove case sensitivity of passwords.

clearall
release procs all
release vars all

; Test Paradox version - has to be 3.5
If version() <> 3.5 THEN
  Beep
  message "Sorry - this application supported by PDOX3.5 only."
  i-getchar()
  return
ENDIF

; Establish accessible libraries by defining value for AUTOLIB variable

Figure 4.2. Listing of STARTSWP.SC

4.5
Idir = "\swpmdb92\"

IF username() = "CL BLACKBURN" THEN
    AutoLib= ldir + "lib\util," + ldir + "lib\RDL," + ldir + "lib\admin," + ldir + "lib\feaedit," + ldir + "lib\forede," + ldir + "lib\detools"
ELSE
    AutoLib= ldir + "lib\util," + ldir + "lib\RDL," + ldir + "lib\feaedit," + ldir + "lib\forede," + ldir + "lib\detools"
ENDIF

; Get a password from the user and test it on a v1.4 file that everyone should have access to with at least read only privileges.
; Allow 3 attempts to enter a valid password before giving up.
pswdcntr = 1

WHILE TRUE
    ;Password loop
    clear
    clear @1,0 ?? "Enter your password: "
    style attribute 64 + 4
    accept "A12" picture "*?" to xpswd
    style
    IF not isassigned(xpswd) or isblank(xpswd) or not retval THEN
        release vars xpswd
        return
    ENDIF
    menu (Tools) (More) (Protect) (ClearPasswords)
    password lower(xpswd);No case sensitivity of password
    release vars xpswd

Figure 4.2. (Contd)

4.6
IF not tablerights("d\forevol","InsDel") THEN
  IF not tablerights("d\forevol","All") THEN
    IF not tablerights("d\forevol","ReadOnly") THEN
      beep
      IF pswdcntr >= 3 THEN
        release vars pswdcntr
        return
      ELSE
        Message "Try again. Password is invalid."
        sleep 1000
        pswdcntr = pswdcntr + 1
      ENDIF
    ELSE
      Quitloop
    ENDIF
  ELSE
    Quitloop
  ENDIF
ELSE
  Quitloop
ENDIF
ENDWHILE

; Establish Notepad hotkey as [Ctrl N] - the notepad is not often invoked via hotkey, but this makes [Ctrl N] consistent with usage within data entry processes.
Setkey 14 NOTEPAD()

; Establish global variables

; General system variables:

GV_SysAbr = "SWPM DB"
GV_SysNam = "FY-1992 Solid Waste Projection Model Database"

Figure 4.2. (Contd)
; Variables containing table names

GV $WGClass = "V\$WGClass"
GV $WGGroup = "V\$WGGroup"
GV $WCat = "V\$WCat"
GV $WClass = "V\$WClass"
GV $HCD = "V\$HCD"
GV $Cont = "V\$Cont"
GV $PCD = "V\$PCD"
GV $WCATGrp = "V\$WCATGrp"
GV $Rad = "V\$Rad"
GV $RadUnit = "V\$Radunit"
GV WG = "D\WG"
GV WGTSD = "D\WG_TSD"
GV Scenario = "D\Scenario"
GV Forecast = "D\Forecast"
GV ForeVol = "D\ForeVol"
GV PCD = "D\PCD"
GV HCD = "D\HCD"
GV Cont = "D\Cont"
GV Rad = "D\Rad"
GV Oper = "DTSD\Oper"
GV Distf = "DTSD\DIST_F"
GV OperFlow = "DTSD\OperFlow"
GV OperCap = "DTSD\OperCap"
GV Notes = "D\Notepad"
GV!ForeVol = "Work\EForeVol"
GV!PCD = "Work\EPcd"
GV!HCD = "Work\EHCD"
GV!Cont = "Work\ECont"
GV!Rad = "Work\ERad"

; Other global variables

GV EdType = ""
TKLibName = "lib\DETools"

; Establish location of this database and set report flag accordingly

Figure 4.2. (Contd)
IF upper(substr(directory(),1,2)) = "J:" THEN
  rptsrс = "Base 03/22"
  GV_where_am_I = "* Network *"
ELSE
  rptsrс = "Unofficial"
  GV_where_am_I = "* Local Copy *"
ENDIF

;-----------------------------------------------------------------------
;Kick off the menu procedure

MenuExe ( "MAIN", GV_SysNam, "Main Menu", GV_where_am_I, TRUE )

Figure 4.2. (Contd)
5.0 Database Processes and Menus

The primary processes performed by the SWPM database are shown in Figure 1.3. These processes correspond approximately to menu options on the main menu. Selecting one of these options displays another menu or initiates some activity. The SWPM menu structure is shown in Figure 5.1 below. Section 5.1 describes each of the processes shown as the lowest level of each branch of the menu diagram.

5.1 Process Dictionary

This dictionary provides a description for each of the processes shown as bottom level selections in the menu structure diagram. Each menu selection that does not lead to another menu is listed in the dictionary along with a brief description and the name of the script or procedure invoked. For further information about the workings of each process, the scripts themselves contain descriptive comments and a file of report process notes is kept in the project notebook.

Unless otherwise noted, PAL scripts that are executed with the PLAY command can be found in \SWPMDB92\PAL. For procedures, the library containing them is listed next to the procedure names. Source code for compiled procedures may be located in \SWPMDB92\UTIL or \SWPMDB92\PAL. To locate a source script, see the entry for the procedure in Chapter 11.

Menu path: Forecast\IDFore -
Main script/procedure: ForeEdit()
Called scripts/procedures:
Tables, forms and reports:
select WG/forecast combination for data entry/update or data validation reporting

Menu path: Forecast\ForeVol -
Main script/procedure: Forevol_DE()
Called scripts/procedures:
Tables, forms and reports:
entry/update of volume data for the WG/forecast combination selected in IDFore above. Activity is performed in a work table using Data Entry Toolkit procedures after which data are added back to d\FOREVOL. See Appendix C, Section 8, for technical notes on the Data Entry Toolkit implementation.

d\FOREVOL, work\EFOREVOL
SWPM Main Menu

- **FORECAST**
  - IDFore: Select WG name and scenario
  - ForeVol: Maintain forecast volumes
  - PCD: Maintain PCD data
  - HCD: Maintain HCD data
  - Cont: Maintain Container data
  - Rad: Maintain Radionuclide data

- **TSD**
  - Oper: Maintain Oper table
  - Dist F: Maintain Dist. Fraction table
  - OperFlow: Maintain Operation Flow table
  - OperCap: Maintain Operation Cap. table

- **RDL**
  - Produce Ref. Data Library files

- **Codes**
  - WG: Maintain WG table
    - WClass: Maintain Waste Class table
    - WCat: Maintain Waste Category table
    - TSD_WClass: Maintain TSD Waste Class table
  - Types
    - PCD: Maintain PCD Types table
    - HCD: Maintain HCD Types table
    - Cont: Maintain Container types table
  - Groups
    - WG_Group: Maintain WG Group table
    - WG_Class: Maintain WG Class table
  - Scenario: Maintain Scenario table
  - Forecast: Maintain Forecast table
  - Rad: Maintain valid rad. names list
  - Units: Maintain units of measure list

- **Report_Manager**
  - (select rpt)
    - Prtnew: Process, then print report
    - New: Process, then store report text
    - Reprint: Reprint stored report text
    - Store: Store report text
    - Table: Return to report table
    - Quit: Quit Report Manager

- **Notes**
  - Notepad: Maintain general notepad
    - PCD_Oth: Maintain "OTHER" PCD notepad
    - Cont_Oth: Maintain "OTHER", cont. notepad
    - HCD_Oth: Maintain "OTHER" HCD notepad
    - Box_Dim: Maintain box dimensions notepad

- **Admin**
  - StatAcc: Status of change requests
  - TabDoc: Automatic table documentor
  - Valchk: Validity check documentor

**Figure 5.1** SWPM Menu Structure
entry/update of PCD percentage data for the WG/forecast combination selected in IDFore above. Activity is performed in a work table using Data Entry Toolkit procedures after which data are added back to d\PCD.

PCD_DE
d\PCD, work\EPCD

entry/update of HCD percentage data for the WG/forecast combination selected in IDFore above. Activity is performed in a work table using Data Entry Toolkit procedures after which data are added back to d\HCD.

HCD_DE
d\HCD, work\EHCD

entry/update of container percentage data for the WG/forecast combination selected in IDFore above. Activity is performed in a work table using Data Entry Toolkit procedures after which data are added back to d\Cont.

Cont_DE
d\CONT, work\ECONT

entry/update of radionuclide data for the WG/forecast combination selected in IDFore above. Activity is performed in a work table using Data Entry Toolkit procedures after which data are added back to d\Rad.

Rad.DE
d\RAD, work\ERAD

5.3
Menu path: **TSD\Oper** - Maintain TSD operations table

Main script/procedure: TSDEdit()
Called scripts/procedures: dtsd\OPER
Tables, forms and reports:

Menu path: **TSD\Dist_F** - Maintain TSD distribution fraction table

Main script/procedure: TSDEdit()
Called scripts/procedures: dtsd\DIST_F
Tables, forms and reports:

Menu path: **TSD\OperFlow** - Maintain TSD operation flow table

Main script/procedure: TSDEdit()
Called scripts/procedures: dtsd\OPERFLOW
Tables, forms and reports:

Menu path: **TSD\OperCap** - Maintain TSD operation capacities table

Main script/procedure: TSDEdit()
Called scripts/procedures: dtsd\OPERCAP
Tables, forms and reports:

Menu path: **RDL** - Executes reference data library creation process (This process is detailed in Section 10.3.)

Main script/procedure: RDLExec()
Called scripts/procedures: WriteRPS(), WriteRPF() and certain queries
Tables, forms and reports:
input tables: work\FVCONVX, dtsd\OPER, mvd\VARDICT, d\WG, dtsd\OPERFLOW, dtsd\OPERCAP, dtsd\DIST_F, various work tables
output tables: work\RPS, work\RPF (and text file versions)

Menu path: **Codes\WG** - Maintain waste generator table

Main script/procedure: FeaEdit()
Called scripts/procedures: d\WG
Tables, forms and reports:
Menu path: **Codes\WClass\WClass** - Maintain waste class table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$WCLASS

Menu path: **Codes\WClass\WCat** - Maintain waste category table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$WCAT

Menu path: **Codes\WClass\TSD_WClass** - Maintain TSD waste class table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$TSDWC

Menu path: **Codes\Types\PCD** - Maintain PCD types table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$PCD

Menu path: **Codes\Types\HCD** - Maintain HCD types table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$HCD

Menu path: **Codes\Types\Cont** - Maintain container types table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$CONT

Menu path: **Codes\Groups\WG_Group** - Maintain waste generator group table

Main script/procedure: FeaEdit()
Called scripts/procedures: 
Tables, forms and reports: \$WGGROUP

5.5
Menu path: Codes\Groups\WG_Class -
Main script/procedure: FeaEdit()
Called scripts/procedures:
Tables, forms and reports: \$WGCLASS
Maintain waste generator class table

Menu path: Codes\Scenario -
Main script/procedure: FeaEdit()
Called scripts/procedures:
Tables, forms and reports: \$SCENARIO
Maintain forecast scenario table

Menu path: Codes\Forecast -
Main script/procedure: FeaEdit()
Called scripts/procedures:
Tables, forms and reports: \$FORECAST
Maintain forecast table (WG and scenario association)

Menu path: Rad\Radionuclides -
Main script/procedure: FeaEdit()
Called scripts/procedures:
Tables, forms and reports: \$RAD
Maintain radionuclide names table

Menu path: Rad\Units -
Main script/procedure: FeaEdit()
Called scripts/procedures:
Tables, forms and reports: \$RADUNIT
Maintain radionuclide units table

Menu path: Report_Mgr\Prtnew -
Main script/procedure: RPTMGR() - source: UTIL\RPTMGR.SC
Called script depends on report selected in WAIT TABLE. See Report Manager technical notes in Appendix C, Section 5, for more information.
STORE_RPT() - source: UTIL\STORERPT.SC
TIMEDIFF() - source: UTIL\TIMEDIFF.SC
r\SWPMRPT.DB
Process, store, and print selected report
Others depend on report selection
Menu path: **Report_Mgr\New** -

Main script/procedure: 
Called scripts/procedures: 
Tables, forms and reports: 

Process and store selected report

Same as Prtnew
Same as Prtnew, except does not use STORE_RPT
Same as Prtnew

Menu path: **Report_Mgr\Reprint** -

Main script/procedure: 
Called scripts/procedures: 
Tables, forms and reports:

Store and print selected report

Same as Prtnew
Same as Prtnew

Menu path: **Report_Mgr\Store** -

Main script/procedure: 
Called scripts/procedures: 
Tables, forms and reports: 

Store selected report

Same as New
Same as Prtnew

Menu path: **Report_Mgr\Table** -

Main script/procedure: 
Called scripts/procedures: 
Tables, forms and reports: 

Return to report selection table

Same as Prtnew
Same as Prtnew

Process description:

Brings table r\SWPMRPT back into view for further report selection

Menu path: **Report_Mgr\Quit** -

Main script/procedure: 
Called scripts/procedures: 
Tables, forms and reports: 

Cancel report manager

Same as Prtnew

Process description:

Provides an obvious exit from the Report Manager, especially for users who call this procedure with a hotkey

Menu path: **Notes\Notepad**

Main script/procedure: 
Called scripts/procedures: 
Tables, forms and reports: 

View or edit general purpose notepad entries.

Genedit()

d\NOTEPAD
Menu path: **Notes\PCD_Oth**

Main script/procedure: Genedit()
Called scripts/procedures: d\PCDOTH
Tables, forms and reports:

View or edit "Other" PCD descriptive notes

Menu path: **Notes\Cont_Oth**

Main script/procedure: Genedit()
Called scripts/procedures: d\CONTOTH
Tables, forms and reports:

View or edit "Other" container type descriptive notes

Menu path: **Notes\HCD_Oth**

Main script/procedure: Genedit()
Called scripts/procedures: d\HCDOTH
Tables, forms and reports:

View or edit "Other" HCD descriptive notes

Menu path: **Notes\Box_Dim**

Main script/procedure: Genedit()
Called scripts/procedures: d\BOXDIM
Tables, forms and reports:

View or edit dimensions and descriptions for box-type containers

Menu path: **Admin\Statacc**

Main script/procedure: Statacc() - source: UTIL\STATACC.SC
Called scripts/procedures: d\DATACHG.DB - holds data change requests
d\SOFTCHG.db - holds software change requests
Tables, forms and reports:

Maintain status of change requests in support of configuration management

Menu path: **Admin\TabDoc**

Main script/procedure: Tabdoc() - source: UTIL\TABDOC.SC
Called scripts/procedures: doc\tables\TAB.DB - contains entries for each database table
doc\tables\TABFLD.DB - contains entries for all fields within each table
Tables, forms and reports:

Adds to or updates tables containing database documentation

5.8
Menu path: Admin\ValChk - Adds to or updates table documenting validity checks on database fields

Main script/procedure: Tabdoc() - source: UTIL\VALCHK.SC
Called scripts/procedures: doc\tables\TABFLD.DB - contains entries for all fields within each table

5.2 Menu System

To simplify maintenance of the application menus, most menu structures in Version 1.4 are defined by data in table MENU\MENUOPT. When this function operates, a procedure is executed that reads the table, finds the correct group of menu items, displays them in standard Paradox menu format and gets a selection from the user. The PAL command needed to perform the user’s choice is contained in a field of the menu table. The table also has an associated report that serves as a part of the system documentation.

5.2.1 PAL Scripts and Procedures

MENUEXE() - The procedure that accesses the menu table and displays user options in the familiar Paradox menu style

menu\MENU.SC - Source script for procedure MENUEXE()

5.2.2 Tables and Related Objects

menu\MENUOPT.DB - Table containing the data that define the application menu structure

menu\MENUOPT.R - Menu report for documentation

5.2.3 Adding an Item to a Menu

1. Determine which menu the item should appear within. See the SWPM Menu Structure in Figure 5.1.

2. Use a menu group name that matches usage for the other options on the same menu.

3. Insert a row in the menu table MENU\MENUOPT, defining each field according the following field descriptions:
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu Abbrev</td>
<td>The name of the group to which this item belongs. This controls which menu the item appears in. This field is linked to menu\MENUGRP and its contents must appear in that table.</td>
</tr>
<tr>
<td>Menu Opt #</td>
<td>The order in which this item appears on its menu.</td>
</tr>
<tr>
<td>Menu Opt</td>
<td>The name to appear on the menu.</td>
</tr>
<tr>
<td>Menu Desc</td>
<td>The description of the menu item to appear on the second line of the menu bar.</td>
</tr>
<tr>
<td>Menu Proc</td>
<td>The action to be taken when the item is selected. It can be a command, script name, or procedure.</td>
</tr>
<tr>
<td>Menu Report Order</td>
<td>The order in which this group appears on the report. Must be the same for all members of the group. This could be supplied by a field in menu\MENUGRP but was implemented this way prior to the creation of the lookup table.</td>
</tr>
</tbody>
</table>
6.0 Forecast Data Entry

Forecast data entry in SWPM is accomplished using the extended facilities provided by the Paradox Data Entry Toolkit (DET) and procedures written specifically for each table. General explanations of DET usage are beyond the scope of this document, but the process of conducting and controlling data entry as applied to SWPM is discussed in this section.

There are a number of objectives addressed in the way data entry procedures are designed and implemented in SWPM:

- Data integrity - volume, percentage and radionuclide data must be logically consistent with each other and internally consistent

- Standardization of terms, order of presentation, abbreviations, etc. - lists of each type of term (waste classes, waste generator names, physical content descriptors, etc.) are provided and their usage is enforced

- Ease of use - those responsible for data entry should not have to know the names of tables, scripts, procedures and other Paradox objects

- Ease of maintenance - an attempt was made to implement processes consistently and use nonspecific parameter-driven procedures where it was advantageous to do so.

6.1 Procedures and Scripts

Data entry procedures typically extract data for a specified waste generator from a permanent table, where data are kept in normal form, into a temporary table resembling a spreadsheet in format. For forecast volume data, the user is restricted to valid waste classes. For PCD and radionuclide data, the user may only enter data for waste classes that have non-zero volumes coming from the waste generator in question. When entering HCD and container data, the user may provide values only for waste class/PCD combinations for which volume data were previously entered. If percentage data are involved (PCD, HCD, and container), the requirement that data are entered as decimal fractions summing to 1 is enforced.

These procedures invoke DET facilities using the DO WAIT command.

Upon completion of data entry and successful validation of the data, any existing rows for the current waste generator/forecast combination are removed from the permanent table and replaced by data in the temporary table.
Data entry process procedure names and source scripts -

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Procedure</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forecast volumes</td>
<td>FOREVOL_DE()</td>
<td>pal\FOREVOL.SC</td>
</tr>
<tr>
<td>PCDs</td>
<td>PCD_DE()</td>
<td>pal\PCD.SC</td>
</tr>
<tr>
<td>HCDs</td>
<td>HCD_DE()</td>
<td>pal\HCD.SC</td>
</tr>
<tr>
<td>Containers</td>
<td>Cont_DE()</td>
<td>pal\CONT.SC</td>
</tr>
<tr>
<td>Radionuclides</td>
<td>Rad_DE()</td>
<td>pal\RAD.SC</td>
</tr>
</tbody>
</table>

Library: lib\FOREDE.LIB

Field arrival procedures - These procedures are invoked by the DET as the user moves to a new field in the data entry table. They generally serve to total up data for the current column or row and display the totals at the top of the screen. They also echo the current WG/Forecast combination, row label, and column name at the top of the screen, because this information at times scrolls off the screen.

Field arrival procedure names and source scripts -

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Procedure</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forecast volumes</td>
<td>ForeVol_NewFld()</td>
<td>pal\FOREVOL.SC</td>
</tr>
<tr>
<td>PCDs</td>
<td>PCD_NewFld()</td>
<td>pal\PCD.SC</td>
</tr>
<tr>
<td>HCDs</td>
<td>HCD_NewFld()</td>
<td>pal\HCD.SC</td>
</tr>
<tr>
<td>Containers</td>
<td>Cont_NewFld()</td>
<td>pal\CONT.SC</td>
</tr>
<tr>
<td>Radionuclides</td>
<td>Rad_NewFld()</td>
<td>pal\RAD.SC</td>
</tr>
</tbody>
</table>

Library: lib\FOREDE.LIB

Field departure procedure - This procedure is called when the user finishes entering data to a field and attempts to leave it. Its only action is to round the field to three decimal places. This procedure may also be used in other instances outside of DET-controlled processes where rounding is needed.

Field departure procedure name and source script -
To control data entry by the frevntReve

Keystroke procedure - This procedure is invoked for every keystroke during all forecast data entry processes performed under the control of the DET. It is used to block entry in certain columns and check for special keystrokes, such as the notepad hot key.

Keystroke procedure name and source script -

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Procedure</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>All forecast data</td>
<td>RndPer()</td>
<td>util\RNDPER.SC</td>
</tr>
</tbody>
</table>

Library: lib\UTIL.LIB

DET definition procedures - These procedures are written by the DET using information supplied interactively by the developer while using TKMenu. They provide all the definitions that the DET needs to control data entry. Included are the table name to which data will be entered and the names of various procedures to be invoked for arrival, departure, and keystroke events. They also map all keys to codes that indicate how each key can be used.

Warning: DET definition procedures are usually created with complete path names associated with tables and files referenced in the procedure. To maintain portability, the drive and application directory names should be removed from these paths before the procedures are compiled and written to the library file. The definition procedure scripts can be edited in the same way as any other PAL script.

DET definition procedure names and source scripts -

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Procedure</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forecast volumes</td>
<td>TKEForeVol_None()</td>
<td>work\EFOREVOL.SC</td>
</tr>
<tr>
<td>PCDs</td>
<td>TKEpcd_None()</td>
<td>work\EPCD.SC</td>
</tr>
<tr>
<td>HCDs</td>
<td>TKEhcd_None()</td>
<td>work\EHCD.SC</td>
</tr>
<tr>
<td>Containers</td>
<td>TKEcont_None()</td>
<td>work\ECONT.SC</td>
</tr>
<tr>
<td>Radionuclides</td>
<td>TKErad_None()</td>
<td>work\ERAD.SC</td>
</tr>
</tbody>
</table>

Library: lib\FOREDE.LIB

6.3
DET "borrow" files - These files are created by TKMenu and are referenced within the DET definition script created during the same process. They are used when the definition procedure is compiled but apparently not at other times. Even though a "borrow" file may exist for a given table, a new one is generated each time TKMenu is used. No manual maintenance of these tables is needed.

"Borrow" file names -

All tables: BORROW'n'.SC, where 'n' is a serial number one larger than used in existing borrow file names.

6.2 Table Names

- work\EFOREVOL.DB - Work table to which volume data are entered
- work\EPCD.DB - Work table to which PCD data are entered
- work\EHCD.DB - Work table to which HCD data are entered
- work\ECONT.DB - Work table to which container data are entered
- work\ERAD.DB - Work table to which radionuclide data are entered
- d\FOREVOL.DB - Permanent table containing volume data
- d\PCD.DB - Permanent table containing PCD data
- d\HCD.DB - Permanent table containing HCD data
- d\CONT.DB - Permanent table containing container data
- d\RAD.DB - Permanent table containing radionuclide data

6.3 Implementation

Complete instructions on DET usage, especially TKMenu, are left to the Paradox PAL manual. The following is an outline of steps necessary to add a DET process to SWPM. Maintenance of existing processes will necessarily involve skipping some of the steps listed.

1. Create the table to which data will be entered.

2. Develop new or identify existing procedures to be used for arrivals, departures, and keystrokes.

3. Develop the procedure that will be executed to perform the data entry process. This procedure must invoke the DET "Do Wait" command on the table that will receive the data entered.

4. Execute the TKMenu script in the DET and use the SetUpDoWait selection to define the process.
5. Modify the DET definition procedure script created in the previous step to remove complete path names that it contains. Path names used should be relative to the application directory (\SWPMDB92 in this case) rather than full paths as the DET defines them.

6. Compile the DET definition procedure and any procedures created in steps 2 and 3 above into a library that will be accessible by the application.

7. Test it.
7.0 Data Entry Notepad Functions

This feature allows a user to make notes into one of a set of database tables designed to capture notes for various purposes. These purposes include documenting non-routine changes to tables, unresolved problems, decisions affecting content of fields, the use of "OTHER" for PCD, HCD, or containers and to capture box dimensions when available. The notes are available to others via report selections on the Report Manager or by viewing the notepad tables. There are some differences in implementation because of requirements posed by various contexts. This function, therefore, actually includes a family of related scripts and procedures that appear similarly when used. The three implementations are

1. NOTEPAD0: General use, outside of SWPM application scripts or procedures.

2. NOTEPAD20: For use within a WAIT TABLE state that is established on a specific table within a PAL script or procedure. Captures general purpose notes only and stores them directly to table d\NOTEPAD.

3. NOTE_SET0, NOTES_A0 and NOTES_B0: To be executed in conjunction with data entry performed under the control of data entry toolkit procedures. Captures all types of notes. Holds them temporarily in arrays and stores them into tables at the end of the data entry session.

See the remainder of this section and Chapter 11 for more information about these procedures.

A typical notepad opens as a two-line window near the bottom of the screen and will accept up to 160 characters per note. It may be preceded by a menu, if the current context is such that more than one type of note may be appropriate. It automatically records some information with the exact elements also depending on context. The number of allowable notes is unlimited. However, during a given data entry session in the Data Entry Toolkit (DET) implementation, array space is available for only ten notes. A warning message results if an attempt is made to exceed the maximum number. In such a case, the user must exit and reenter the process. This forces the procedures involved to dump the notes to the notepad table and reinitialize the arrays that temporarily hold the notes.

7.1 Procedures and Scripts

NOTEPAD0 Procedure that invokes general purpose notepad, usable in routine interactive Paradox work where an application script does not have control. Adds notes directly to D\NOTEPAD table. Contained in LIB\UTIL.LIB.

NOTEPAD20 Procedure that performs WAIT TABLE version of notepad function. Contained in LIB\UTIL.LIB.
NOTE_SET()  Initializes arrays for DET notepad implementation. Used in conjunction with
NOTES_A() and NOTES_B(). Contained in LIB\FOREDE.LIB.

NOTES_A()  During DET operations where this procedure is referenced, notes and other data
are captured into arrays initialized by NOTE_SET(). Contained in
LIB\FOREDE.LIB.

NOTES_B()  Immediately after a data entry process using the DET, NOTE_SET() and
NOTES_A(), this procedure must be called to write data in the arrays to the
notepad tables. Contained in LIB\FOREDE.LIB.

NOTEPAD.SC  Located in UTIL subdirectory. Compiles into NOTEPAD().

NOTEPAD2.SC  Located in UTIL subdirectory. Compiles into NOTEPAD2(). Similar to
NOTEPAD.SC, except that the procedure created is intended to be called by
other procedures or scripts that have a table in WAIT TABLE status. Requires
procedures LU_sSAVE and LU_sRESTORE to be run before and after.

NOTE_SET.SC  Located in UTIL Subdirectory. Source script for NOTE_SET(). Compiled by
UTIL\FOREDE.SC.

NOTES_A.SC  Located in UTIL Subdirectory. Source script for NOTES_A(). Compiled by
UTIL\FOREDE.SC.

NOTES_B.SC  Located in UTIL Subdirectory. Source script for NOTES_B(). Compiled by
UTIL\FOREDE.SC.

7.2 Notepad Tables and Related Objects

BOXDIM.DB  Table containing dimensions and description of a box-type container, as well as
certain automatically captured contextual data.

BOXDIM.F  Form used for box dimensions notepad editing.

BOXDIM.R  Report specification used to print contents of box dimensions notepad.

CONTOTH.DB  Table containing additional information about instances of "OTHER" container
types, as well as certain automatically captured contextual data.

CONTOTH.F  Form used for "OTHER" container type notepad editing.

CONTOTH.R  Report specification used to print contents of "OTHER" container type notepad.
**7.3 Implementation**

**7.3.1 NOTEPA0** - this version can be executed as a miniscript or via hotkey. [Ctrl N] is recommended. Execute a script that includes the following line:

```
SETKEY 14 notepad()
```

**7.3.2 NOTEPA02** - execution should only be from another procedure. It requires the use of `LU_sSAVE()` and `LU_sRESTORE()` to save and then restore to the current state. Certain variables initialized by `LU_sSAVE()` are used to record the table and field names as well as the field contents in the notepad. The following example is a code fragment that would presumably be one of several CASE sections in a SWITCH structure:

```c
CASE retval = 14:  ; Invoked in this example by [Ctrl N]
    Loc_sSAVE()    ; call this first to save location in
                    ; table being edited
    Do_it!         ; ends EDIT or COEDIT
    notepad2()     ; executes procedure
    Loc_sRestore() ; restores position in original table
    Coeditkey      ; return to edit mode
```
7.3.3 *NOTE_SET0*, *NOTES_A0* and *NOTES_B0* - to be executed within a data entry process that uses the DET.

Invoke *NOTE_SET0* within the procedure, before the DO WAIT command occurs. This procedure creates arrays to temporarily store the note and other data to be captured in the notepad. The procedure allocates array space for as many as 10 notes.

Invoke *NOTES_B0* immediately after exiting the DO WAIT, in order to store any notes that were captured in arrays.

*NOTES_A0* is implemented indirectly. A keystroke level procedure must be defined in the DET setup for each table for which the notepad is to be used. The keystroke procedure (UTIL\KEYBLK.SC, for example) must then check for a particular keystroke ([Ctrl N], in this case) used to invoke the notepad. In addition, the script used to define entry, exit and keystroke procedures and usable keys must allow the keystroke. See WORK\EFOREVOL.SC as an example. The TKPosKey variable in such a script is used to define key availability.

7.4 **Automatically Captured Contextual Information**

7.4.1 Any use of the general purpose notepad causes these items to be recorded along with the 160 characters of note text:

- Current table name
- Current field name
- Current field contents
- Current date
- User ID of person making the note

7.4.2 When box dimensions notes are entered, the following information is also captured:

- Waste generator name
- Waste class name
- PCD type
- Container type

7.4.3 When notes are taken about "OTHER" container occurrences, the following information is also captured:

- Waste generator name
- Waste class name
- PCD type
- Current field value
7.4.4 When notes are taken about "OTHER" HCD occurrences, the following information is also captured:

- Waste generator name
- Waste class name
- PCD type
- Current field value

7.4.5 When notes are taken about "OTHER" PCD occurrences, the following information is also captured:

- Waste generator name
- Waste class name
- PCD type
8.0 Reporting Functions

Most report generation is accomplished using a feature of the SWPM database application called the Report Manager. The Report Manager displays a table of available reports and accepts the user’s selection as if the table were a menu. This feature is intended to eliminate the need to code report processing into every script that creates data for reporting and to make it simple to add a report selection to the application. Since project requirements include storage of report text as well as generation of hard copy, routing of report output to text files is also handled within this function. The Report Manager is intended as a generic tool to be usable in other Paradox applications. The method of defining a report in the report table was made fairly general but assumes a standard Paradox application directory structure. The structure needed is similar to that proposed by Brian J. Smith in his column titled "Up the Organization" published in the June 1990 issue of "Data Based Advisor" magazine. SWPM database applications for FY90, FY91, Reference Database and FY92 all conform to the model advanced by Smith. Certain directories have been added, however, to incorporate additional functions that Smith's structure did not include.

8.1 Descriptions and Locations of Files

Location of files (unless otherwise specified):

J:\SWPMDB92\R

8.1.1 PAL Scripts and Procedures

RPTMGR() Procedure that performs Report Manager function. Contained in LIB\UTIL.LIB.

STORE_RPT() Procedure that writes a report out to a text file. Located in LIB\UTIL.LIB. See Appendix E.

RPTMGR.SC Located in UTIL subdirectory. Contains PAL code that compiles into RPTMGR(). Allows selection of report to be run and/or printed via WAIT TABLE on table R\SWPMRPT.DB. See description of menu choices in Section 3.1 - Process Dictionary. This script also contains code for RPTMSG(), a dummy procedure used when no data manipulation processing is required before a report can be generated.

UTIL.SC Located in LIB subdirectory. Compiles all UTIL library procedures, including RPTMGR() into LIB\UTIL.LIB, entirely replacing the old version of the library.
8.1.2 Tables and Related Objects

REPORT.DB - Table containing information about all available SWPM database reports. RPTMGR() allows this table to be viewed and a report chosen under the control of WAIT TABLE. See Table 3.1 for information about the structure of this table.

REPORT.R - Report specification used to generate a report showing information about all available SWPM reports.

8.2 Adding a Report to the Report Manager

8.2.1 Defining a new Report Manager Record

Enter information in each field of a new report manager record according to the following field definitions:

<table>
<thead>
<tr>
<th>Column</th>
<th>Entry Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPT ID</td>
<td>Enter a unique ID for the report. This is the key to this table. The content</td>
</tr>
<tr>
<td></td>
<td>of this field is also used to define the name of the text file to which the</td>
</tr>
<tr>
<td></td>
<td>report will be written.</td>
</tr>
<tr>
<td>RPT ABBREV</td>
<td>The name of the table from which the report will be produced. Do not</td>
</tr>
<tr>
<td></td>
<td>include its path here.</td>
</tr>
<tr>
<td>REPORT NUMBER</td>
<td>The Paradox report number R, 1, 2, ..., 14 that indicates one of the 15 possible</td>
</tr>
<tr>
<td></td>
<td>reports defined for the table named in RPT ABBREV. The content of this field</td>
</tr>
<tr>
<td></td>
<td>relates to the file name extension of the file containing the report</td>
</tr>
<tr>
<td></td>
<td>specification. Extensions are &quot;.R&quot;, &quot;.RI&quot;, etc., preceded by the table name.</td>
</tr>
<tr>
<td>RPT NAME</td>
<td>The report title to be displayed on the Report Manager selection screen.</td>
</tr>
<tr>
<td></td>
<td>Should be all or most of the first line of the report title.</td>
</tr>
<tr>
<td>RPT SCRIPT</td>
<td>The script or procedure that will create the report data. Omit the extension</td>
</tr>
<tr>
<td></td>
<td>if it is a script. Include &quot;O&quot; after the name for procedures. Procedures</td>
</tr>
<tr>
<td></td>
<td>must be available for Paradox to find via AUTOLIB or other means. Use a &quot;do</td>
</tr>
<tr>
<td></td>
<td>nothing&quot; process here if no data preparation is required, such as &quot;RPTMSG()&quot;.</td>
</tr>
</tbody>
</table>
|                    | If a script is referenced, it must be in the directory defined by variable "scdir". This variable is defined in the Report Manager procedure and is set to "?PAL\" for this application.
8.2.2 Other Preparation

Unless otherwise noted, the following instructions assume that default file names and directory locations are used.

1. Create required queries and scripts. The initial script or procedure used to begin processing must be named to match the contents of field RPT SCRIPT. If no processing is needed, procedure "rptmsg0" can be referenced in the table. This avoids an error caused by a nonexistent procedure or script, but no processing is performed. Queries should be placed in the Q\ subdirectory, although no enforcement of query location is provided by the Report Manager.

2. If it does not already exist, create the table from which the report will be generated. If the table is created specifically to hold data extracted to produce a report, the table should be located in the WORK\ subdirectory. Using a table located anywhere else requires naming the directory location in the TABLE LOCATION field of the report table.
3. Create the report specification for the table defined in instruction 2 above. It is standard practice to include the report ID on the output, in the lower left corner of each page width. The source database for the report should also be displayed by placing a calculated field using variable 'rptsr'. Allow a calculated field width of 20 characters. The variable 'Rptsr' is set by the script PAL\STARTSWP.SC when the application is started.

8.2.3 Documenting a New Report

The PAL scripts used to create the report should be liberally commented. Place as much information about the report as possible in the Report Manager table. If the report process involves multiple steps, use "Reports Process Notes" forms to document each step. These notes are filed in the project notebook or project files.

8.3 Execution of the Report Manager

The selection Report_Mgr is part of the MAIN menu group in all versions of the application. See Chapter 5 for information about adding menu selections to the menu system. Lower level menu selections are coded within the Report Manager procedure and therefore do not appear in the menu system table.

After execution begins, the SWPM report table R\REPORT.DB is viewed in WAIT TABLE mode. The table used is defined as variable "rpttable" in the procedure script. The user makes a choice by pointing at the desired report and pressing the Do_it! key [F2]. A screen of information is then displayed that gives some indication how long the entire report process takes and how big the print file will be. Last run date is also shown.
9.0 Upload and Download Processing

Upload and download processing allows the movement of SWPM data between network disk drives and local disks in PCs connected to the network. The purpose of this facility is to allow users to copy a reference database to their own PCs to get the performance benefit of using a local hard disk. Also, changes can be made to a local copy of the data while preserving the reference database. Upload/download capability supports the practice of entering forecast data on a local machine and uploading batches of data to the network. When data are moved from local installations to the network, certain quality control procedures are applied before the data become part of or replace the existing network reference database.

In addition to the movement of forecast data, the download portion of this facility also allows users to copy the SWPM database application itself. The creation of an executable copy involves the movement of software and a number of tables of support data that are needed by the application. Support tables include the entire V\ subdirectory and other tables that support the menu system and the Report Manager. A complete download is done, therefore, in two steps: forecast data and software. It should be noted that a complete download does not create a stand-alone application on a local hard drive, since Paradox software is still required to support the database application.

9.1 Descriptions and Locations of Files

9.1.1 DOS Batch Files

SWPMUPLD.BAT - Upload process, see listing in Figure 9.1

The upload batch file is located in the application directory, \SWPMDB92, on the user's local drive. This file is installed on the local drive when the software download process is executed.

SWPMDNLD.BAT - Download process, see listing in Figure 9.2

The download batch file is located in the network application directory, \SWPMDB92.

SWPMLOG.BAT - Logs upload and download activities, see listing in Figure 9.3

The upload and download logging batch file is located in the network application utility directory, \SWPMDB92\UTIL.
9.1.2 Log Files

UPLOAD.LOG - Log of upload messages
DOWNLOAD.LOG - Log of download messages

These are text files containing messages generated during processing. Both are located in the D\ directory.

9.1.3 Quality Assurance Query Scripts

QA1.SC - Tests waste generator name usage
QA2.SC - Tests waste class usage
QA3.SC - Identifies new waste generators
QA4.SC - Identifies deleted waste generators

All QA scripts are located in the UPLOAD\ directory. See Section 9.4 for guidance on performing the QA tests.

9.2 Process Descriptions

9.2.1 Upload Processing

• The user executes this batch file with no parameters from the local SWPM application directory.

• An attempt is made to locate the logging batch file on the network. If this file cannot be found, the upload process assumes the network is not available. The process terminates with an error message.

• The beginning of the upload process is logged.

• The UPLOAD\ directory is checked. It must be free of all files before the process will proceed. If any files are detected, the occurrence is logged and the process ends with an error message.

• The DOS XCOPY utility is used to copy all files in the local D\ directory to the UPLOAD\ directory on the network.

• Any error messages that appeared are logged. Otherwise, an entry is made to log file noting a successful completion of the process.
9.2.2 Download Processing

- The user executes this batch file with one parameter, two parameters, or none from the network SWPM application directory. The first parameter is the target hard drive. Valid values are "C" or "D" with the default being the user's C drive. The second parameter is the type of download to perform. The value supplied can be "Data," "SW" (for software), or "All." The default download type is data only. Parameter validation is performed and then must be confirmed by the user before the process continues.

- The nature of the event is logged by redirecting a message to the log file. The time and date are also recorded by calling SWPMLOG.BAT.

- The selected download process is performed using the DOS XCOPY utility. Error status is checked after XCOPY finishes.

- Any error messages that appeared are logged. Otherwise, an entry is made to log file noting a successful completion of the process.

- If both types of download processes were selected, the second process executes in a manner similar to the first.

9.2.3 Upload and Download Logging

Some messages are simply added to the log file by echoing a message to the file using DOS redirection with append, i.e., the " >> " characters.

Time stamping log messages requires a more elaborate procedure:

- The upload or download batch file calls UTIL\SWPMLOG.BAT with two parameters: a label that serves as both a temporary file name and a line label in the log file, and the name of the log file to be written to.

- The logging batch file verifies that parameters were passed and then checks to make sure that it can create a file using the first parameter as a name and "_." as the extension.

- The file is created using the Echo command and redirection. As this file is created, DOS includes the current date and time in the directory.

- The DOS DIR command is used to get the directory entry and its output is filtered through the FIND command via DOS piping, so that only the just-created file will be reported out. This filtered directory output is redirected to the log file where it appears as a label that identifies the event followed by the current date and time.

9.3
• The temporary file created to capture the current system date and time is deleted.

9.3 Quality Control of Uploaded Data

It is assumed that data entry performed on a local copy of the database will be performed in accordance with the current User’s Guide, and that none of the data integrity checks will have been bypassed or defeated in any way. To avoid serious errors in reporting and analysis of forecast data and in model results, a few quality checks are made on uploaded data before the tables are copied into the D\ subdirectory. These checks consist of running queries located in the UPLOAD\ subdirectory and examining the Answer table that each creates. The descriptions below include the type of check being performed, the name of the query that performs the test, the desired result, and some possible implications when some other result occurs.

9.3.1 Verification of Waste Generator Abbreviation Usage

Query: UPLOAD\QC1.SC

Desired result: Empty ANSWER table

Possible problems: When one or more rows appear in the ANSWER table, there are invalid waste generators in the forecast volume table FOREVOL. The other tables containing the WG Abbrev field should also be considered suspect. The data integrity check that forces usage of waste generator abbreviations that exist in the WG table was violated, or the local WG table was changed after data entry occurred.

9.3.2 Verification of Waste Class Abbreviation Usage

Query: UPLOAD\QC2.SC

Desired result: Empty ANSWER table

Possible problems: When one or more rows appear in the ANSWER table, there are invalid waste classes appearing in the forecast volume table FOREVOL. The other tables containing the WCLASS Abbrev field should also be considered suspect. The data integrity check that forces usage of waste class abbreviations that exist in the $WCLASS table was violated, or the network version of the $WCLASS table is different than the local copy.

9.3.3 Detection of New Waste Generator Names

Query: Upload\QC3.SC

Desired result: Depends
Possible problems: New entries in the uploaded WG table are usually intended. Additions to the waste generator table will generally be made on the local database first and may not be added concurrently to the network WG table. This test serves to notify the database administrator of such occurrences in case additional verification is desired.

9.3.4 Detection of Existing Waste Generator Names Now Absent From Upload

Query: Upload\QC4.SC

Desired result: Depends

Possible problems: This check will reveal any waste generators that have been deleted from the local database. The deletion may be appropriate, but the database administrator should verify. This type of event should also have been recorded in the notepad.

9.4 Installation of Uploaded Data

After quality control checks are made, the uploaded data are ready to be placed in the D\ directory. The method used depends on how much concurrent activity takes place in both local and network databases. The following are basic guidelines.

9.4.1 Installation When All Data Entry is Local

Copy all files from the UPLOAD\ directory to the D\ directory. Alternately, a utility such as DIRMATCH or UPD can be used to identify the files that have changed and copy only the ones that have later time stamps on them. This activity is most efficiently performed in DOS.

9.4.2 Installation When Data Entry Conducted Concurrently

Concurrent entry of forecast data must be managed by distributing the work on some reasonable basis, such as by waste generator name. If the locally originating tables contain only data for certain waste generators, these tables must be added to network tables rather than copied over them. If the local user was given a full set of forecast generators and has responsibility of updating or adding only a subset, then the subset must be extracted from the uploaded tables and added to the network tables. This type of upload management must be conducted within Paradox, since selective queries and table adds are involved.
9.5 Preparation for the Next Upload

The upload processing batch file, SWPMUPLD.BAT, will not perform the upload unless the UPLOAD\ directory is empty. It is suggested, however, that the UPLOAD\ directory not be cleaned out until immediately before an upload is to take place. This practice will avoid uploads occurring without the knowledge of the database administrator. In addition, an occasional review of the upload log located in file D\UPLOAD.LOG, will help to verify that the upload process is being performed in accordance with procedures included in the User’s Guide.
@Echo off

:******************************************************************************
: Procedure name: SWPMUPLD.BAT
:
: Author: Cary Blackburn
:
: (c) 1993 by Pacific Northwest Laboratory
:
: Purpose: Uploads SWPM database files from a local disk to the
: network reference database.
:
: Parameters: None
:
: History:
:
: 10/08/92 - created by Cary Blackburn
:
:******************************************************************************
Cls
Echo.
Echo.
Echo.
Echo.
Echo.

SWPM Database v1.4 Upload Process
Echo.
Echo.
Echo Warning: This process should not be performed without
Echo notifying the database administrator first.
Echo If you wish to terminate, press [Ctrl] C.
Echo.
Echo.
Echo If you wish to terminate, press [Ctrl] C. Otherwise
Pause
:Step1

: | Step 1: Test for availability of network by checking for the |
: | presence of an application file on J: drive |
: |
: |

Figure 9.1 - Listing of SWPMUPLD.BAT
if exist J:\SWPMDB92\util\swpmlog.bat goto Step2
: Error detected - can't find network files so network may not be available
Echo.
Echo ^GNetwork not available - upload not performed.
Echo.
Goto END :

: Step2

| Step 2: First, write a message to the log file |
| Next call SWPMLOG.BAT with 2 parameters: |
| 1. Name of a temp file to create that will be given the |
| current date/time stamp |
| 2. Name of log file to receive the line of text that |
| contains the date/time stamp. |

Echo.

Echo Logging start of upload... .................................>>J:\swpmdb92\d\upload.log
Echo Upload process beginning...>>J:\swpmdb92\d\upload.log
call J:\swpmdb92\util\swpmlog_start J:\swpmdb92\d\upload.log
:

| Step 3: Test for readiness of upload area - directory must be |
| empty. |

if not exist J:\SWPMDB92\UPLOAD\*.* goto Step3
: Error detected - the upload directory is not empty
Echo.
Echo ^GUpload area not ready - upload not performed.
Echo.
Echo Logging attempted upload...
Echo Upload attempted without preparation >>J:\swpmdb92\d\upload.log
Goto END :

: Step4

| Step 4: Perform upload - content of D\ subdirectory is copied to |
| the upload area on the network |

Echo.

Figure 9.1. (contd)
Echo Uploading data..

Echo.

Xcopy D\*.* J:\swpmdb92\upload
if errorlevel 4 goto Step4er4
if errorlevel 2 goto Step4er2
if errorlevel 1 goto Step4er1
if errorlevel 0 goto Step5

:Step4er1

Echo.
Echo ^GWarning: No files found to copy - upload not performed.
Echo.
Echo Logging failed upload...
Echo Upload failed - no files found on local disk>>J:\swpmdb92\d\upload.log
goto END

:Step4er2

Echo.
Echo ^GProcess aborted by user - upload not completed.
Echo.
Echo Logging aborted upload...
Echo Upload aborted by user>>J:\swpmdb92\d\upload.log
goto END

:Step4er4

Echo.
Echo ^GOut of memory or disk space - upload not completed.
Echo.
Echo Logging failed upload...
Echo Upload incomplete - mem/disk limit or syntax>>J:\swpmdb92\d\upload.log
goto END

:Step5

: | Step 5: Successful completion - notify user and log it. |
: | |
: | |
|--------------------------|--------------------------|

Echo.
Echo Upload finished.
Echo.
Echo Logging completion of process...
Echo.
Echo Upload completed without error>>J:\swpmdb92\d\upload.log
call j:\swpmdb92\util\swpmlog _end J:\swpmdb92\d\upload.log
Echo ^G

:END

---

Figure 9.1. (contd)
@Echo off

Procedure name: SWPMONLD.BAT

Author: Cary Blackburn

(c) 1993 by Pacific Northwest Laboratory

Purpose: Downloads SWPM database application files to a local disk from the network reference database.

There are three modes of operation:

- Data - only tables from the D\ subdirectory are copied
- SW - only application software and support files are copied
- All - the entire application is copied to the local disk

Parameters:
1. Target disk - C or D (can also be specified as C: or D:)
2. Type of process - Data, SW or All

History:

10/09/92 - created by Cary Blackburn

Stepla - Attempt to validate parameters or establish defaults
IF %1x::=x Goto ALLDEF
IF %2x::=x Goto TYPDEF
goto :SETTYPE

:ALLDEF
Set _disk=C:
Set _proc=DATA
goto Step1

:TYPDEF
Set _proc=DATA
goto SETDISK

:SETTYPE
set _proc=%2
IF %_proc%::=ALL goto SETDISK
IF %_proc%::=DATA goto SETDISK
IF %_proc%::=SW goto SETDISK
goto BADTYPE

Figure 9.2 - Listing of SWPMONLD.BAT
:SETDISK
set _disk=%1
IF %_disk%%==C: goto Stepl
IF %_disk%%==D: goto Stepl
IF %_disk%%==C goto SETDISKC
IF %_disk%%==D goto SETDISKD
goto BADDISK
:
:SETDISKC
set _disk=C:
goto Stepl
:
:SETDISKD
set _disk=D:
:
:Step1

| Step 1: Provide warning about file overwrites - give the user a chance to chicken out. If the process continues, log the start of it.

Cls
Echo.
Echo.
Echo.
Echo.
Echo SWPM Database v1.4 Download Process
Echo Process: %_proc%
Echo Target drive: %_disk%
Echo Warning: This process will overwrite existing files.
Echo If you wish to terminate, press [Ctrl] C.
Echo.
Echo Otherwise -
Pause
Echo.
Echo Logging start of download...
Echo ------------------------------>>J:\swpmdb92\d\download.log
Echo Download of %_proc% to %_disk% beginning...>>J:\swpmdb92\d\download.log

Figure 9.2. (contd)
Next call SWPMLOG.BAT with 2 parameters:

1. Name of a temp file to create that will be given the current date/time stamp
2. Name of log file to receive the line of text that contains the date/time stamp.

call J:\swpmdb92\util\swpmlog_start J:\swpmdb92\d\download.log

:Step2

Step 2: Data download - performed if process specified is Data or All

IF %_proc%==SW goto Step3
Echo.
Echo Downloading data...
Echo.
xcopy J:\swpmdb92\d\*. * % disk%\swpmdb92\d\*. *
del % disk%\swpmdb92\d\*. Log>nul
if errorlevel 4 goto Step2er4
if errorlevel 2 goto Step2er2
if errorlevel 1 goto Step2er1
if errorlevel 0 goto Step2a
:Step2er1
Echo.
Echo ^GWarning: No files found to copy - download not performed.
Echo.
Echo Logging failed attempt to download...
Echo Download failed - no files found on local disk>>J:\swpmdb92\d\download.log
goto END
:Step2er2
Echo.
Echo ^GProcess aborted by user - download not completed.
Echo.
Echo Logging aborted download...
Echo Download aborted by user>>J:\swpmdb92\d\download.log
goto END
:Step2er4
Echo.
Echo ^GOut of memory or disk space - download not completed.
Echo.

Figure 9.2. (contd)
Echo Logging aborted download...
Echo Download incomplete - mem/disk limit or syntax
Goto END:

:Step2a

Echo.
Echo Data download complete, logging...
Echo Download of D\ files to %_disk% completed.

:Step3

| Step 3: Software download - performed if process specified is SW or All |

IF %_proc%==DATA goto Step4

Echo.
Echo Downloading software...
Echo.
xcopy J:\swpmdb92\lib*.* %_disk%\swpmdb92\lib*.*
xcopy J:\swpmdb92\menu*.* %_disk%\swpmdb92\menu*.*
xcopy J:\swpmdb92\mvd*.* %_disk%\swpmdb92\mvd*.*
xcopy J:\swpmdb92\pal*.* %_disk%\swpmdb92\pal*.*
xcopy J:\swpmdb92\q*.* %_disk%\swpmdb92\q*.*
xcopy J:\swpmdb92\r\report*.* %_disk%\swpmdb92\r\report*.*
xcopy J:\swpmdb92\statacc*.* %_disk%\swpmdb92\statacc*.*
xcopy J:\swpmdb92\tools*.* %_disk%\swpmdb92\tools*.* /s
xcopy J:\swpmdb92\util\*.* %_disk%\swpmdb92\util\*.*
del %_disk%\swpmdnld.bat>nul
del %_disk%\swpmdlog.bat>nul
xcopy J:\swpmdb92\v*.* %_disk%\swpmdb92\v*.*
xcopy J:\swpmdb92\work*.* %_disk%\swpmdb92\work*.*
xcopy J:\swpmdb92\*. sc %_disk%\swpmdb92\*. sc
if errorlevel 4 goto Step3er4
if errorlevel 2 goto Step3er2
if errorlevel 0 goto Step3a

:Step3er2

Echo.
Echo ^GProcess aborted by user - download not completed.
Echo.
Echo Logging aborted download...
Echo Download aborted by user

Figure 9.2. (contd)
goto END
:Step3er4
Echo.
Echo ^GOut of memory or disk space - download not completed.
Echo.
Echo Logging failed download...
Echo Download incomplete - mem/disk limit or syntax

\[\text{goto J:\swpmdb92\d\download.log}\]

\[\text{goto END}\]

:Step3a
Echo.
Echo SW download complete. Logging...
Echo Download of SW files to _disk_ completed.
\[\text{call J:\swpmdb92\util\swpmlog \_swend J:\swpmdb92\d\download.log}\]

:Step4
| Step 4: Successful completion - notify user and log it. |
| : | |
Echo.
Echo Logging completion of process...
Echo.
Echo Download completed without error.
\[\text{goto J:\swpmdb92\d\download.log}\]
Echo.
Echo ^GDownload finished.
\[\text{goto END}\]

:BADTYPE
Echo.
Echo ^GError: Invalid download process type specified.
Echo Valid values are DATA, SW or ALL.
Echo.
\[\text{goto END}\]

:BADDISK
Echo.
Echo ^GError: Invalid target disk drive specified.
Echo Valid values are D or C.
Echo.
\[\text{:END}\]
set _disk=
set _proc=

Figure 9.2. (contd)
Procedure name: SWPMLOG.BAT

Author: Cary Blackburn

(c) 1993 by Pacific Northwest Laboratory

Purpose: This is a routine to be called during the upload and download process to capture a date/time stamp. It does so rather crudely by creating a temporary file and capturing its directory entry in a way that serves to indicate when an event happens.

This process will not operate unless two parameters are passed to it.

Parameters: 1. Name to use for the temporary file to create. If this file exists, the log entry will not include a date/time stamp because doing so would wipe out the existing file. This file must be specified without an extension. An extension of "._" will be added here.

2. The name of the log file to write to. If this is not an existing file, then an error message is displayed and no attempt is made to log anything.

History:

10/08/92 - created by Cary Blackburn

Figure 9.3. Listing of SWPMLOG.BAT
Step2 - verify that file name passed as 1st parameter does not exist with an extension of "_."

If exist %1. goto BADPARM1

Step3 - verify that log file name passed as 2nd parameter does exist.

If not exist %2 goto NOPARM2

Step4 - All tests passed - make log entry
First, write nonsense text to the temp file named by parameter 1.
Then, capture what the dir command says about the file, including the date/time stamp.

Echo.>%1.
dir *.|find "%1">%2
goto END
:BADPARM1
Echo.
Echo ^GSWPMLOG Error: 1st parameter names existing file, cannot log event.
goto END :
:NOPARM1
Echo.
Echo ^GSWPMLOG Error: 1st parameter must be specified as a valid file name :
:NOPARM2
Echo.
Echo ^GSWPMLOG Error: 2nd parameter must be the name of the log file
Echo.
Echo Event not logged due to one or more errors.
goto END :
:BADPARM3
Echo ^GSWPMLOG Error: Too many parameters used in call to SWPMLOG.
:END
if exist %1. del %1.>nul

Figure 9.3. (contd)
10.0 Reference Data Library Generator

This process produces files for input to the SWPM. It is written entirely in PAL and can be executed from the SWPM database application menu. In Version 1.4, it includes scenario management features that allow the modeler to identify sources for various inputs to the RDL process and define a destination for output.

There are actually three phases to RDL usage. The first, described in Section 10.1, involves interaction with the user to define the source locations and destination for the scenario of interest. The next phase, covered in Section 10.2, is optional and serves to regenerate forecast data in the form needed by the RDL process itself. The final phase, in Section 10.3, is the RDL generator process itself, in which model input files are created.

10.1 Scenario Management

This module is entirely new for Version 1.4 and is yet to be coded. A section will be added when construction is complete and the module has been tested.

10.2 Forecast Data Preparation

This module will be similar to the reference database version but is yet to be coded. A section will be added to this document when construction is complete and the module has been tested.

10.3 RDL File Generation Process

This section will be added when preparation of the RDL software is complete.

10.4 RDL File Generator Operation

This section will be added when preparation of the RDL software is complete.
11.0 Procedure Libraries

This chapter describes the individual procedures contained in libraries used by the SWPM application. The six procedure libraries are ADMIN, DETOOLS, FEAEDIT, FOREDE, RDL, and UTIL. All libraries are located in the LIB\ directory in files with ".LIB" as the file name extension. Use of these libraries involves the standard Paradox practice of including their names in the definition for the system variable AUTOLIB. This definition is supplied automatically during SWPM startup processing as described in Chapter 4.

11.1 ADMIN Library

This library contains procedures used by the database administrator and developer. The procedures are not intended to be used by the general database user. See Chapter 12, Sections 12.3 and 12.4 for more information.

11.2 DETOOLS Library

This library contains some of the DET procedures provided by Paradox. The library has been copied to the application library directory for convenience. Individual procedures are documented in the DET section of the Paradox PAL manual.

11.3 FEAEDIT Library

This library contains edit procedures used for maintaining tables containing codes and parameters, such as the waste generator and forecast tables.

FeaEdit() Description: Provides view and edit facility for table whose name is passed as a parameter. User's rights are checked and special handling is given to operations involving creation, deletion, and changing of key field via other procedures called by this procedure.

Parameters:
1. Type: A
   Descriptive label of table to be shown as part of a Wait Table prompt

2. Type: A
   Name of table to be edited
3. Type: A
   Key field of table

Return: None

Application: SWPM Codes table maintenance

Scripts -
Source: PAL\FEAEDIT.SC
Compile: LIB\FEAEDIT.SC

FeaIns() Description: Called by FeaEdit() to handle inserts to the specified table. The user is prompted for a new item to add to the table. User's response becomes key to new table row. A check is made for a duplicate. If insert is successful, the cursor is positioned so that the other columns of the new row may be defined.

Parameters: 1. Type: A
   Descriptive label of table to be shown as part of prompt for new feature

2. Type: A
   Name of table to be edited

3. Type: A
   Key field of table

Return: None

Application: SWPM Codes table maintenance

Scripts -
Source: PAL\FEAEDIT.SC
Compile: LIB\FEAEDIT.SC

FeaUpd() Description: Called by FeaEdit() to handle updates of the key field in the specified table. The user must move to the row to be modified and is prompted for a new feature. A check is made for a duplicate. If the new key definition does not duplicate some existing one, the change is made. The procedure maintains referential integrity by changing other tables using the key value that was modified here.
Parameters: 1. Type: A  
   Name of table to be edited

2. Type: A  
   Key field of table

Return: None

Application: SWPM Codes table maintenance

Scripts -  
   Source: PAL\FEAEDIT.SC  
   Compile: LIB\FEAEDIT.SC

FeaDel()  
Description: Called by FeaEdit() to handle the deletion of a row in the specified table. The user must move to the row to be deleted first. Checks are made for rows in other tables. If the key of the row to be deleted is also a key in certain other tables, the delete is not allowed.

Parameters: 1. Type: A  
   Name of table to be edited

2. Type: A  
   Key field of table

Return: None

Application: SWPM Codes table maintenance

Scripts -  
   Source: PAL\FEAEDIT.SC  
   Compile: LIB\FEAEDIT.SC

11.4 FOREDE Library

This library contains data entry procedures that are executed directly or invoked in response to events by the DET.

Cont_DE()  
Description: Extracts container data (if any) from d\CONT for the current WG/Scenario combination into a work table for use as a data entry/update area. Data entry toolkit procedures are called.
Certain data validation checks are made before the user may exit. Data from this work table replace all data for the WG/Scenario in the original table.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
Source: PAL\CONT.SC
Compile: LIB\FOREDE.SC

Cont_NewFld()
Description: Data entry toolkit procedure invoked whenever the cursor is moved to a new field during data entry. Maintains current row totals, prompts and sets blocking status for fields which cannot accept data.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
Source: PAL\CONT.SC
Compile: LIB\FOREDE.SC

Denorm()
Description: Normalizes data entry tables in a process that is the inverse of crosstabbing. The receiving table is first purged of all rows containing data for the current WG name and scenario. This process is used to return data to the original table after data entry or update is performed in a work table.

Parameters:
1. Type: A
   The name of the table to be normalized

2. Type: A
   The name of an existing normal form table to receive the data
3. Type: N
   The number of "key" fields, i.e., those fields that define the rows

   Return: None
   Application: SWPM data entry

   Scripts -
   Source: PAL\FOREDE.SC
   Compile: LIB\FOREDE.SC

   De_Rep_Ins()
   Description: Builds an insert query to add data for the current WG to table work\DE_REP for reporting purposes.
   Parameters: 1. Type: A
                The WG to be used as selection criterion in the insert query
   Return: None
   Application: called by FORE_DE_REP()

   Scripts -
   Source: PAL\FOREDE.SC
   Compile: LIB\FOREDE.SC

   Forede()
   Description: Establishes screen messages and invokes menu procedure to get a data entry process selection from the user.
   Parameters: None
   Return: None
   Application: SWPM data entry

   Scripts -
   Source: PAL\FOREDE.SC
   Compile: LIB\FOREDE.SC

   Foreedit()
   Description: Gets a WG/Scenario selection from the user prior to data entry or verification reporting. This is a mandatory step that defines WG and scenario variables that are used by other processes.
Even though the procedure name and code comments imply otherwise, no editing is done within this procedure.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
Source: PAL\FOREEDIT.SC
Compile: LIB\FOREDE.SC

Foreval() Description: Verifies the validity of a WG/Scenario combination by looking for it in the d\FORECAST table.

Parameters: None

Return: "True" if valid, "False" if combination does not exist

Application: SWPM data entry

Scripts -
Source: PAL\FOREDE.SC
Compile: LIB\FOREDE.SC

Forevol_De() Description: Extracts volume data (if any) from d\FOREVOL for the current WG/Scenario combination into a work table for use as a data entry/update area. Data entry toolkit procedures are called. Certain data validation checks are made before the user may exit. Data are returned to the original table.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
Source: PAL\FOREVOL.SC
Compile: LIB\FOREDE.SC

11.6
Forevol_NewFld()  Description: DET procedure invoked whenever the cursor is moved to a new field during data entry. Maintains current row totals, prompts and sets blocking status for fields that accept data.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
Source: PAL\FOREVOL.SC
Compile: LIB\FOREDE.SC

Fore_De_Rep()  Description: Empties and refills forecast report table with current data by calling DE_REP_INS. Sends a report to the printer.

Parameters: None

Return: None

Application: ?

Scripts -
Source: PAL\FOREDE.SC
Compile: LIB\FOREDE.SC

Get_Period_Data()  Description: Executes a query to retrieve data for a specified WG, scenario, and forecast period.

Parameters: 1. Type: A
 Table to run query against

2. Type: A
 WG name

3. Type: A
 Scenario name

4. Type: A
 Period (year, "Held," etc)
Return: Number of records in answer table

Application: Data entry processes

Scripts -
  Source: PAL\FOREDE.SC
  Compile: LIB\FOREDE.SC

HCD_De() Description: Extracts HCD data (if any) from d\HCD for the current WG/Scenario combination into a work table for use as a data entry/update area. DET procedures are called. Certain data validation checks are made before the user may exit. Data are returned to the original table.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
  Source: PAL\HCD.SC
  Compile: LIB\FOREDE.SC

HCD_NewFld() Description: DET procedure invoked whenever the cursor is moved to a new field during data entry. Maintains current row totals, prompts and sets blocking status for fields that cannot accept data.

Parameters: None

Return: None

Application: SWPM data entry

Scripts -
  Source: PAL\HCD.SC
  Compile: LIB\FOREDE.SC

Notes_A() Description: Invoked (commonly by ALT N) to capture notes for storage in d\NOTEPAD. This half of the process maintains the notes in an array that must be written to the notepad data table by NOTES_B. Immediate storage to the table was not possible
since this procedure is called during DET processes. Updating another table while using DET is not worth the coding effort and loss of performance. NOTES_SET() must be used first. See implementation notes in source code.

Parameters: None
Return: None
Application: SWPM data entry

Scripts -
Source: UTIL\NOTES_A.SC
Compile: LIB\FOREDE.SC

Notes_B()
Description: This procedure must be called after exiting from a DET process to store any notes made during the process via NOTES_A. Implementation notes in source code for NOTES_A include instructions on use of NOTES_B.

Parameters: None
Return: None
Application: SWPM data entry

Scripts -
Source: UTIL\NOTES_B.SC
Compile: LIB\FOREDE.SC

Notes_Set()
Description: Establishes arrays for use by NOTES_A and NOTES_B in capturing notepad entries during DET processes. See implementation notes in NOTES_A source code.

Parameters: None
Return: None
Application: SWPM data entry

11.9
Scripts -
Source: UTIL\NOTES_SET.SC
Compile: LIB\FOREDE.SC

PCD_De()
Description: Extracts container data (if any) from d\PCD for the current WG/Scenario combination into a work table for use as a data entry/update area. Data entry toolkit procedures are called. Certain data validation checks are made before the user may exit. Data are returned to the original table.

Parameters: None
Return: None
Application: SWPM data entry

PCD_NewFld()
Description: DET procedure invoked whenever the cursor is moved to a new field during data entry. Maintains current row totals, prompts and sets blocking status for fields that cannot accept data.

Parameters: None
Return: None
Application: SWPM data entry

Rad_De()
Description: Extracts radionuclide data (if any) from d\RAD for the current WG/Scenario combination into a work table for use as a data entry/update area. DET procedures are called. Certain data validation checks are made before the user may exit. Data are returned to the original table.

Parameters: None
Return: None
Application: SWPM data entry

Scripts -
  Source: PAL\RAD.SC
  Compile: LIB\FOREDE.SC

Tkecont_None()  Description: DET procedure that defines arrival, departure and keystroke procedures for container data entry. It also establishes which keys are valid and invalid during data entry.
Parameters: None
Return: None
Application: SWPM data entry

Scripts -
  Source: WORK\ECONT.SC
  Compile: LIB\FOREDE.SC

Tkeforevol_None()  Description: DET procedure that defines arrival, departure, and keystroke procedures for volume data entry. It also establishes which keys are valid and invalid during data entry.
Parameters: None
Return: None
Application: SWPM data entry

Scripts -
  Source: WORK\FOREVOL.SC
  Compile: LIB\FOREDE.SC

TkeHCD_None()  Description: DET procedure that defines arrival, departure, and keystroke procedures for HCD data entry. It also establishes which keys are valid and invalid during data entry.
Parameters: None
Return: None
Application: SWPM data entry

Scripts -
Source: WORK\EHCD.SC
Compile: LIB\FOREDE.SC

TkePCD_None() Description: DET procedure that defines arrival, departure, and keystroke procedures for PCD data entry. It also establishes which keys are valid and invalid during data entry.
Parameters: None
Return: None
Application: SWPM data entry

Scripts -
Source: WORK\EPCD.SC
Compile: LIB\FOREDE.SC

11.5 Reference Data Library

This library contains procedures that support generation of reference data library (RDL) files for input to the model. See Chapter 10 for a description of the RDL generator software.

WriteRDL() Description: Initiates RDL process, calls WRITERPS() and WRITERPF().
Parameters: None
Return: None
Application: SWPM RDL

Scripts -
Source: PAL\WRITERDL.SC
Compile: LIB\RDL.SC
WriteRPF()  
Description: Compiles RPF data and exports to a text file.

Parameters:  
1. Type: A
   Name of file to export to

Return: None

Application: SWPM RDL

Scripts -  
Source: PAL\WRITERPF.SC
Compile: LIB\RDL.SC

WriteRPS()  
Description: Compiles RPS data and exports to a text file.

Parameters:  
1. Type: A
   Name of file to export to

Return: None

Application: SWPM RDL

Scripts -  
Source: PAL\WRITERPS.SC
Compile: LIB\RDL.SC

11.6 UTIL Library

This library contains a set of miscellaneous procedures used throughout the SWPM database application.

Beeps()  
Description: Beeps three times. Used to notify user of end of a process.

Parameters: None

Return: None

Application: Various procedures and scripts

Scripts -  
Source: UTIL\BEEPS.SC
Compile: LIB\UTIL.SC

11.13
BlnkZero()  

**Description:** Converts zeros or small values to blanks in numeric columns of a specified table. Used for improved report appearance. If a small non-zero value is to be used as the threshold value for conversion to blank, it cannot exceed 0.1.

**Parameters:**

1. **Type: A**  
   Table name - including directory path

2. **Type: N**  
   The first column number to be inspected for the threshold value and changed to blank if it is less than or equal the value of parameter 4.

3. **Type: N**  
   The number of columns to be inspected for the threshold value. If the range of columns defined by parameters 2 and 3 includes columns that are not of "N" types, these columns are not changed. They are counted as part of the number of columns to be examined for update, however. The procedure automatically compensates if this number is too large. In situations where all columns after the one specified as parameter 2 are to be processed, a large number (e.g., 100) may be used for convenience.

4. **Type: N**  
   The threshold value at or below which a conversion to blank is made. This allows a very small non-zero number to be blanked out as if it were zero. A maximum of 0.1 is enforced in the procedure, however.

**Return:** True if no problems were encountered. Returns False if table does not exist or starting column number is too large. On-screen error messages are also displayed.

**Application:** Various report process scripts

**Scripts -**

- **Source:** UTIL\BLNKZERO.SC
- **Compile:** UTIL\UTIL.SC

11.14
DupeRec() Description: Duplicate an entire record in edit or coedit mode.
Parameters: None
Return: None
Application: Available for connection to user-defined hot key
Scripts -
Source: UTIL\DUPEREC.SC
Compile: LIB\UTIL.SC

ExportSS() Description: Export a Paradox table to a Quattro or Lotus spreadsheet.
Parameters: None
Return: None
Application: All X'nnn'.SC scripts - 'nnn' indicates the ID number
Scripts -
Source: UTIL\EXPORTSS.SC
Compile: LIB\UTIL.SC

Fill_Tbl() Description: Inserts normalized data into spreadsheet style format data entry tables. Similar to cross-tabulation operation.
Parameters: 1. Type: A
   The first field of the source table that will be the key or part of the key for the destination table. Additional key fields in the source must appear immediately following the first key. The column identifier and table value must appear following the row key(s) in the source.

   2. Type: N
   The number of fields in the source table that uniquely defines records in the destination tables. These fields must appear consecutively after the field named in parameter 1. Acceptable values are 1 or 2 only.

   3. A comma delimited list of column names that are valid for data entry. An asterisk means all columns are valid.
4. The name of the spreadsheet style data entry table to receive the data.

Return: None

Applications: Most SWPM data entry processes

Scripts -
Source: UTIL\FILLTBL.SC
Compile: LIB\UTIL.SC

Genedit()
Description: Generic table edit facility.

Parameters:
1. Type: A
   The name of the table to be edited
2. Type: A
   Title to show on display
3. Form number
   Must be a valid form or null placeholder. A valid form designation ("F" or 1-14) displays the table in form view mode.
4. Notepad2 call enable/disable (Passed as True/False)

Return: True if user exits with F2. Returns False if user exits with Esc

Applications: General table editing, such as NOTEPAD, BOXDIM, and other notepad tables

Scripts -
Source: UTIL\GENEDIT.SC
Compile: LIB\UTIL.SC

Getstr_Lu()
Description: Accepts a string from the user and checks for its presence as a first field value in a lookup table.

Parameters:
1. Type: A
   Text to be used as a prompt

11.16
2. Type: A
   Data type of value to be accepted

3. Type: A
   Name of lookup table

Return: Value of string entered, of type specified by parameter 2

Applications:

Scripts -
Source: UTIL\GETSTR.SC
Compile: LIB\UTIL.SC

Getstr_YN()  Description: Accepts a string from the user that must be either "Y" or "N". Input is automatically shifted to upper case.

Parameters: 1. Type: A
   Text to be used as a prompt

Return: Either a "Y" or "N", depending on user response.

Applications: SWPM get a string application

Scripts -
Source: UTIL\GETSTR.SC
Compile: LIB\UTIL.SC

KeyBlk()  Description: Blocks data entry into a field of a table used with the DET. To implement, specify this procedure as a keystroke procedure for each field on which it is to be applied. Set the variable ZZDataOK to FALSE in the calling procedure to block entry.

Parameters: None, although this procedure checks the value of global variable ZZDataOK

Return: Sets TKAccept to True or False

Applications: SWPM data entry processes that use the DET
Scripts
Source: KEYBLK.SC
Compile: LIB\UTIL.SC

LineCtr()
Description: Displays centered text on the screen line specified. Text is truncated if longer than 78 characters. Out-of-range line specification causes function to use line 11 of the display.
Parameters: 1. Type: A, up to 78 characters
Text to be displayed
2. Type: N
Screen line number (0-23)
Applications: Various processes that display messages to the user

Scripts
Source: LINECTR.SC
Compile: LIB\UTIL.SC

Loc_Restore()
Description: Restore the workspace location to a previously saved position. LOC_SAVE() must have been used to record data about location.
Parameters: None
Return: None
Applications: Lookup()

Scripts
Source: UTIL\LOOKUP.SC
Compile: LIB\UTIL.SC

Loc_Save()
Description: Save the current location relative to the workspace. Assumes recovery will be made with LOC_RESTORE. Allows for multiple linked tables as well as a single table.
Parameters: None
Return: None - defines variables

11.18
Loc_sSave()  Description: Save the current location relative to the workspace. Assumes recovery will be made with LOC_sRESTORE. Works for a single table in workspace. Also assigns certain variables that Notepad2() uses to record the current table name, field name, and field content.

Parameters: None

Return: None - defines variables

Applications: Genedit(), Feaedit(), and ForeEdit()

Loc_sRestore()  Description: Restore the workspace location to a previously saved position. LOC_sSAVE() must have been used to record data about location.

Parameters: None

Return: None

Applications: Genedit(), Feaedit(), and ForeEdit()

Loc_sSave()  Description: Save the current location relative to the workspace. Assumes recovery will be made with LOC_sRESTORE. Works for a single table in workspace. Also assigns certain variables that Notepad2() uses to record the current table name, field name, and field content.

Parameters: None

Return: None - defines variables

Applications: Genedit(), Feaedit(), and ForeEdit()

Loc_sRestore()  Description: Restore the workspace location to a previously saved position. LOC_sSAVE() must have been used to record data about location.

Parameters: None

Return: None

Applications: Genedit(), Feaedit(), and ForeEdit()
MaxLen()  
Description: To specify the maximum string length of contents of the current field. This procedure is useful for planning report field specifications.

Parameters: None
Return: None
Application: Ad hoc data manipulation

MenuExe()  
Description: Provides a data-driven menu facility that allows menu content to be entirely determined by the contents of a table. See Menu System, Chapter 11, for additional documentation.

Parameters:  
1. Type: A
   Menu option name - the text appearing in the MENUOPT field of menu\MENU. This parameter establishes the group of options that are to appear as a single Paradox-style bar menu.

2. Type: A
   The first title line of the menu screen - appears in central area of the screen
3. Type: A
   The second title line of the menu screen

4. Type: A
   The third (caption) line of the menu screen.

5. Type: logical
   Determines whether border is shown - "False" for no border displayed

Return: None

Application: SWPM database application

Scripts -
Source: MENU\MENU.SC
Compile: LIB\UTIL.SC

Mov_Flds()
Description: Copies the contents of a specified number of fields from the source table to the destination table. Each record in the source table provides data to be copied to the destination.

Parameters:
1. Type: A
   Name of the field in the current table in the workspace that will be the first of a specified number of fields in each row to be transferred

2. Type: A
   Name of destination table

3. Type: A
   Name of first field in destination table that will receive data transferred from source. This table must also be in the workspace.

4. Type: N
   Number of consecutive fields to be transferred from source table rows to destination table rows

Return: None

Applications: Most data entry processes

11.21
Scripts
Source: UTIL\MOVFLD.SC
Compile: LIB\UTIL.SC

Msg()
Description: Displays a message at lower right of screen followed by instruction to press ENTER to continue.
Parameters: 1. Type: A
Text of message
Return: Message text in lower right of screen, followed by prompt to press ENTER key and a pause induced by an ACCEPT statement. Current colors are overridden with white on red background.
Applications: Various processes that pass messages to the user and require acknowledgement before continuing

Scripts -
Source: UTIL\MSG.SC
Compile: LIB\UTIL.SC

MsgCtr()
Description: Displays a message at the center of the screen. Unlike MSG(), the process that calls this procedure continues immediately with no user intervention needed. It uses the colors currently in effect.
Parameters: 1. Type: A
Text of message
Return: Message text in center of screen
Applications: Various processes that pass information messages to the user

Scripts -
Source: UTIL\MSG.SC
Compile: LIB\UTIL.SC

Normal()
Description: Normalizes table specified as source table into table named as destination table, allowing for number of key fields in source table.

11.22
Parameters: 1. Type: A
   Name of source table

2. Type: A
   Name of destination table

3. Type: N
   Number of key fields in source table

Return: None

Applications: Ad hoc data manipulation

Scripts -
Source: UTIL\NORMAL.SC
Compile: LIB\UTIL.SC

NotePad()
Description: General purpose note-taking utility that captures information about the current table and field. Not intended to be used within WAIT TABLE function. See Notepad Functions, Chapter 10, for additional documentation.

Parameters: None

Return: None

Application: Can be invoked as a miniscript or connected to a hotkey via SETKEY. Key 14 [Ctrl N] is recommended.

Scripts -
Source: UTIL\NOTEPAD.SC
Compile: LIB\UTIL.SC

NotePad2()
Description: Similar to NOTEPAD() except that it is intended to be used within a WAIT TABLE section of another procedure or script. See Notepad Functions, Chapter 10, for documentation. Also see source code comments for implementation information.

Parameters: None

Return: None
Application: Available from within certain SWPM table maintenance processes that do not use the DET, such as Genedit()

Scripts -
Source: UTIL\NOTEPAD2.SC
Compile: LIB\UTIL.SC

Proper() Description: Converts the enclosed string to proper name capitalization. The first letter of each word is capitalized and remaining letters are lower case. The user is responsible for passing data of string type.

Parameters: 1. Type: A
       Text string to be capitalized as a proper name

Return: Text string

Application: Ad hoc data manipulation

Scripts -
Source: UTIL\PROPER.SC
Compile: LIB\UTIL.SC

RndPer() Description: Rounds percentages to three decimal places. Call as a departure procedure for a field or to update the current field. Uses DET features.

Parameters: None - works on current field

Return: Replaces current field with rounded data

Applications: Data entry processes

Scripts -
Source: UTIL\RNDPER.SC
Compile: LIB\UTIL.SC


Parameters: None
Return: None
Applications: Invoked from SWPM database application menu

Scripts -
   Source: UTIL\RPTMGR.SC
   Compile: LIB\UTIL.SC

RptMsg()
Description: Specified as the procedure to be executed by the Report Manager when no actual processing is required to prepare report data. A message is displayed stating that no processing is required and printing or storage of the report text proceeds immediately.
Parameters: None
Return: None
Applications: Report Manager

Scripts -
   Source: UTIL\RPTMGR.SC
   Compile: LIB\UTIL.SC

RptNA()
Description: Invoked by Report Manager when report processing is chosen for a report that is not yet available. Provides an informational message and avoids aborting the Report Manager when the report has been specified and added to the report table but not yet programmed.
Parameters: None
Return: None
Applications: Report Manager

Scripts -
   Source: UTIL\RPTMGR.SC
   Compile: LIB\UTIL.SC

StatAcc()
Description: Provides update and report capability for SWPM database Version 1.4 configuration management.
StatAcc2()  
Description: Allows editing of software change request or data change request tables.
Parameters: 1. Type: A  
            Name of table to be edited in this procedure
Return: None
Applications: Statacc()

Script -  
Source: UTIL\STATACC2.SC  
Compile: LIB\UTIL.SC

StatAcc3()  
Description: Reporting procedure for status accounting forms.
Parameters: 1. Type: A  
            Name of table to be edited in this procedure
Return: None
Applications: Statacc()

Script -  
Source: UTIL\STATACC3.SC  
Compile: LIB\UTIL.SC

Store_Rpt()  
Description: Prints Paradox report to text file.
Parameters: 1. Type: A  
            Name of table to which the report is related

11.26
2. Type: A
Report number of report desired. Values "R" and "R1" through "R14" are valid.

3. Type: A
Output file to receive report. Extension must be omitted. Paradox default ".RPT" is used. Existing file by this name is written over without warning.

Return: None

Applications: Rptmgr()

Scripts -
Source: UTIL\STORERPT.SC
Compile: LIB\UTIL.SC

TabDoc()
Description: Adds table and field names to tables in which these are documented. See Technical Notes, Chapter 12, for documentation of this feature.

Parameters: None

Return: None

Applications: Used in documentation phase only

Scripts -
Source: UTIL\TABDOC.SC
Compile: LIB\UTIL.SC

TimeDiff()
Description: Calculates difference between times to show elapsed time. Return value format selectable by mode parameter. If either store or end time is null or blank, returned value is "99:99:99" or "999.9", depending on format mode. If mode is invalid, "0" is returned.

Parameters:
1. Type: A
   Starting time of timed period

2. Type: A
   Ending time of timed period
3. Type: N
   Format mode selection. Set to 1 if elapsed time is to be
   returned in HH:MM:SS format. Set to 2 if MMM.M format
   is desired.

Return: Difference between times as string in format selected by third
         parameter. A return of 0 indicates an invalid mode was sent. A
         return value of all nines indicates a missing start or end time.

Applications: Rptmgr()

Trims leading and trailing blanks from a string passed as a
parameter, returns trimmed value. Imbedded blanks are not
removed.

Parameters: 1. Type: A
             Character string to be trimmed

Return: Character string received, less any trailing or leading blanks

Applications: General purpose tool

ValChk() Description: Validity check information is captured and stored in table/field
documentation table "TABFLD".

Parameters: None

Return: None

Applications: SWPM database administration

Scripts -
Source: UTIL\TIMEDIFF.SC
Compile: LIB\UTIL.SC

Scripts -
Source: UTIL\VALCHK.SC
Compile: LIB\UTIL.SC

11.28
ValExist()  Description: Determines whether there are any occurrences of a specified value in a particular column of a table.

Parameters:
1. Type: A
   Name of table to search
2. Type: A
   Field of table to be searched
3. Type: same as type of field specified by parameter 2

Applications: Codes table maintenance processes
Return: True or False

Scripts -
Source: UTIL\VALEXIST.SC
Compile: LIB\UTIL.SC

ValPdDel()  Description: Deletes all rows of a table with a Period ID field that contains a specified value.

Parameters:
1. Type: A
   Name of table in which delete is to take place
2. Type: A
   Dummy parameter
3. Type: A
   Period ID of rows to be deleted

Return: None

Applications: Some SWPM codes table maintenance processes

Scripts -
Source: UTIL\VALPDDEL.SC
Compile: LIB\UTIL.SC

ValUpd()  Description: Updates all rows in a designated table containing a specified field value to a new value. A "changeto" query is executed.
Parameters: 1. Type: A  
   Name of table in which update is to take place

2. Type: A  
   Name of field to change

3. Type: A  
   Old value of field

4. Type: A  
   New value of field

Return: None

Applications: Codes table maintenance

Scripts
Source: UTIL\VALUPD.SC
Compile: LIB\UTIL.SC

ValYrDel() Description: Deletes rows from a specified table in which other parameters match certain field values. Uses "delete" query.

Parameters: 1. Type: A  
   Name of field in which deletes are to take place

2. Type: A  
   Name of field containing WG names

3-5. Dummy parameters

6. Type: depends on field type of parameter 2  
   Value contained in field specified by parameter 2

7. Type: depends on field to the right of that specified in parameter 2  
   Value of scenario contained in this field

8. Type: A  
   Year value of row(s) to be deleted
9. Type: A
   Period ID of row(s) to be deleted

Return: None

Applications: Codes table maintenance

Scripts -
Source: UTIL\VALRDEL.SC
Compile: LIB\UTIL.SC
12.0 Development and Administration Tools

This chapter describes various tools provided in the SWPM database for both administration and development needs. Included with each description are details on the procedures, scripts, and implementation of that tool.

12.1 Status Accounting

Under sound configuration management practices, requested changes of data and software must be approved in accordance with the configuration management plan. Approved changes should be tracked and their current status should be easily determinable. This subsystem provides a place to receive data describing approved changes and supports the determination of the status of any change through reports or displays on a screen. It also serves as a history of changes to the system made since the configuration management baseline was established.

Data and software change requests are treated separately for maintenance purposes, although a combined status report is available. A single method of supplying change ID numbers is used for both data and software changes.

12.1.1 Procedures and Scripts

Statacc() Procedure that invokes status accounting function. Access privileges are verified and the menu is provided to allow the selection of maintenance screens for either software or data change requests and for reports. Contained in LIB\UTIL.LIB.

Statacc2() Called by Statacc() - provides display and update screen for either data or software change requests.

Statacc3() Called by Statacc() - allows user to select a particular data or software request to print

STATACC.SC Located in UTIL subdirectory - compiles into Statacc()

STATACC2.SC Located in UTIL subdirectory - compiles into Statacc2()

STATACC3.SC Located in UTIL subdirectory - compiles into Statacc3()

12.1.2 Tables and Related Objects

d\DATACHG.DB Table containing data change request records
d\DATACHG.F  Form used for display and maintenance of DATACHG table
work\DATACHG.DB Table used to receive a single change request record to be printed
work\DATACHG.R Report specification used to print contents of work\DATACHG table
d\SOFTCHG.DB Table containing software change request records
d\SOFTCHG.F  Form used for display and maintenance of SOFTCHG table
work\SOFTCHG.DB Table used to receive a single change request record to be printed
work\SOFTCHG.R Report specification used to print contents of work\SOFTCHG table

12.1.3 Implementation

This feature is implemented by adding menu selections in the menuing system as described in Chapter 5. There are additional menus generated within the procedures.

12.1.4 Operation

All requested changes to software should be entered into the status accounting system. If data changes involve forecast information, no entries are needed during the data entry phase of the forecast data collection cycle. Later, after forecast volumes have been published, resulting in a de facto baseline, data changes should be recorded here. Data changes that involve lookup tables should always be recorded. This type of change will often be accompanied by software changes. In such cases, the data and software change request records should reference each other's change ID numbers.

12.1.5 Maintenance

Barring changes to requirements, the only maintenance needed is to change the starting change ID number in util\STATACC.SC. This number starts the sequence of automatically assigned IDs and should include the fiscal year number ("92" for Version 1.4) as the first two digits.

12.2 Table/Field Documenter

This set of procedures captures basic table structure and validity check information and adds it to documentation tables from which the data dictionary is produced. Space in the documentation tables is provided so that descriptions of each table and field can be entered manually. When minor changes such as expansion of a data field or a data type is changed, it is usually easier to manually update the documentation tables.
12.2.1 Procedures and Scripts

Tabdoc() Procedure that gets user's selection of a table to add to the documentation. A warning is issued if the table has already been documented. The procedure uses the D\ directory as the default but may be easily overridden. Contained in LIB\UTIL\LIB.

Valchk() Procedure that examines each documented table for field validity checks. These checks are recorded and are available for reporting. Contained in LIB\UTIL\LIB.

TABDOC.SC Located in UTIL subdirectory - compiles into Tabdoc()

VALCHK.SC Located in UTIL subdirectory - compiles into Valchk()

12.2.2 Tables and Related Objects

doc\tables\TAB.DB Table containing names and descriptions of application tables

doc\tables\TABFLD.DB Table containing names, descriptions, and validity check information for each field of each table in the database

work\DOCTMP.DB Temporary table used by VALCHK() to create a copy of doc\tables\TABFLD.DB. Validity check determination must be done on a copy to avoid conflicts when examining validity checks on the same table that receives validity check documentation.

12.2.3 Implementation

This feature is executed via the menuing system described in Chapter 5. Implementation requires installation of the tables in directory DOC\TABLES within the application directory. Procedures must be included in a library available to the user. Report "R" for table DOC\TABLES\TABFLD contains a report title that should be changed to reflect the application in which the documenter feature is installed. Otherwise, this feature is generic and needs no other customization.

12.2.4 Operation

Select Admin from the SWPM application menu. Then choose TabDoc or ValChk from the administration menu. Validity check documentation requires no further action. For table documentation, each table must be individually selected. The user is asked to approve overwrites of documentation for tables that have been previously included.

12.3
12.2.5 Maintenance

No routine maintenance requirements exist.

12.3 Report Development Tools

A number of the tasks required to prepare reports of the type used frequently in the SWPM database application are highly repetitious and can involve several steps each time they are performed. For example, defining subtotal and total fields for a report covering the thirty-year forecast period will entail the creation of 60, 90 or more summary fields, all of which may also have to be formatted. The report development tools detailed in this section address this problem and a few others by way of menu selections. Figures 12.1 and 12.2 show these menus as they appear over the Paradox report definition screen. These tools are intended to be generic and therefore usable for other applications besides SWPM.

12.3.1 Menu Options

Group Totals

The report is scanned for group footers. Within each footer discovered, a summary field is defined for every numeric field in the table band, using the same field format as used for the detail values. An underline is inserted above each group total. Group footers must be empty prior to using this option. The procedure automatically inserts additional lines into the group footer band when needed. To complete, line labels such as "Group Total" must be added manually to each page width. The group footer should be examined for meaningless totals, such as an ID number that receives a total along with other values in the detail. These items must be removed manually.

Report Totals

Report totals fields are placed in the report footer for every numeric field in the table band. Single underlines are placed above totals and double underlines appear below. The report band must be empty or this option will terminate with an error message. A line label such as "Grand Total" must be inserted manually in each page width. As with group totals, the report footer should be examined for meaningless report totals.

Page Setup

Another menu is displayed, as shown in Figure 12.2. There are six page layout options and two page duplexing choices. Selection 1 resets the report to the defaults, which are assumed to be in accordance with those described in Section 2.3, Paradox Default Settings. Selections 1 through 6 assume an HP LaserJet printer is connected to LPT1 or available via that port through the...
network. Duplexing functions chosen using selections 7 and 8 require that the HP printer is a model IIIsi. A number of individual setup parameters are established in response to selections 1-6, any one of which can be changed afterward, if further refinements are needed. Selections 7 and 8 only add to the existing printer setup string and do not change existing settings. Duplexing selections are relative to the short and long edges of the paper without regard to orientation. As a rule of thumb for duplexing, select the edge of the paper that will be punched or bound.

Show Report Parameters -

The current report settings are displayed as in Figure 12.3. No changes can be made with this menu selection.

12.3.2 Procedures and Scripts

RptBandName() Returns the name of the current report band
RptCurRight() Moves the cursor on the PAL canvas and returns the exact report column position
RptFindTbl() Finds the table band, positions the cursor in it and returns the row number
RptFindFld() Finds the next numeric field on the current line by searching for the character "9"
RptGetFmt() Records the format of the numeric field, so total fields will be created similarly
RptTools() Creates and displays report tools menu by using Popup2() procedure
RptGrpTtot() Creates group total fields within every group footer; adds lines in footers as needed
RptRptTtot() Creates report total fields; adds lines to report footer if needed
RptSetup() Creates and displays report setup menu using Popup2() procedure
RptSpecs() Retrieves and displays current report specification parameters
Popup2() DET procedure that displays pop up lists to be used as menus

All of the above procedures are contained in LIB\ADMIN.LIB.
RTOOLSRC.SC  Contains PAL code for the first six procedures listed above
GRPTOT.SC  Contains PAL code for procedure RptGrpTot()
RPTTOT.SC  Contains PAL code for procedure RptRptTot()
RPTSETUP.SC  Contains PAL code for procedure RptSetup()
RPTSPECS.SC  Contains PAL code for procedure RptSpecs()
POPUP2.SC  Contains PAL code for DET procedure popup2()

All of the above script files are contained in the TOOLS subdirectory

12.3.3 Implementation

The library containing the procedures must be included in the definition of the AUTOLIB variable. The initial procedure, RptTools(), is most conveniently used when associated with a hot key such as [Shift-F1] using a PAL setkey script such as:

Setkey -84 Rpttools()

12.4 Password Protection

Password protection is implemented on key database tables. Protection is installed using a set of procedures that are executed from the PAL menu that is accessed via [Alt-F10] outside of any application.

12.4.1 Levels of Security

Three levels are established, with a single password to be shared by all users assigned to each level. The levels are:

Master - for administrator use - includes all privileges - implemented via a Paradox master password

Intermediate - for modelers and data analysts - implemented via a Paradox auxiliary password definition

General user - for data entry, general purpose viewing, running reports - implemented by way of a Paradox auxiliary password definition

12.6
12.4.2 Procedures and Scripts

Pswds() Establishes passwords to be used, calls Pswds_2() and Pswds_3() to define privileges for each password in each of the protected tables

Pswds_2() Called by Pswds() to establish master password for a table

Pswds_3() Called one or more times for each table by Pswds() to establish privileges for one or both auxiliary passwords

All of the above procedures are contained in LIB\ADMIN.LIB.

PSWDS.SC Contains PAL code for procedure Pswds()

PSWDS_2.SC Contains PAL code for procedure Pswds_2()

PSWDS_3.SC Contains PAL code for procedure Pswds_3()

All of the above script files are located in the UTIL subdirectory

12.4.3 Implementation

No special implementation actions are recommended, since these procedures are not used often. The library containing the procedures must be included in the definition of the AUTOLIB variable. The Pswds() procedure is then executed as a miniscript from the PAL menu. Changes to script file PSWDS.SC must be done via Paradox Scripts Editor Edit menu selections, since the script itself is encrypted via the master password.

12.5 Miscellaneous Tools

These software tools are used to conveniently perform some action that often must be done repetitively. See the source scripts for additional documentation stored as comments. Hot key usage shown below is suggested only, the tools may be tied to any hot keys that do not conflict with other Paradox usage.

Maxlen() Returns maximum length of character data in the current column of a table - executed as a miniscript

Duperec() Duplicates the current record immediately below its position in the table - not to be used for keyed tables - executed as a miniscript or by a hot key established with the SET KEY command
Cuttext() Records text or contents of a field for later pasting in variable XXXTEMP - companion procedure to Pasetxt() - executed by connecting to a hot key via SETKEY command - [Ctrl-C] recommended, i.e., SETKEY 3 Cuttext()

Pasetxt() Pastes stored text or contents of a field previously saved with CUTTEXT() to variable XXXTEMP - replaces existing text or field contents - executed by connecting to a hot key via SETKEY command - [Ctrl-P] recommended, i.e., SETKEY 16 Cuttext()

All of the above procedures are contained in LIB\ADMIN.LIB.

MAXLEN.SC Located in UTIL subdirectory - compiles into Maxlen()

DUPREC.SC Located in UTIL subdirectory - compiles into Duprec()

CUTTEXT.SC Located in UTIL subdirectory - compiles into Cuttext()

PASTETXT.SC Located in UTIL subdirectory - compiles into Pasetxt()

CALCSUM.SC Located in TOOLS subdirectory - enters "CALCSUM AS 'fieldnam'" in a query field and moves the cursor one field to the right - useful when creating a group summary queries for 30-year forecast tables where all 30 years must be

CALCSUMed - executed by connecting to a hotkey via SETKEY command - [Ctrl-N] recommended, i.e., SETKEY 14 play "tools\calcsum"
REPORT DEVELOPMENT TOOLS

| Group Totals | Report Totals | Page Setup | Show Report Parameters | Quit |

Figure 12.1 Report Development Tools Menu

SELECT PAGE SETUP


Figure 12.2 Page Setup Menu

REPORT SPECIFICATIONS

Setup String: \027E\027&110\027(s\16.66H
Printer Port: LPT1
Report Format: TableOfGroups
Group Repeats: Retain
Wait Between Pages: No
Margin: 0
Page Length: 45
Page Width: 178
Number of Page Widths: 1

Figure 12.3 Report Parameters Screen
13.0 Maintaining Treatment, Storage, and Disposal Data

This chapter provides guidance in entering and maintaining treatment, storage, and disposal (TSD) data. This activity is the responsibility of PNL staff who are involved with model operation. Changes to these tables are therefore not made as a part of routine forecast data maintenance. These data describe the details of the solid waste management system to the SWPM. Unlike forecast information, TSD data do not originate with the waste generators. Designers of systems for the treatment, storage, and ultimate disposal of waste develop TSD data to guide the model as it simulates the flow of wastes through the waste management system. Forecast data and TSD data are, therefore, two distinct parts of the SWPM database and are generally developed and maintained by different people. Forecast information is viewed as an externally imposed "given," while TSD data can be modified experimentally in the course of searching for waste management solutions. In practice, the TSD database tables are infrequently modified, but serve as a baseline from which copies are made to develop treatment scenarios. Instructions for TSD scenario creation and usage are available in separate documents for those users who are involved in this activity.

Procedures for entering TSD data are very similar to those used for forecast data: menu options are chosen to select a particular table to work in, and editing procedures conform to standard Paradox usage.

Four menu options provide access to tables that contain TSD descriptive data:

**OPER**
This table lists the available operations to be performed on waste passing through the system. Attribute fields in this table include cost information and waste classes to which the operations apply.

**DIST_F**
This table contains the distribution fractions that are used to apportion the waste from each waste generator to the various operations. The table contains the waste generator abbreviated name, the fraction itself, the operation receiving the fraction of waste, and other information.

**OPER_FLOW**
This table describes the flow of waste from one operation to another. It includes the names of the current and next operations, the fraction of waste to go to the next operation, and other information.

**OPERCAP**
This table lists the capacities of each operation in the system. It includes the name of the operation, the name of the applicable waste class, the year for which the capacity applies, and the capacity itself.

Two other menu options involve tables containing information used during the production of Reference Data Library files used for model input:
WG_TSD This table contains a list of waste generators to be recognized by the system. It should contain every TSD waste generator name referenced in the forecast waste generator table, plus specially grouped waste generator names to be used when creating the VOL_TSD table.

WSTSTRM This table contains the combinations of waste classes and container types that make up waste streams that will be passed to the model. The contents of this table are used for constructing a query rather than as data directly.

13.1 The Oper Table

- Select the TSD option from the Main Menu.

- Select Oper from the TSD Data Entry Menu.

- The table containing operation data appears. Use the [Ins] key to add new items and standard Paradox editing procedures for updating existing ones.

- Press [F2] to finish. Since editing work is performed in the Paradox COEDIT mode, all changes are posted as soon as work on a row is complete. Thus, pressing [Esc] will not cancel inserts, deletes, or updates made during an editing session.

13.2 The Dist_F Table

- Select the TSD option from the Main Menu.

- Select Dist_F from the TSD Data Entry Menu.

- The table containing distribution fraction data appears. Use the [Ins] key to add new items and standard Paradox editing procedures for updating existing ones.

- Press [F2] to finish. Since editing work is performed in the Paradox COEDIT mode, all changes are posted as soon as work on a row is complete. Thus, pressing [Esc] will not cancel inserts, deletes, or updates made during an editing session.

13.3 The Operflow Table

- Select the TSD option from the Main Menu.

- Select OperFlow from the TSD Data Entry Menu.
• The table containing operation flow data appears. Use the [Ins] key to add new items and standard Paradox editing procedures for updating existing ones.

• Press [F2] to finish. Since editing work is performed in the Paradox COEDIT mode, all changes are posted as soon as work on a row is complete. Thus, pressing [Esc] will not cancel inserts, deletes, or updates made during an editing session.

13.4 The Opercap Table

• Select the TSD option from the Main Menu.

• Select OperCap from the TSD Data Entry Menu.

• The table containing operation capacity data appears. Use the [Ins] key to add new items and standard Paradox editing procedures for updating existing ones.

• Press [F2] to finish. Since editing work is performed in the Paradox COEDIT mode, all changes are posted as soon as work on a row is complete. Thus, pressing [Esc] will not cancel inserts, deletes, or updates made during an editing session.

13.5 The WG_TSD Table

• Select the TSD option from the Main Menu.

• Select WG_TSD from the TSD Data Entry Menu.

• The table containing TSD waste generator names appears. Use the [Ins] key to add new items and standard Paradox editing procedures for updating existing ones.

• Press [F2] to finish. Since editing work is performed in the Paradox COEDIT mode, all changes are posted as soon as work on a row is complete. Thus, pressing [Esc] will not cancel inserts, deletes, or updates made during an editing session.

13.6 The WSTSTRM Table

• Select the TSD option from the Main Menu.

• Select WstStrm from the TSD Data Entry Menu.

• The table containing waste stream definition data appears. Use the [Ins] key to add new items and standard Paradox editing procedures for updating existing ones.
Fields in this table are:

Waste Class - Corresponds to the Waste Class Abbrev field in forecast data

Container - A standard container type or valid Paradox pattern - may include a logical "not"

Waste Stream - The name to be assigned to any waste volume that matches the waste class and container types specified in the two fields above

- Press [F2] to finish. Since editing work is performed in the Paradox COEDIT mode, all changes are posted as soon as work on a row is complete. Thus, pressing [Esc] will not cancel inserts, deletes, or updates made during an editing session.
<table>
<thead>
<tr>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OFFSITE</strong></td>
</tr>
<tr>
<td>2 Office of Scientific and Technical Information</td>
</tr>
<tr>
<td><strong>OFFSITE</strong></td>
</tr>
<tr>
<td>3 Westinghouse Hanford Company</td>
</tr>
<tr>
<td>A. O. Vance</td>
</tr>
<tr>
<td>O. J. Valero</td>
</tr>
<tr>
<td>K. L. Hladek</td>
</tr>
<tr>
<td><strong>ONSITE</strong></td>
</tr>
<tr>
<td>14 Pacific Northwest Laboratory</td>
</tr>
<tr>
<td>L. L. Armacost (5)</td>
</tr>
<tr>
<td>J. A. Jamison</td>
</tr>
<tr>
<td>C. L. Blackburn</td>
</tr>
<tr>
<td>G. M. Holter</td>
</tr>
<tr>
<td>Publishing Coordination</td>
</tr>
<tr>
<td>Technical Report Files (5)</td>
</tr>
</tbody>
</table>

Distr. 1
DATE
FILMED
12/8/93

END