DISTRIBUTION SHEET

To CDMS Team	From E.L. Gray	Page 1 of 1
Project Title/Work Order CDMS System Design Description		Date 10/25/94
		EDT No. N/A
		ECN No. 606990

Name	MSIN	Text With All Attach.	Text Only	Attach./ Appendix Only	EDT/ECN Only
Leroy S. Legowik	L4-89	X			
Timothy O. Haggett	H5-14	X			
Terri M. Lutter	G6-14	X			
Robert L. Martell	S2-01	X			
Ronald O. Shelby	H5-14	X			
D.M. Dell	L4-89	X			
Bary D. Elison	HQ-41	X			
Donald M. Brown	HQ-41	X			
Randall M. Yeats	H5-14	X			
Edward R. Carroll	A3-74	X			
E.L. (Buzz) Gray	A3-74	X			
James A. Warren	A1-04	X			
Central Files	L8-04	X			

RECEIVED
NOV 23 1994
OSTI

OSTI

LS-07 (2)

COMPLETE

ENGINEERING CHANGE NOTICE

Page 1 of 2

1. ECN 606990

Proj.
ECN

2. ECN Category (mark one) Supplemental ☐ Direct Revision ☒ Change ECN ☐ Temporary ☐ Standby ☐ Supersedure ☐ Cancel/Void ☐		3. Originator's Name, Organization, MSIN, and Telephone No. E. L. Gray, 62420, B5-26, 376-1434		4. Date 09/30/94	
		5. Project Title/No./Work Order No. CDMS System Design Description	6. Bldg./Sys./Fac. No. N/A	7. Approval Designator Q	
		8. Document Numbers Changed by this ECN (includes sheet no. and rev.) WHC-SD-GN-SDD-30007, REV. 0	9. Related ECN No(s). N/A	10. Related PO No. N/A	
11a. Modification Work ☐ Yes (fill out Blk. 11b) ☒ No (NA Blks. 11b, 11c, 11d)		11b. Work Package No. N/A	11c. Modification Work Complete N/A _____ Cog. Engineer Signature & Date	11d. Restored to Original Condition (Temp. or Standby ECN only) N/A _____ Cog. Engineer Signature & Date	
12. Description of Change The CDMS System Design Description is being changed in it's entirety to allow for the following: Design change for Graphical User Interface Change in graphic representations Modification of screen menus and data entry screens					
13a. Justification (mark one) Criteria Change ☒ Design Improvement ☐ Environmental ☐ As-Found ☐ Facilitate Const. ☐ Const. Error/Omission ☐ Design Error/Omission ☐					
13b. Justification Details This release will describe the software product in the 'As-built' state after applying modifications outlined in the Section 12. Description of Change, above. This was necessary to enhance and clarify user interface by utilizing Visual Basic software, not included in the initial release of this document.					
14. Distribution (include name, MSIN, and no. of copies) See attached Distribution Sheet				RELEASE STAMP OFFICIAL RELEASE BY WHC DATE OCT 25 1994 <i>Sta. 21</i>	

RELEASE AUTHORIZATION

Document Number: WHC-SD-GN-SDD-30007, Rev. 1

Document Title: CAD Data Set Management Syetem (CDMS) System Design Description

Release Date: 10/31/94

* * * * *

This document was reviewed following the
procedures described in WHC-CM-3-4 and is:

APPROVED FOR PUBLIC RELEASE

* * * * *

WHC Information Release Administration Specialist:

C. Willingham
C. WILLINGHAM

10/31/94

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

SUPPORTING DOCUMENT

1. Total Pages 111

2. Title

CDMS - CAD Data Set System Design Description

3. Number

WHC-SD-GN-SDD-30007

4. Rev No.

1

5. Key Words

CDMS
CAD
DMCS
Design Description
SDRC

6. Author

Name: E. L. Gray

Signature

Organization/Charge Code 62400/R40760

APPROVED FOR
PUBLIC RELEASE
OW 10-21-94

7. Abstract

This document will define the functional and technical design description required to modify a commercial software product that manages and controls engineering drawings (CDMS), to interface with the final plotting of CAD Data Sets

8. PURPOSE AND USE OF DOCUMENT - This document was prepared for use within the U.S. Department of Energy and its contractors. It is to be used only to perform, direct, or integrate work under U.S. Department of Energy contracts. This document is not approved for public release until reviewed.

PATENT STATUS - This document copy, since it is transmitted in advance of patent clearance, is made available in confidence solely for use in performance of work under contracts with the U.S. Department of Energy. This document is not to be published nor its contents otherwise disseminated or used for purposes other than specified above, before patent approval for such release or use has been secured, upon request, from the Patent Counsel, U.S. Department of Energy Field Office, Richland, WA.

DISCLAIMER - This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

10.

RELEASE STAMP

OFFICIAL RELEASE
BY WHC
DATE OCT 25 1994

Sta. 21

9. Impact Level 20

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

RECORD OF REVISION

(1) Document Number

WHC-SD GN-SDD-
30007

Page 1

(2) Title

CAD Data Set Management System (CDMS) System Design Description

CHANGE CONTROL RECORD

(3) Revision

(4) Description of Change - Replace, Add, and Delete Pages

Authorized for Release

(5) Cog. Engr.

(6) Cog. Mgr. Date

(7) WHC-SD-GN-SDD-30007, Rev 0, EDT 160646 04/06/94

E.L. Gray

E.R. Carroll 09/30/94

1 RS

Incorporate the description of change in ECN-606990 dated 9/30/94

2294

Paul

**CAD Data Set Management System
(CDMS)**

SYSTEM DESIGN DESCRIPTION

WHC-SD-GN-SDD-30007, Rev. 1

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

js

TABLE OF CONTENTS

1.0	INTRODUCTION	1
1.1	PURPOSE	1
1.2	SCOPE	1
1.3	OVERVIEW	1
1.4	DEFINITIONS AND ACRONYMS	3
2.0	DECOMPOSITION DESCRIPTION	5
2.1	MODULE DECOMPOSITION	5
2.1.1	CDMS Logical Data Model	5
2.1.2	CDMS Logical Process Model	7
3.0	DEPENDENCY DESCRIPTION	12
3.1	INTERMODULE DEPENDENCIES	12
3.2	INTERPROCESS DEPENDENCIES	12
3.3	DATA DEPENDENCIES	12
3.4	CONVERSION DEPENDENCIES	12
4.0	INTERFACE DESCRIPTION	13
4.1	MODULE INTERFACES	13
4.2	PROCESS INTERFACES	13
4.2.1	File Transfers	13
4.2.2	Final Plot	13
4.2.3	CDMS PCI Interface	14
5.0	DETAILED DESIGN	15
5.1	CDMS MENUS AND DIALOGS	15
5.1.1	FILE MENU	15
5.1.2	EDIT MENU	16
5.1.3	SEARCH/QUERY MENU	19
5.1.4	USER OPTION MENU	20
5.1.5	HELP MENU	20
5.1.6	DIALOG BOXES	21
5.2	PROCESS DESCRIPTION	31
5.2.1	DMCS Interfaces	32
5.2.2	User Interface (CDMSUI)	50
5.2.3	Final Plot (AutoCAD User Interface)	55
5.2.4	Report Generation	57
5.3	DATA DETAIL DESIGN	63
5.3.1	CDMS Field Definitions	63
5.3.2	DMCS Setup Definitions	65
6.0	REFERENCE DOCUMENTS	76
APPENDIX A	CDMS Hardware Environment	77
APPENDIX B	CDMS LAN INTERFACES	78
APPENDIX C	GUI/PCI INTERFACE	80
APPENDIX D	UNIX System and DMCS Customization	82

CAD Data Set Management System**(CDMS)****SYSTEM DESIGN DESCRIPTION (SDD)****1.0 INTRODUCTION****1.1 PURPOSE**

This document is intended to formalize the program design of the CAD Data Set Management System (CDMS) and to be the vehicle to communicate the design to the Engineering, Design Services, and Configuration Management organizations and the WHC IRM Analysts/Programmers.

The SDD shows how the software system will be structured to satisfy the requirements identified in the WHC-SD-GN-CSRS-30005 CDMS Software Requirements Specification (SRS). It is a description of the software structure, software components, interfaces, and data that make up the CDMS System.

This document is to be a "living document" and shall be maintained by revision for the life of the CDMS System, through system operation and maintenance.

1.2 SCOPE

The design descriptions contained within this document will describe in detail the software product that will be developed to assist the aforementioned organizations for the express purpose of managing CAD data sets associated with released drawings, replacing the existing locally developed system and laying the foundation for automating the configuration management.

1.3 OVERVIEW

The CAD Data Set Management System will provide for the storage, management and easy retrieval of the computer files (data sets) created for released drawings through the use of drafting or design software. These drawings are created on several different computer platforms, such as: AutoCAD on UNIX (Sun, HP and DEC), and MS/DOS computers, ComputerVision Dimension III on HP UNIX computers, and ME10 on UNIX computers.

See Appendix A for a description of the CDMS hardware environment.

See Appendix B for a description of the LAN interfaces and data paths.

This document is organized as prescribed in WHC-CM-3-10 Software Practices. A brief description of the sections found in this document are listed below and will be used to facilitate the software construction for this project.

- 2.0 Decomposition Description - Contains Logical Data Model and Logical Process Model including descriptions and definitions
- 3.0 Dependency Description - Includes Intermodule, Interprocess, Data, and Conversion dependencies
- 4.0 Interface Description - Module and Process Interface
- 5.0 Detail Design - System Flow Diagram, Detailed Process Flows and Detailed Data Descriptions

1.4 DEFINITIONS AND ACRONYMS

Hardware - Physical equipment used in data processing, as opposed to computer programs, procedures, rules, and associated documentation.

Quality Assurance (QA) - The process of ensuring that the end product conforms to established technical requirements as well as all the applicable standards in effect during the software development effort.

Requirement - (1) A condition or capability needed by a user to solve a problem or achieve an objective. (2) A condition or capability that must be met by a system or system component to satisfy a contract, standard, specification, or other formally imposed document. The set of all requirements forms the basis for subsequent development of the system or system component.

Software - Computer programs, procedures, rules, and associated documentation and data pertaining to the operation of a computer system.

System Design Description (SDD) - The system detail design which contains detail descriptions of the processing logic, data structures, and data definitions, in sufficient detail that the development team can begin software code construction.

System Project Management Plan (SPMP) - The controlling document for managing a software project. Defines the technical and managerial project functions, activities, and tasks necessary to satisfy the requirements of a software project.

System Requirements Specification (SRS) - A document that prescribes, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system or system component.

Validation - The process of evaluating system software and documentation at the end of the system development process to ensure compliance with requirements.

Verification - The process of determining whether or not the products of a given phase of the system development cycle fulfill the requirements established during the previous phase.

ACRONYMS

ANSI	-	American National Standards Institute
CDMS	-	CAD Data Set Management System
CDWS	-	Configuration Documentation Work Station
CM	-	Configuration Management
DMCS	-	Data Management and Control System (SDRC Product)
DOE-RL	-	Department of Energy, Richland Office
DS	-	Design Services
FTP	-	File Transfer Protocol
HAN	-	Hanford Area Network
IRM	-	Information Resource Management
PCI	-	Program Callable Interface
QA	-	Quality Assurance
SDD	-	System Design Document
SDRC	-	Structural Dynamics Research Corporation
SRS	-	System Requirements Specifications
V&V	-	Verification and Validation

2.0 DECOMPOSITION DESCRIPTION

2.1 MODULE DECOMPOSITION

2.1.1 CDMS Logical Data Model

(See following page for textual portion)

2.1.1 CDMS Logical Data Model (Cont'd)

Logical Entities

REQUESTER

Description: The requester can be anyone possessing a user id and password for the CDMS system. This includes individuals acting in the capacity of; system administrator, drawing release, drawing review, drawing modification, and final plotting of drawings.

Data Elements: Requester name, user id and encrypted password.

DRAWING/SHEET/REVISION

Description: Released drawings created on several different computer platforms; including AutoCAD on UNIX (Sun, HP, and DEC) and MS/DOS computers, ComputerVision Dimension III on HP UNIX computers and ME10 on unix computers. The drawings are owned by either KEH, PNL, or WHC.

Data Elements: Drawing No., Sheet No., Revision, Release Date, Comments, Title, Status (Check-out, Cancel, In Project), Project ID, Drawing Design Authority, Essential Flag, CAD Code and CAD File Name.

ORGANIZATION

Description: The structure within Westinghouse Hanford Company, Kaiser Engineers Hanford, or Pacific Northwest Labs associated with an organization identification code.

Data Elements: Organization Code, Manager, Off-Site AE, and Company (KEH, PNL, and WHC)

2.1.2 CDMS Logical Process Model

Information System Processes

2.1.2.1 Create/Update Labels

Design Services creates the drawing label or updates an existing drawing label in the Unreleased storage.

2.1.2.2 Check in Drawing Files

Design Services checks in the data sets received in 2.1.2.1 (above) to the Unreleased Drawing Folder and purges previous revisions of the file from the folder, if any.

2.1.2.3 Review Label/Folder

Configuration Management ensures the label exists and reviews the metadata and ensures the data set is in the folder. If the metadata and/or data set is not acceptable, Design Services is contacted for resolution of the anomaly.

2.1.2.4 Issue Label/Folder

Configuration Management issues the label for Release.

2.1.2.5 Label Revision

Design Services queries CDMS for the desired released label and CDMS creates a new unreleased revision of the label. Design Services checks the file out of the unreleased folder for revision of the drawing.

2.1.2.6 Replot Drawing

Design Services queries CDMS for the desired unreleased label and the drawing file is checked out of the folder for modification.

2.1.2.7 Cancel/Assign Ownership

Configuration Management Administrator queries CDMS for an unreleased label and cancels or reassigns ownership.

2.1.2.8 Copy File

General User queries CDMS for a released label and copies a file out of the folder.

External Entities

Design Services

Design Services has the responsibility for maintaining and updating the drawings represented by the files tracked by CDMS.

Administrator

The CDMS System Administrator is appointed by Configuration Management and granted the authority in managing the system.

General User

Any individual on site with the authority to view drawings, including supporting documents, for the purpose of fulfilling their job requirements.

Data Stores

Unreleased State (NULL)

A temporary state for drawing labels and their supporting files while they are being reviewed, altered if necessary, and approved for release. The label's metadata is subject to change while it is in this state.

Released State (RELEASE)

This state is the final repository or archive of active records. The current revision cannot be modified and remains in "read only" mode. A new version of the drawing requires a revision of the drawing label, including a new revision number. The new label is then in the unreleased state.

A drawing label's metadata may only be changed by a designated individual with proper access authority while in this state.

Superceded revisions of drawing labels may be moved to archive media while in this state.

Data Flow

Drawing File/Metadata

Drawing File contains electronic copies of drawing and related documents that will ultimately be stored in a drawing label's folder. Metadata is data describing the Drawing File that will be entered into the CDMS by Design Services and verified by Configuration Management.

Label

Represents the drawing entity and is made up of the metadata for that entity.

Folder

The folder "contains" the actual files and is attached to and travels with the label.

Reject

An anomaly to the acceptable data set and metadata being reviewed for final release. Notification is forwarded to Design Services for resolution.

Accepted Package

Label and Folder has passed Configuration Management's review prior to release.

Released Package

Label and Folder is promoted to the released state (issued).

Revision Request

Design Services has need to revise existing drawing.

Current Revision

Latest release of drawing and metadata is retrieved for review.

Revised Drawing

Next revision of drawing is created.

Replot Request

There is a need for a drawing residing in the unreleased folder to be replotted.

Unreleased Label/Folder

A Label and it's Folder remain in the unreleased state until approval for release by Configuration Management.

Replotted Drawing

Drawing retrieved from unreleased folder was replotted. Each time a drawing is "replotted" a new copy of the dataset is created. Purging removes the older copies.

Change Owner Request

CDMS System Administrator desires to cancel or assign ownership of data set.

Copy Request

General user requests a copy of data set.

Copy

Desired data set is retrieved, copied, for user.

3.0 DEPENDENCY DESCRIPTION

3.1 INTERMODULE DEPENDENCIES

The intermodule dependencies are addressed in the DMCS System documentation, namely, "Software Installation and System Administrators Guide (P-0632)" and "Planning, Implementation and Customization Guide (P-6028)".

3.2 INTERPROCESS DEPENDENCIES

The interprocess dependencies are addressed in the DMCS "Software Installation and System Administrators Guide" and "Planning, Implementation and Customization Guide".

A command interpreter based on DMCS's Open Program Callable Interface (PCI) will be created. This command interpreter will allow external programs access to pertinent CDMS functions without having to use the DMCS interface.

The CDMS PCI interface will consist mainly of a command parser/interpreter which will call functions written to use routines in the DMCS PCI library. It will be invoked from the Unix shell command line after logging onto the CDMS system. See Section 5.0 Detailed Design and appendix D for more detail.

3.3 DATA DEPENDENCIES

The data management of CDMS is controlled by the DMCS database. The customization of the CDMS System, required to accommodate the onsite CAD archiving and tracking, is accomplished through "menu driven" solicitations. The data structures and data relationships are constructed and maintained by the DMCS after soliciting the user in an interactive mode for data attributes, including keys.

See Appendix B for an description of the data paths.

3.4 CONVERSION DEPENDENCIES

There is no conversion requirement for the CAD based drawings or the data associated with the drawings.

4.0 INTERFACE DESCRIPTION

4.1 MODULE INTERFACES

The module interfaces are addressed in the DMCS "Software Intallation and Systems Administrators Guide" and "Planning, Implementation and Customization Guide". The guides will be in the possession of the CDMS system administrator. Any modications to the modules of the vendor software (DMCS), for this phase, will be defined in detail in this document.

See Appendix A for a description of the LAN interfaces.

4.2 PROCESS INTERFACES

4.2.1 File Transfers

The CDMS system is structured to work with two types of file areas: an Application Space and the Vault. The user's store their files in the Application Space and the files under CDMS control are placed in the Vault. Several Application Spaces can be defined to the system and the Vault can be designed to span several file systems as well. It should be noted, however, that the Vault is considered to be one entity and not many.

Those computers that allow File Transfer Protocol (FTP) connections to be made to them will have an Application Space established on the local file system and allow CDMS to copy files to/from that computer. Files that are ready to be placed under CDMS control will need to be placed in that Application Space.

All other systems that are able will transfer files directly to/from an Application Space area accessible by CDMS.

Most DOS systems will transfer files to an Application Space on an NT file server. CDMS will copy the files to/from the NT file server's Application Space by using FTP services on that file server.

4.2.2 Final Plot

The CDMS will accomodate the final plot process on all AutoCAD systems. The user will access and operate the final plot sequence from within the vendor software, as modified by this project. This final plot interface will accomplish the following:

- Be invoked through menu selection.
- Query the drawing file for information pertinent to the release process.

- Decode the CADCODE designator into its component parts (eg., OS Type, OS Version, CAD Software, Cad Software Revision).
- Obtain the system time and date, and present this data with the data obtained above to the user for final edit.
- Calculate a verification code for the released drawing data sets and reference files.
- Copy the CAD data set and associated reference files to an established CDMS Pre-Released Holding Area.
- Create the label for use by a Configuration Documentation Work Station (CDWS) at the time of drawing release.

4.2.3 CDMS PCI Interface

The CDMS PCI interface will consist mainly of a command parser/interpreter which will call functions written to use routines in the DMCS PCI library. DMCS PCI programming is described in the DMCS System documentation, namely, "Open PCI Programmer's Guide (P-0634)".

The command interpreter will be written in C and invoked from a Unix shell. It will accept typed commands from stdin and report results and data to stdout. A special Unix account will be given to the command interpreter and programs external to the CDMS system will be placed in the command interpreter when they log on.

The command interpreter will also be established to listen to socket connections on a designated port on the Unix system. The inetd process will perform the necessary establishment of sockets communications when a request comes in on the designated port. For more information, refer to appendix D.

5.0 DETAILED DESIGN

5.1 CDMS MENUS AND DIALOGS

The CDMS user interface will be a Windows interface with pull down menu's and pop-up dialog boxes. When the user starts CDMS they will be presented with the following menus and dialog boxes to interact with. *The detailed processes for interacting with the menus and dialog screens are described in 5.2.2 User Interface in the Process Descriptions.*

Logon CDMS will start with a login dialog prompting the user for their user name and password. The user name and password is sent to CDMS for login verification. The user authorization will be returned if the user name and password are accepted by CDMS.

Menu selections

5.1.1 FILE MENU

FILE

Reference	This menu selection will be used to retrieve the currently active data set for reference. This menu item will only be activated after a query for reference or checkout has been requested.
Checkout	This menu selection, used by Design Services (DS) only, will be used to check out the currently displayed drawing for revision. This menu item will only be active after a query for checkout has been requested.
Update	This menu selection, used by DS only, will be used to retrieve the currently displayed drawing for replot. This menu item will only be active after a query for replot has been requested.

Cancel This menu selection will allow DS and Configuration Management (CM) to cancel or return the currently displayed label/drawing. This menu item will only be active after a query for Cancel is requested. The screens will be different depending on the individual needs of DS and CM.

Print This menu selection will print the current label information to a local printer.

5.1.2 EDIT MENU

<u>E</u>dit
<u>S</u>ubmit for release...
<u>I</u>ssue/<u>R</u>elease

EDIT

Submit for Release (Design Services only)

This menu item is used to submit a data set without the use of the final plot program. The user will be prompted for the label information. The user will then enter the information from the key board or an IDT file.

The non-plot submittal will involve the end user logging onto the CDMS system and interactively using DMCS. Initially, this interface will be the standard DMCS interface, simplified by customizing the DMCS menus for the different user groups that will be using the system. The underlying transactions in DMCS's "Process Graph" will also be customized where necessary to simplify the interface.

A locally developed front end to the CDMS system will be developed and interfaced through the same PCI program developed for the Final Plot process. This program will use the same functions and features developed for the DMCS interface, but present them to the user in a more modern format.

CDMS Process FlowRelease Process Flow

DS creates the Drawing label or updates an existing Drawing label

DS checks files into the Drawing label folder

DS takes physical drawing to the CDWS

CM ensures the label exists and reviews metadata

CM notifies DS if there are any problems with the drawing or metadata, otherwise

CM issues the label

Update Process Flow

DS queries DMCS for an unreleased label

DS checks the file out of the folder for modification

The rest of the process follows the Release Process

Revise Process Flow

DS queries DMCS for a Released label
 DS creates a new revision of the label
 DS checks the file out of the folder for modification
 The rest of the process follows the Release Process

Issue/Release (Configuration Management only)

This menu option is used by CM to approve a selected data set for release.

Following the successful completion of the final plot process the drawing is forwarded to a CDWS for release. At the CDWS the drawing is processed by the CDWS clerk who verifies required Configuration Management drawing information is accurate and that the drawing qualifies for official release. To assist the CDWS clerk in the release and verification of the associated CAD data set, CDMS shall provide:

1. A convenient means of verifying that the associated CAD data set is in the Pre-Release Holding Area (Unreleased Folder).
2. Assurance that the data set in the Pre-Released directory can not be overwritten or disturbed during the release process.
3. A display of the final plot metadata on a CDWS screen.

Upon verification that all release information is correct and acceptable, CDMS shall provide an option for the CDWS clerk to initiate action to move the data set from the the Pre-Release state to the Released Data state. At this time CDMS will also change the data set status such that it is available to an authorized drawing requester for checkout. Upon completion of the release process a data set will be immediately available for checkout.

In the event that the drawing is not approved for release, CDMS will provide a means for the CDWS clerk to notify Design Services of any anomalies.

The CDMS will allow data sets checked out for modification to only be checked back in by the requester who originally checked the data set out. The following screen layouts will be used to accomplish the release processes.

5.1.3 SEARCH/QUERY MENU

SEARCH/QUERY

- Reference** This menu option will be used to search for data sets that may be used for reference only. All users will have access. The Queries will be limited to data sets that have been approved for release.
- Checkout** This DS menu item is used to Query for data sets that may be checked out for revision. The query will only select those drawings that the users authorization and project status will allow them to checkout.
- Update** This DS menu item will return a list of all drawings that the current user has submitted for release. The user will then select from the list the drawing that they wish to replot.
- Cancel** This DS menu item will return a list of all drawings that the current user has checked out to them. The user may then select the drawing from the list that they wish to cancel.
- This CM menu item will allow CM personal to query for drawings that have been checked out for revision.

Issue This CM menu item will retrieve a list of all drawings that have been submitted for release and allow them to select a drawing to issue for release.

Note: All Queries that return more than one label but less than 20 will display the list of drawings in list dialog and allow the user to select the drawing that they wish to work with. Queries that return more than 20 items will display a message to the user asking them to be more specific with their query.

5.1.4 USER OPTION MENU

<u>O</u> ption
<u>C</u> hange Password...
<u>S</u> et Communication Timeout (5 minutes)

OPTION

Change Password	This menu item will display the password dialog which allows the user to change their password.
Set Communication Timeout	This menu option will allow the user to control the timeout interval for communication with the CDMS. If very large files are to be transferred, this number should be increased.

5.1.5 HELP MENU

<u>H</u> elp
<u>C</u> ontents
<u>A</u> bout

HELP

Help	This menu item will display help on the current topic or the help table of contents (Hot Key F1).
About	Display program title screen and revision status.

5.1.6 DIALOG BOXES

5.1.6.1 SEARCH

Search

Search Criteria

Drawing No:

Sheet No:

Design Auth:

Project:

Creator:

OK **Cancel**

Search

The search dialog box will be displayed when a user selects one of the search menu options. This dialog will allow the user to define a CDMS Query allowing the users access to drawing data.

5.1.6.2 FILE LIST

Query Result					
Zoom Selected Rows		Print Table		Close	
#	Drawing No	Sheet No	Rev	Title	Release Date
1	1234	1	0	TEST CASE 1234	
2	H-2-123456	1	2	TEST DRAWING FOR CDMS	

Files

The files dialog box will be displayed when ever a query returns more than one label but less than 20. This dialog will allow the users to select a drawings label from the list and then display the data in the display dialog.

5.1.6.3 LABEL DISPLAY

Drawing:H-2-123456 Sht:1			
Drawing #:	H-2-123456	Sheet:	1
		Revision:	2
Title:	TEST DRAWING FOR CDMS 123456		
Rel. Date:		Authority:	WHC
Project ID:	standard	Project Status:	☐ Essential: ☐
CAD Code:	2B:IBM:ACD2:12.0:NN	CAD File:	B123456A
Comments:	submit by max		
Submitted By:	DSWHC		
Files:	#	File Name	Description
	1	B123456A.DWG	B123456A.DWG : Master Drawing
	2	TEST02.DAT	TEST02.DAT : Reference File
<div> <div>Done</div> <div>Print</div> <div>Reference</div> <div>Checkout</div> </div>			

Display

The display dialog will display label data after a user has selected a label to review. Five labels may be reviewed at one time.

5.1.6.4 PRINT QUERY RESULT

Print

☒ All Columns ☐ Selected Columns [below]

(Click on column names to select for printing in same order as you select)

#	Drawing No	Sheet No	Title	Revision	Release Date	Drawn By	Drawing Authority	Software Used
+								

☒ All Records ☐ Selected Records ☐ Range [below]

From: To:

Records To Print

Font: Arial, 9.8 point

Ready

Print Query Result

The dialog will allow the user to print information from the rows in the query result grid to their Windows printer.

5.1.6.5 CHANGE PASSWORD

The image shows a graphical user interface dialog box titled "Change Password". The dialog has a title bar with a standard window icon on the left. Inside the dialog, there are three text input fields. The first is labeled "Old Password:", the second is labeled "New Password:", and the third is labeled "Verify:". Below these fields are two buttons: "OK" and "Cancel".

Change Password	
Old Password:	<input type="text"/>
New Password:	<input type="text"/>
Verify:	<input type="text"/>
<input type="button" value="OK"/> <input type="button" value="Cancel"/>	

Password The password dialog will be displayed when the users select the Change Password option from the Options menu. This dialog will allow the users to change their password.

5.1.6.6 DIRECTORY

Directory The target directory dialog will be displayed when a user selects a files menu item that copies a drawing from CDMS to the users workstation. This dialog allows the users to specify the directory in which they want the files copied.

5.1.6.7 LOGIN

Login The login dialog will be displayed as the user starts CDMS and will accept the users name and password.

5.1.6.8 Main Form

The image shows a graphical user interface window titled "CDMS". The window has a standard menu bar with the following items: File, Edit, Search, Option, Window, and Help. Below the menu bar is a large, empty rectangular area, likely for displaying data or messages. At the bottom of the window is a status bar. The status bar is divided into sections: the left section contains the word "Ready", and the right section contains the word "RELEASE".

This is the main form for the CDMS. Other forms normally appear as child forms to this form. Menu choices are along the top. A status bar is at the bottom of the form. Sections of the status bar are a User Message, current state and log flag (logging of transactions to and from CDMS).

5.1.6.9 SUBMIT

Submit			
Drawing #:	H-2-123456	Sheet:	1
		Rev:	2
Title:	TEST DRAWING FOR CDMS 123456		
Project ID:		Project Status:	☐ Essential ☐
CAD Code:	2B-IBM:ACD2:12.0:NN	CAD File:	B123456A
Comments:			
Submit By:		Authority:	WHC
Description:		Attached Files:	B123456A.DWG TEST02.DAT
Master Drawing			
Add	Cancel	Read IDT File	Attach Files

Save Path as Default		Path: c:\	Select the main file name first then select reference file names.
Driver:	c: [VOLUME 1]	File Names:	
Directories:	<ul style="list-style-type: none"> c:\ 123 backup bwthr cdms cdmsdev 	<ul style="list-style-type: none"> winpath.bat autoexec.bak autoexec.bat autoset.bat bmmfiles.oi command.com config.org config.sys icons laplink.tre 	Add All --> Clear All Remove <-- Done

Submit

The Submit dialog will be displayed to DS personnel when they select the Submit for Release button from the Edit menu.

5.1.6.10 START FORM

This start form will be presented to the user each time the CDMS UI program starts. If the user does not have the latest version of the program, the new version will be downloaded and instructions displayed in the form.

5.1.6.11 ABOUT

ABOUT FORM Displayed when user clicks About in the Help menu. The version number is extracted from the file VERSION.DAT. The product support section is derived from the file SUPPORT.TXT.

5.1.6.12 PRINTER/FONT

Printer Dialog for selecting standard Windows printer

Font Dialog

5.2 PROCESS DESCRIPTION

Data Management and Control System (DMCS) supplied by Structural Dynamics Research Corporation (SDRC) is a commercial supplied data management package. The system design description contained in this document (WHC-SD-GN-30007) will customize DMCS with the procedures described below to meet the CDMS requirements.

5.2.1 DMCS Interfaces

5.2.1.1 DMCSIF

The standard DMCS user interface will be customized for local use. This customization is described in detail in the DMCS System documentation, namely, "Planning, Implementation and Customization Guide (P-6028)". See also section 5.2.2.

5.2.1.2 CDMS PCI Interface

The CDMS PCI interface will consist mainly of a command parser/interpreter which will call functions written to use routines in the DMCS PCI library. DMCS PCI programming is described in the DMCS System documentation, namely, "Open PCI Programmer's Guide (P-0634)". For more information on the implementation of the CDMS PCI interface, refer to appendix D.

5.2.1.3 Final Plot Interface for AutoCAD (DMCS View)

The final plot interface will essentially use the same underlying processes of DMCS that a user logging on interactively would use. The interface will be developed two ways:

1. Through a PCI program invoked by the final plot process after logging in to the CDMS system. This interface will involve sending streams of data to the program and receiving information back in a stream.
2. The interface above will be migrated to a TCP/IP socket interface to provide more efficient use of the network resources and to take advantage of the site switching all DOS computers to TCP/IP as the standard network protocol.

The processes that the final plot program will have to use are outlined here:

1. Log on to CDMS and invoke the PCI interface program
2. Query the knowledge base for an existing label for the data to be added to the system.
3. If an existing label doesn't exist, create one with the current information retrieved from the drawing. Continue at step 8.
4. If a label already exists, check that it is in the NULL, unissued state.
5. If the label is in the NULL state, verify that it has been checked out for modification by the current user.
6. If either of the tests in 4 and 5 fail, return an error message to the user and quit.
7. If the tests in 4 and 5 both pass, compare the current information to that retrieved from the existing label and make changes where necessary.

8. Check the file(s) into the folder.

9. Exit.

5.2.1.4 Copy Requested Data Set

After validation of authorization and data set status, a copy of the data set will be copied to the requestor's local file system. This will be accomplished in the following manner:

Input: The name, area and password of the source CDMS file server area (i.e., \\CDMFS\ASPACE PASSWD)

The directory of the source file including the drive letter (i.e., C:\CAD1)

The name of the file to be copied (i.e., CADWF.001)

Output: The file to be copied will be placed on the local drive and directory.

A 16 bit return status.

Method: The file is copied from the source to the destination and verified to insure it was copied correctly. The buffer is then written to the destination using the 8.3 name of the source (i.e., CADWF.001). If a file already exists on the destination with the same name, the file will be deleted and replaced with the new source file. Upon reaching the end of file of source, the status is returned to the caller. If errors are detected, a non-zero completion status is returned. The buffer size should be a multiple of 128 bytes, be as large as practical, and not exceed 64K bytes. The error completion codes are as follows:

- (1) The source file could not be opened.
- (2) The destination file could not be opened,
- (3) There was not enough room on the destination to receive all the data.
- (4) The source file is empty.

Diagram for process on next page.

Reverse the procedure for copying data sets to the CDMS file server from the requestor's hard drive.

Note: The CDMS system will use crc codes to verify the file transfers between the Application Space and the CDMS vault. The end user is responsible for providing a

good copy of the file to the Application Space. This may be handled by the various user interfaces being created for the system.

5.2.1.5 CDMS PCI Command Descriptions

The following commands will be implemented in the CDMS PCI command interface. This interface will be used with CDMSUI. Each of the commands is shown with its' abbreviation in paranthesis.

help (?)

usage: help <cmd_name>

example: help help

description:

help prints out a short description of the commands implemented in the cdmspci interpreter. For a list of commands, type 'help commands'.

exit (ex)

usage: exit

example: exit

description:

exits from CDMS:

See also - quit

quit (q)

usage: quit

example: quit

description:

exits from CDMS.

See also - exit

list_users (lusr)

usage: list_users

example: list_users

description:

List all the user names known to CDMS.

user_info (uinf)

usage: user_info <userid>

example: user_info joe

description:

Lists the user information about the indicated user.
If no user name is given, information about the
current user is given.

user_auth (uauth)

usage: user_auth <userid>

example: user_auth joe

description:

Lists the groups the indicated user belongs to.
These groups indicate the authorizations the user
has. If no user name is given, information about
the current user is given.

list_aspace (lasp)

usage: list_aspace

example: list_aspace

description:

Lists the available application spaces known to
CDMS. These application spaces are where DMCS
copies file to/from.

change_password (chpw)

usage: change_password <old password> <new password>

example: change_password

description:

Allows the user to change his/her password for
CDMS. If the old and new passwords are not given
on the command line, the user will be prompted for
them. During prompting, the user will be prompted
for the new password a second time as a verification.

setenv (set)

usage: set <envname> = <value>

example: set current state = NULL

description:

CDMSPCI keeps track of certain global information in "environment" variables that can be changed on the command line to affect operation. For instance, the 'current state' variable determines whether label queries will show RELEASED or UNRELEASED (NULL) state labels.

New variables can be added if desired. If no value is given, the variable will still exist, but the value will be erased.

See also - showenv

showenv (show)

usage: show <envname>

example: show current state

description:

Displays the current setting of the indicated environment variable. If no environment variable is given, all variables are shown. Variables that have had their value erased, or non-existent variables will be shown as '(null)'.

See also - setenv

label_add (laad)

usage: label_add

example: label_add

description:

Causes a new label to be created in the CDMS knowledge base. The label attributes must be sent following the label add command enclosed between a pair of parenthesis, one per line with the attribute name first on the line, followed by a colon, then the attribute value. An example session would be as follows:

```
>> label_add
(
dwg_no: H-2-12345
```

```
dwg_sheet: 1
:
:
)
```

An indication of the success or failure of the label_add will follow. If during the process of sending attributes, an unrecognized attribute name is given, an immediate notification will be given.

See also - label_query, label_update

label_query (laqy)

usage: label_query

example: label_query

description:

Queries the CDMS knowledge base for labels meeting the required attributes. The attributes are indicated using the same method as label_add. The number of labels that match the query is returned.

See also - label_add, get_label

label_update (laup)

usage: label_update <label number>

example: label_update 1

description:

Updates the label indicated with the attributes given following the label_update command. These attributes are given in the same fashion as the label_add command. The label number refers to the number assigned to it in the current query. To view the information before changing, use the get_label command.

See also - label_add, get_label

note: only labels in the unreleased (NULL) state can be updated.

get_label (getlab)

usage: get_label <label number>

example: get_label 1

description:

Displays the attribute information about the indicated label from the CDMS knowledge base. The information is displayed enclosed in a pair of parenthesis, one attribute per line with the attribute name first, followed by a colon then the attribute value. This is the same format used by label_add, label_query and label_update when entering data into the knowledge base. If the word 'all' is given in place of the label number, all labels from the current query will be displayed, each enclosed in its own set of parenthesis.

See also - label_add, label_query, label_update

label_issue (lais)

usage: label_issue <label number>

example: label_issue 1

description:

Tell CDMS to move the indicated label to the RELEASE state. The label number is the number assigned to it from the current label query.

See also - label_query

note: only labels in the unreleased (NULL) state can be issued.

label_createrev (lacr)

usage: label_createrev <label number>

example: label_createrev 1

description:

Tell CDMS to create a new revision for the indicated label. The label number is the number assigned to it from the current label query.

See also - label_query

note: only labels in the released (RELEASE) state can be revised.

cancel_rev (CARV)

usage: cancel_rev <label number>

example: cancel_rev 1

description:

Tell CDMS to cancel the current revision for the indicated label and revert to the former revision. The label number is the number assigned to it from the current label query.

See also - label_query

note: only labels in the unreleased (NULL) state that are revisions of previously release labels can be cancelled. If new folder items have been added to the folder, these must be deleted before the revision can be cancelled. New labels cannot be cancelled, they must be deleted.

folder_query (foqy)

usage: folder_query <label number>

example: folder_query 1

description:

Query the folder for the indicated label number. The label number is the number assigned to it from the current label query. The folder query returns the number of items in the label's folder. To retrieve information about each item, use get_folder.

See also - get_folder, label_query

get_folder (getfol)

usage: get_folder <folder item number>

example: get_folder 1

description:

Displays the attribute information about the indicated folder item using the same format as get_label. The folder item number is the number assigned from the current folder query. If the word 'all' is used in place of the folder item number, the attribute information about each item in the folder will be displayed.

See also - `folder_query`, `label_query`, `get_label`

note: The folder items returned are from the current folder query which is related to a particular label item from the current label query. They should not be confused with the folders related to other labels in the query.

`folder_checkin (foci)`

usage: `folder_checkin` <label number> <description> \
 <location> <filename>

example: `folder_checkin` 1 main my_aspace BxxxxxA.DWG

description:

Copies a file from the application space indicated by 'location' into the folder for the indicated label. The label number is the number assigned from the current label query. The description should uniquely identify the folder item in the folder (must not already exist). If the description is more than one word, it must be enclosed in single (') or double (") quotes. If the information above is not given on the command line, it will be prompted for.

See also - `label_query`, `folder_replace`

`folder_delete (fode)`

usage: `folder_delete` <label number> <folder item>

example: `folder_delete` 1 1

description:

Delete the indicated folder item out of the folder. If the word "all" is used for the folder item, all items in the folder will be deleted.

`folder_replace (fore)`

usage: `folder_replace` <label number> <folder item> \
 <location> <filename>

example: `folder_replace` 1 1 my_aspace BxxxxxA.DWG

description:

Copies a file from the application space indicated by 'location' into the folder for the indicated label, replacing the indicated folder item. The label number

is the number assigned from the current label query.
The folder item is the number assigned in the current
folder query. If the information above is not given
on the command line, it will be prompted for.

See also - label_query, folder_checkin

folder_copyout (foco)

usage: folder_copyout <label number> <folder item> \
 <destination> <filename>

example: folder_copyout 1 1 my_aspace BxxxxxA.DWG

description:

Copies a file from the folder to the application space
specified as 'destination'. The label number and folder
item specify which file is to be copied and correspond
to the current label and folder queries. The filename
is used to name the file on the destination. If the
required information is not given on the command line,
it will be prompted for.

See also - label_query, folder_query, folder_modify

folder_modify (fomo)

usage: folder_modify <label number> <folder item> \
 <destination> <filename>

example: folder_modify 1 1 my_aspace BxxxxxA.DWG

description:

Copies a file from the folder to the application space
specified as 'destination' and marks it for modification
in the knowledge base. The label number and folder
item specify which file is to be copied and correspond
to the current label and folder queries. The filename
is used to name the file on the destination. If the
required information is not given on the command line,
it will be prompted for.

See also - label_query, folder_query, folder_copyout

cancel_owner (caown)

usage: cancel_owner <label number> <folder item>

example: cancel_owner 1 1

description:

Cancels the ownership of the folder item in the indicated label. The label number and folder item correspond to the current label and folder queries.

note: This operation can only be performed by the current owner of the folder item.

change_owner (chown)

usage: change_owner <label number> <folder item> <userid>

example: change_owner 1 1 user2

description:

Changes the ownership of the indicated folder item to userid. If there was previously no owner, this assigns one. If the new userid is empty, ownership is removed. The label number and folder item correspond to the current label and folder queries.

note: This operation can only be performed by a user who has the authority to assign the ownership of file items. The label must also be in the pre-release (NULL) state.

label_reject (larj)

usage: label_reject <label number>

example: label_reject 1

description:

Send a rejection message about the indicated label to the person who last modified the label data. The label number corresponds to the last label query. The message follows the command line and is enclosed in parenthesis. The closing parenthesis must appear on a line by itself. For example:

```
>> label_reject 1
(
This is the message to send.
:
:
)
```

Messages are sent through the Unix mail system. The label's drawing number, sheet and revision and the sender's user ID are put in the message header.

note: This operation can only be performed by a user who has the authority to issue/reject drawings.

label_archive (laar)

usage: label_archive <label number>

example: label_archive 1

description:

Tell CDMS to mark the indicated label for archival. Archived labels will not have their associated folder items readily accessible. The actual archiving of the folder items takes place in a background task that is run on a periodic basis.

See also - label_restore

label_restore (lars)

usage: label_restore <label number>

example: label_restore 1

description:

Tell CDMS to mark the indicated label for restoration after its folder items have been previously archived. The restoration process takes place in the background only as requested by the system operator.

See also - label_archive

noop

usage: noop

example: noop

description:

No operation - do nothing.

mail (ml)

usage: mail <user ID>

example: mail user1

description:

Send a Unix mail message to the indicate user ID. The message is collected on subsequent lines, enclosed

in parenthesis. The closing parenthesis must appear on a line by itself. For example:

```
>> mail user1
(
This is the mail message.
:
:
)
```

The sender's user ID is put in the message header.

label_owner (laown)

usage: label_owner <user ID>

example: label_owner user1

description:

Get a list of all the labels where the indicated user has checked a file out of the folder for modification. If no user name is given, the current user id is used.

share_info (shin)

usage: share_info <application space name>

example: share_info aspace1

description:

Get DOS share information for the indicated application space.

5.2.1.6 DMCS Access Controls

5.2.1.6.1 Verify Request Authority

Input: Twenty five character user name.

Eight character user password (case sensitive).

Output: If user is not in DMCS system, an error status is returned, otherwise a status of zero is returned.

Method: DMCS controls access and user authorizations through user groups. The privileges and capabilities of each user group is inherited from the user group's parent(s). At each group level, the privileges can be restricted. The individual user is a group that has no children.

CDMS will establish the following user types.

- CDMS Administrator:** These user's will be able to do all the DMCS functions. These users will add and delete other users, establish user groups as needed, assign and designate drawing ownerships as needed, request reports, maintain table fields, such as projects and check out reasons, and update any drawing data, delete, add, or replace folder items, delete drawing data, lock and unlock controlled entities (drawings), and all other functions related to system control.
- CDMS Release:** These users will promote a drawing label into the DMCS **released** state. They will review and modify any drawing data and use the issue command to make the promotion.
- Drawing Requestor:** These users retrieve drawings for review. They will query drawings and retrieve drawing folder items to their local hard disks.
- Controlling Authority:** These users will request a revision for drawings (check out for modification) that are under company control. There will be at least three different company control groups; WHC, Kaiser, and PNL.
- Essential:** These users will be able to request a revision (check out for modification) for drawings that have the essential flag set. Each Design Authority (company) will have an essential group.
- Designer:** These users will do a final plot (add a new drawing to DMCS), and revise (check out for modification) released drawings. To be able work with "in project" and company controlled drawings, the designer will also be a member of the "In Project" and "Controlling Authority" groups.

5.2.1.6.2 Authorized Access and User Authorization Matrix

	CDMS Admin.	CDMS Release	Drawing Request	Control Auth.	Essential	Design
Release Drawing	X	X				
Issue Contolled Entities	X	X				
Review Metadata	X	X	X	X	X	X
Modify Metadata	X	X		X	X	X
Retrieve Drawing/ Folder Items	X	X	X			
Check Out/Modify Company Controlled				X		
Check Out/ Modify In Essential Drawings					X	
Add Final Plot						X
Revise Final Plot						X

5.2.1.6.3 CDMS General Access Rules

Any GENERAL or Design Services user can get a copy of a dataset.

Only Design Services can check a dataset out for modification.

Only those who belong to the same company group as the Design Authority listed in the drawing's label can check out for modification a dataset with a non blank design authority field.

Only those who belong to the correct essential drawing group can check a dataset out for modification if that dataset has the essential flag set.

Any Design Services user can cancel his/her ownership of a dataset.

Configuration Management can cancel/reassign ownership of any dataset.

Configuration Management will perform all issuing (releasing) of labels (datasets).

Design Services will be able to create/update labels that are in the null (prereleased) state.

Designed Services will create revisions of already issued (released) labels.

Setup option

This menu item will allow the user to define the CDMS setup parameters for their workstation.

5.2.1.7 CDMS Backup Procedures

Database Backup

Backups will be executed nightly.

Oracle export facilities will be used to create the backup file which will subsequently be copied to the backup media.

On weekdays an incremental backup will be performed and overwritten each week.

On Saturday a full export and backup will be performed and saved for a year.

An annual full export and backup will be performed and saved indefinitely.

Database Redo Logs (Transaction Logs)

The automatic archiving option of Oracle's redo logs will be used.

The archived logs will reside on a separate disk from the database files.

The redo log files will be backed up on the same basis as the exported database files with the exception that no special provision will be made to save a past year's redo logs during the annual backup.

The database control files will be backed up with the redo logs.

Dataset Vault Backup

Backups will be executed nightly.

On weekdays an incremental backup will be performed and overwritten each week.

On Saturday an incremental backup will be performed and saved for a year.

An annual full backup will be performed and saved indefinitely.

Archiving Datasets to Offline Media

CM will flag the labels for the datasets to be archived using the DMCS facility provided for this function.

A notification will be sent to the operator that archival needs to be accomplished.

The operator will mount the archive backup media and execute a script to perform the archival.

The script will perform the following:

'archive archive': a DMCS utility which lists the files to be archived.

Parse the resulting list file and copy the affected files to the backup media.

'post_archive archive': a DMCS utility which updates the DMCS knowledge base with the archive status of the affected files.

Restoring Archived Datasets from the Offline Media

CM will flag the labels for the datasets to be restored using the DMCS facility provided for this function.

A notification will be sent to the operator that restoration needs to be accomplished.

The operator will mount the archive media and execute a script to perform the restoration.

The script will perform the following:

'archive restore': a DMCS utility which lists the files to be restored.

Parse the resulting list file and copy the affected files from the backup media to the Dataset Vault.

'post_archive restore': a DMCS utility which updates the DMCS knowledge base with the archive status of the affected files.

5.2.1.8 CDMS Version Control

Version control on the individual pieces of the CDMS system will be handled through the IRM Change Control Board Process (see also the Configuration Management Plan).

5.2.2 User Interface (CDMSUI)

The CDMSUI is a windows based user interface that communicates with DMCS through the PCI interface. The CDMSUI will be used by Design Services, Configuration Management and General users when interacting with DMCS. CDMSUI will provide the following functionality:

Log the user on to DMCS.

Collect user requirement for and send data Queries.

Display the Query results and allow the users to retrieve CAD data from the DMCS for reference or revision.

Allow users to submit revised, existing or add new CAD data to DMCS.

Allow Configuration Management to approve revised or new CAD data before release.

Allow users to set and change the DMCS password.

The following outline defines the basic CDMSUI interface. Each section represents the actions taken when the Windows menu options are requested.

(Numbers enclosed in parenthesis, i.e. (5.1.1), indicate section number in 5.0 Detailed Design that contains screen layout.)

For more information on how the fields were implemented on the CDMSUI, refer to appendix C.

User interface outline

Logon

- Display Logon dialog (5.1.6.7)
- Get user name and password from dialog
- Send LOGON (User name, Password)
- Wait for response
- Get user autorizations
- Get user Application Space information

5.2.2.1 FILE (5.1.1)

Reference

Note: This routine is the same for Reference, Checkout and Update.

- Get record number(s) of active Label Display dialog
- For each record number in the input array:
 - Get label information.
 - Extract CAD file name (strip off extension)
 - Create subdirectory in application space.
 - If copying for reference, then create \ref\cadfilename, otherwise, create \cadfilename.
 - Get number of files in folder (folder query)

- For each file
 - Get file name
 - If copying for Reference then, issue PCI folder copy out otherwise issue PCI folder modify.
 - Create a log file (cadfilename.log) in application space root directory.

If copying down for reference, then also

- Ask user for local directory to copy files.
- If user specifies local directory:
 - Copy files from application space subdirectory to user's local directory.
 - Remove application space subdirectory.
 - Delete the log file.

Checkout (DS only)

Allow checkout of only one drawing at a time.

Check 'revision pending flag' must not be '+' (already checked out).

Send Create Revision.

Set to NULL state.

Issue label query for drawing and sheet.

Perform Folder download (see above routine for Reference).

Update (DS only)

See above routine for Reference.

Cancel (DS and CM Different for each)

Get record number of active Label Display dialog

Send Folder Query (record #)

Send Folder Delete (record #, ALL)

Wait 60 seconds (DMCS needs time to perform its work).

Send CANCEL_REV (Record #)

Print

Get data from the active Label Display dialog

Print current record to local printer

Print Setup

Show dialog for changing Windows printer setup.

Exit

Drop connection to user's application space.

End program

5.2.2.2 EDIT (5.1.2)

Submit for Release (DS only)

Clear child forms.

Set State to NULL

Display Submit dialog (5.1.6.9)

If IDT button

Display file dialog (5.1.6.2)

Get file name

Read IDT file

Display data in Submit dialog (5.1.6.9)

Label_Query using Drawing and Sheet from IDT file.

If Query was successful, then update screen fields with Query results and turn on UPDATE button, otherwise turn on ADD button. Update screen fields with IDT data.

If user enters or modifies Drawing or Sheet number, then issue Label_Query and update screen fields with results (if found).

If ATTACH BUTTON

Display file input dialog (5.1.6.2)
Collect list of files.

If ADD/UPDATE BUTTON

Edit fields on screen.
Send LABEL_ADD or LABEL_UPDATE as appropriate.
Create a subdirectory of CAD File Name in the user's application space.
Do FOLDER QUERY
 Get a list of current file Descriptions by sending a GET_FOLDER for each of the files.
For each of the Files in the file list:
 Copy the file from its current location to the directory created (if not already there).
 Check the CRC after copying to make sure the copy was successful.
 Send the file to CDMS using FORE (folder replace) or FOCI (folder copyin) as appropriate. If the file description is already in use, then a replace will be used.
 Delete file from server

Remove directory from server.
Delete the IDT file.

Issue/Release (CM only)

Get record number of active Label Display dialog
Send LABEL_LABEL_ISSUE (Record #)
Wait for response

5.2.2.3 SEARCH/QUERY (5.1.3)

Reference

Clear child forms
Set Query State = RELEASED
Display query dialog (5.1.6.1)
Build query (drawing, sheet, drawing auth., project, creator)
Send LABEL_QUERY (Query)
 Wait for response (Number of Records matched)
 If error display message
 If more than 20 items in query response
 Display Search dialog w/message to refine query (5.1.6.1)
else
 Retrieve each record
 Store data in data_struct
 Add struct to list

Display File List dialog
 Display Label Display dialog for each drawing selected 5 max (5.1.6.3)
 Update menu status
 Highlight file/reference gray others

Checkout (DS only)
 Clear child forms
 Set Query State = RELEASED
 Display query dialog (5.1.6.1)
 Build query (drawing, sheet, drawing auth., project, creator)
 Send LABEL_QUERY (Query)
 Wait for response (Number of Records matched)
 If error display message
 If more than 20 items in query response
 Display Search dialog with message to refine query (5.1.6.1)
 else
 Retrieve each record
 Store data in data_struct
 Add struct to list
 Display File List dialog (5.1.6.2)
 Display Label Display dialog for each drawing selected 5 max (5.1.6.3)
 Update menu status
 Highlight file/reference and file/checkout gray others

Update (DS only)
 Clear child forms.
 Set Query State = NULL
 Display query dialog (5.1.6.1)
 Default creator to current User.ID
 Build query (drawing, sheet, drawing auth., project, creator)
 Send LABEL_QUERY (Query)
 Wait for response (Number of Records matched)
 If error display message
 If more than 20 items in query response
 Display Search dialog with message to refine query
 else
 Retrieve each record
 Store data in data_struct
 Add struct to list
 Display drawing list (5.1.6.2)
 Display data dialog for selected drawings (5.1.6.3)
 Update menu status
 Highlight file/update gray others

Cancel (DS only)
 Set Query State = NULL
 Display dialog to get User.ID (default to current user)
 Send LAOWN (label owner) query to CDMS.
 Build a list of labels from result of query.
 Show the list to user and have them select one.
 Build Query (drawing, sheet, revision)
 Send LABEL_QUERY (Query)
 Wait for response (Number of Records matched)
 If error display message
 Retrieve each record

- Store data in data_struct
- Add struct to list
- Display drawing list (5.1.6.2)
- Display data dialog for selected drawings (5.1.6.3)
- Update menu status
- Highlight file/Cancel gray others

Issue (CM only)

- Clear child forms.
- Set Query State = NULL
- Display query dialog (5.1.6.1)
- Default creator to current User.ID
- Build query (drawing, sheet, drawing auth., project, creator)
- Send LABEL_QUERY (Query)
 - Wait for response (Number of Records matched)
 - If error display message
 - If more than 20 items in query response
 - Display Search dialog with message to refine query
 - else
 - Retrieve each record
 - Store data in data_struct
 - Add struct to list
- Display drawing list (5.1.6.2)
- Display data dialog for selected drawings (5.1.6.3)
- Update menu status
- Highlight edit/issue gray others

5.2.2.4 OPTION (5.1.4)

Change Password

- Get user name
- Display Change Password dialog (5.1.6.5)
- Get password info
- Verify new password and verify are equal
- If Not
 - Goto Display change password dialog (5.1.6.5)
- Else
 - Send CHANGE_PASSWORD (User name, Old Password, New password)
 - Wait for response (Result code)

Set Communication Timeout

- Display dialog for user to change the timeout interval (in minutes) for communications with CDMS. This value is saved in the ini file for the user.

5.2.2.5 WINDOW ()

Cascade

- Standard windows cascade of child windows

Window List

- Standard windows open window list.

5.2.2.6 HELP (5.1.5)

Help (hot key F1)

Display current topic or Contents

About

Display program title screen and revision status

LABEL DISPLAY DIALOG

Folder Button

Set help flag

Get data from active Label Display

Send FOLDER_QUERY (Record #)

Wait for response (Result code or Data)

Display folder data in Folder dialog (5.1.6.4)

5.2.3 Final Plot (AutoCAD User Interface)

FinalPlot is a AutoCAD ADS application that will allow the users to submit AutoCAD data sets to CDMS from within a AutoCAD session at the time of the final plot. FinalPlot will extract data from the AutoCAD drawing file and display for review by the user. After aproval the meta data and CAD files will be submitted to DMCS for approval by Configuration Management.

Query the drawing file for information pertinent to the release process.

Decode the CADCODE designator into its component parts (eg., OS Type, OS Version, CAD Software, Cad Software Revision).

Obtain the system time and date, and present this data with the data obtained above to the user for final edit.

Calculate a verification code for the released drawing data sets and reference files.

Copy the CAD data set and associated reference files to an established CDMS Pre-Released Holding Area.

Create the label for use by a Configuration Documentation Work Station (CDWS) at the time of drawing release.

The Final Plot process will be executed from AutoCAD as part of the process to create a plot of the drawing to be circulated for approvals for the release process. The function will do the following:

1. Get user name and password from user.
2. Logon to the CDMS.
3. Send QUERY_LABEL with existing drawing and sheet number.

4. If label exists check to see if label is in the null state and that the user is the one who checked it out. If NOT error out.
5. Reset and save the drawing with the AutoCAD menu reference set to ACAD or NULL.
6. Check the drawing to determine if a purge is required. Save and purge the drawing if the user approves.
7. Display Submit dialog (see 5.1.1.6) With metadata from the data set. Allow the user to correct the data. Update the drawing, if required.
8. Break the CAD code into the individual pieces required for CDMS.
9. Collect a list of all externally referenced files: eg. fonts, shapes and Xref's).
10. Compare the Shapes against a list of approved shapes. Non-standard shapes will be copied as reference data.
11. Compare the existing data to the new information and change it where necessary.
12. Copy the data set from the local hard disk or server, to the CDMS system. Create a check value that can be used to verify the validity of the file.
13. Copy the externally referenced files to the CDMS system, if required.
14. Send LABEL_ADD or LABEL_UPDATE with meta data and CRC to CDMS.
15. Send FOLDER_CHECKIN for the drawing and all reference files.
16. Wait for verification of the label and Data set from CDMS.
17. Log off of CDMS
18. Plot the user data.
19. Remove the file from the user's local hard disk after plotting the file.

Note: The method for extracting the metadata for multiple drawing sheets in a single dataset must still be defined.

5.2.4 Report Generation

Standard Reports

All standard reports in CDMS will be generated automatically on a nightly basis using the UNIX "cron" facility. The reports will be stored on a single location on the CDMS system for easy retrieval by anyone who wishes to view them. Those reports that are to be displayed in Soft Reporting will be made available to that system. For more information on how the reports are scheduled to run on the Unix system, refer to appendix D.

Other Reports

A menu will be established within DMCS to allow CM to generate other reports as desired.

5.2.4.1 Usage By Organization Code

NUMBER:	CDMS-101
SYSTEM:	General
PURPOSE:	Lists times of system usage by organization
SEQUENCE:	Company, Organization, Requestor Name
SELECTION PARAMETERS:	Format is fixed
INPUT MEDIUM:	CRT
OUTPUT MEDIUM:	CRT, Hard Copy, Disk
FREQUENCY:	As required
INPUT FILE(S):	Organization, Requestor, System Provided
COMMENTS:	
CONTENT:	Org_Company, Org_Organization, Req_Name, Sys-Time, Sys-Date

CDMS-101

CAD DATA SET MANAGEMENT SYSTEM

11:02:00 10/04/93

USAGE BY ORGANIZATION

PAGE: 1

COMPANY	ORGANIZATION	REQUESTOR NAME	TIME	<---SIGNED ON--->		ELAPSED TIME	
				DATE			
KEH	K33330	GREEN, SARAH	10:00:00	10/05/93		00:43:00	
		SNITT, FRED	09:45:00	10/06/93		00:15:00	
		OTT, JIM	08:00:00	10/04/93		00:10:00	
		OTT, JIM	08:10:00	10/05/93		00:30:00	
	* TOTAL ORGANIZATION K33330					01:38:00	
	K34440	BROWN, ERIC	10:00:00	10/05/93		00:43:00	
		GROSS, ED	09:45:00	10/06/93		00:15:00	
		* TOTAL ORGANIZATION K34440					00:58:00
		** TOTAL COMPANY KEH					02:36:00
	PNL	P33330	BLUE, SARAH	10:00:00	10/05/93		00:43:00
SNOOP, FRED			09:45:00	10/06/93		00:15:00	
OTTO, JIM			08:00:00	10/04/93		00:10:00	
OTTO, JIM			08:10:00	10/05/93		00:30:00	
* TOTAL ORGANIZATION P33330					01:38:00		
P34440		BRUHN, ERIC	10:00:00	10/05/93		00:43:00	
		GRASS, ED	09:45:00	10/06/93		00:15:00	
		* TOTAL ORGANIZATION P34440					00:58:00
		** TOTAL COMPANY PNL					02:36:00

5.2.4.2 Data Set Information

NUMBER: CDMS-102
SYSTEM: General
PURPOSE: Lists data set information for CAD drawings
SEQUENCE: Drawing No., Sheet No., Rev. No.
**SELECTION
PARAMETERS:** Format is fixed
INPUT MEDIUM: CRT
OUTPUT MEDIUM: Soft Reporting
FREQUENCY: Daily
INPUT FILE(S): Drawing, CAD, Organization
COMMENTS:
CONTENT: Drw_Drawing_No, Drw_Sheet_No, Drw_Rev_No,
 Drw_Release_Date, Drw_Drawing_Title, Cad-File_Name,
 Cad_Cad-Code, Org_Off-Site_AE

CDMS-102

CAD DATA SET MANAGEMENT SYSTEM

11:02:00 10/04/93

DATA SET INFORMATION

PAGE: 1

DRAWING NUMBER	SHEET	REV.	RELEASED	FILE NAME	CAD CODE	TITLE	OFF-SITE AE
H-1-080001	1	0	06-02-93	A080001A.R00	5L:IBM:ACD2:11.0:SS	TANK NO. 100-B-1	ABCD

5.2.4.3 Data Set Check-out

NUMBER: CDMS-103
SYSTEM: General
PURPOSE: Lists CAD drawings checked out/checked out by
SEQUENCE: Drawing No., Sheet No., Rev. No.
SELECTION PARAMETERS: Format is fixed
INPUT MEDIUM: CRT
OUTPUT MEDIUM: Soft Reporting
FREQUENCY: Daily
INPUT FILE(S): Drawing, CAD, Organization
COMMENTS:
CONTENT: Drw_Drawing_No, Drw_Sheet_No, Drw_Rev_No,
Drw_Checkout_Date, Drw_Checkout_Name, Cad-File_Name,
Cad_Cad-Code, Org_Off_Site_AE

CDMS-103

CAD DATA SET MANAGEMENT SYSTEM

11:02:00 10/04/93

DATA SET CHECK-OUT

PAGE: 1

DRAWING NUMBER	SHEET	REV.	CAD CODE	CAD FILE NAME	CHECK-OUT	CHECKED OUT BY	OFF-SITE AE
H-1-080001	1	0	5L:IBM:ACD2:11.0:SS	A080001A.R000	09-27-93	JODY SLOKE	XXXX

5.2.4.4 Drawing Responsibility

NUMBER: CDMS-104
SYSTEM: General
PURPOSE: Lists CAD drawings checked out/checked out by
with check outs > 90 days
flagged.
SEQUENCE: Drawing No., Sheet No., Rev. No.
**SELECTION
PARAMETERS:** Format is fixed
INPUT MEDIUM: CRT
OUTPUT MEDIUM: CRT, Hard Copy, Disk
FREQUENCY: Monthly
INPUT FILE(S): Drawing, CAD, Organization
COMMENTS:
CONTENT: Drw_Drawing_No, Drw_Sheet_No, Drw_Rev_No,
Drw_Checkout_Date, Drw_Checkout_Name,
Drw_Drawing_Title, Org_Off_Site_AE

CDMS-104

CAD DATA SET MANAGEMENT SYSTEM

11:02:00 10/04/93

DRAWING RESPONSIBILITY

PAGE: 1

(* DENOTES DRAWING CHECKED OUT > 90 DAYS)

DRAWING NUMBER	SHEET REV.	CHECK-OUT	CHECKED OUT BY	OFF-SITE AE	TITLE
H-1-080001	1	0 9/15/93	JODY SLOKE	ABCD	TANK NO. 100-B-1
H-1-080001	2	0 06/26/93*	JODY SLOKE	ABCD	TANK NO. 100-B-1

5.2.4.5 List of Authorizations

NUMBER: CDMS-105

SYSTEM: General

PURPOSE: Lists Users and their corresponding authorization level

SEQUENCE: User Name

SELECTION PARAMETERS: Format is fixed

INPUT MEDIUM: CRT

OUTPUT MEDIUM: CRT, Hard Copy, Disk

FREQUENCY: As required

INPUT FILE(S): Authority, Requestor, Organization

COMMENTS:

CONTENT: Req_Name, Req_ID, Org_Code, Auth_Auth_Level

CDMS-X105-R

CAD DATA SET MANAGEMENT SYSTEM

11:02:00 10/04/93

LIST OF AUTHORIZATIONS

PAGE: 1

USER NAME	USER ID	ORG. CODE	GROUP AUTHORITY
JODY SLOKE	SLOKEJD	123456	XXXXXXXXXXXXXXXX

5.3 DATA DETAIL DESIGN

5.3.1 CDMS Field Definitions

The fields defined in the SRS will be incorporated into the DMCS system as user defined fields or by utilizing existing DMCS fields or utilities as noted. For information on how the fields were implemented in the CDMSUI, see appendix C.

SRS Name	Field Name	Data Type	DMCS/Local	Notes
Drawing Number	dwg_no	char(15)	local	
Sheet No.	dwg_sheet	num(4)	local	
	dwg_title	char(72)	local	
Revision	revision	num(10)	dmcs	
Release Date	approval_date	char(8)	local	
Drawn By	dwg_drawnby	char(30)	local	
Owner	dwg_auth	char(30)	local	
Project Status	project_id	char(14)	dmcs	
Checkout Status	group_id	char(25)	dmcs	from access history
Cancel Status	dwg_cancel	char(1)	local	
Comments	dwg_comments	char(60)	local	
	dwg_essential	char(1)	local	
	current state	char(8)	dmcs	
	prepared by	char(25)	dmcs	
	date prepared	char(8)	dmcs	
CAD Filename ¹	dwg_cadfile	char(25)	local	
Reference Filename ¹			dmcs	no direct table access
Data Set Location ¹			dmcs	no direct table access
CAD Code	dwg_cadcode	char(30)	local	
Org:				
Manager ²		char(30)		not directly implementable

Orgcode ²		char(6)		not directly implementable
Off-Site AE	group id	char(25)		
Company ²		char(4)		not directly implementable
Drawing Requester:				
User ID	group id	char(25)	dmcs	
Password	password	char(8)	dmcs	
Name	group name	char(40)	dmcs	
	process_name	char(80)	dmcs	this is the email id
	group location	char(100)	dmcs	
Date ³				not directly implementable
Time ³				not directly implementable
Authorization:				
Auth Level ⁴				not directly implementable
Project Code ⁴				not directly implementable

Notes:

1. This information will be obtained through facilities available in the DMCS PCI functions where necessary.
2. DMCS does not provide the capability to modify the groups table. This information will probably have to be encoded in the comments field.
3. DMCS does not track user access history information. This information will have to be tracked externally.
4. Handled through the DMCS group relationships facility coupled with the process control facility.

5.3.2 DMCS Setup Definitions

DMCS Setup Definitions

PROJECT DEFINITIONS

Project ID	Description
standard	General CAD Dataset Management System

Project ID: A code to identify a specific project
Description: A description of the project

Menu: CD-PR Reference: Chapter 14 Table: PROJECT_ID

ACCESS REASONS

Access Reason	Description
-----	-----
READ	Read

Access Reason: An access reason code
Description: A description of the the access reason

Menu: CD-AC Reference: Chapter 10 Table: ACCESS_REASON

REVISION REASONS

Revision Code	Description
-----	-----
UPD	Update

Revision Code: A revision reason code
Description: A description of the revision reason

Menu: CD-RE Reference: Chapter 15 Table: REVISION_REASON

LIFE CYCLE STATE CODES

State	Description
-----	-----
RELEASE	The final released state for datasets

State: A life cycle state code
Description: A description of the state code

Menu: CD-ST Reference: Chapter 16 Table: STATE

GROUP DEFINITIONS

Group ID	Password	Name	Location	Phone	Comment	Logon ID
CDMS		Parent Gp.				
CM		Conf. Mgmt.				
CMADMIN		Admin.				
DS		Des. Serv.				
GENERAL		Gen. User				
WHC		WHC				
KEH		KEH				
PNL		PNL				
KEHESSN		KEH Essential				
PNLESSN		PNL Essential				
WHCESSN		WHC Essential				
(not fully defined)						

Group ID: The id used to identify an individual or a group
 Password: The password associated with the group id
 Name: The name of the individual or group
 Location: The physical location of the individual or group
 Phone: The phone number of the individual or group
 Comment: Any additional information regarding the individual or group
 Logon ID: A computer signon id used for receiving email

Menu: OD Reference: Chapter 36 Table: GROUPS

GROUP RELATIONSHIPS

Parent ID	Child Group
CDMS	CM
CDMS	CMADMIN
CDMS	DS
CDMS	GENERAL
CDMS	WHC
CDMS	KEH
CDMS	PNL
CDMS	KEHESSN
CDMS	WHCESSN
CDMS	PNLESSN
(not fully defined)	

Parent ID: Collect and point to the group defn you want to specify as the parent in the relationship
 Child Group: The individual or group that is to be attached as a child to the parent group

Menu: OD-AT Reference: Chapter 36-5 Table: GROUP_MEMBERSHIP

PICTURE DEFINITIONS

Picture ID	Picture String
dwg_no	X(15)
dwg_sheet	9(4)
dwg_title	X(72)
dwg_auth	x(30)
dwg_prj_st	x(1)
dwg_essential	x(1)
dwg_cadcode	X(30)
dwg_comments	x(60)
dwg_current	x(1)
dwg_notify	x(30)
dwg_reldate	99/99/99
dwg_cadfile	x(25)
dwg_rev	9(10)

Picture ID: The key identifier by which DMCS can reference the picture definition
Picture String: The picture by which user prompts will be edited

Menu: IM-PI Reference: Chapter 34 Table: PICTURE

ATTRIBUTE DEFINITIONS

Attribute Name	Picture ID	Attribute Prompt	Description	Edit Type
dwg_no	dwg_no	Drawing No.	Drawing number	-
dwg_sheet	dwg_sheet	Sheet No.	Drawing sheet number	-
dwg_title	dwg_title	Title	Drawing sheet title	-
dwg_auth	dwg_auth	Drawing Authority	Drawing sheet design auth	V
dwg_prj_st	dwg_prj_st	Project Status	Drawing sheet project stat	V
dwg_essential	dwg_essential	Essential	Drawing sheet essential fl	V
dwg_comments	dwg_comments	Comments	Drawing sheet comments	-
dwg_current	dwg_current	Current Drawing	Current Drawing flag	V
dwg_notify	dwg_not who	Notify	Drawing's last modifier	-
dwg_cadfile	dwg_cadfile	CAD Filename	Drawing Filename	-
dwg_cadcode	dwg_cadcode	CAD Code	CAD Code describes pl...	-
dwg_rev	dwg_rev	Revision	Drawing Revision	-
dwg_reldate	dwg_reldate	Rel Date	Date the dwg label issued	-

Attrib. Name: The name associated with the field
Picture ID: The picture identifier to be used with the picture attrib.
Attrib. Prompt: The data entry prompt to be displayed to the user on the form
Description: Descriptive text which explains the usage of the attrib.
Edit Type: This field specifies the type of legal value set that is attached to this attrib. Valid entries are: R (Range), V(Value), D(Date - PAST, PRESENT or FUTURE) and - (none)
Menu: IM-AT Reference: Chapter 31 Table: ATRIB

LEGAL VALUES

LV Subset Name	Qualifier Name	Qualifier Field Value	Value	Description
Flags	ce_type	Drawing	blank	no flag
True			+	true
Companies	ce_type	Drawing		
Westinghouse			WHC	
Kaiser			KEH	
Battelle			PNL	

Subset Name: The name given to this list of legal values.

Qualifier Name: This list is further qualified by:

blank (no criteria)

'userid' (lower case) - if this list is for a specific user.

'ce type' (lower case) - if this list is for a specific CE type.

Qual. Fld. Val: Blank if 'blank' in Qualifier Name.

A valid user id if 'userid' was used.

A valid CE type if 'ce type' was used.

Value: A valid value which will be used for validation in data entry.

Description: A description of the legal value.

Menu: IM-AT-LE/LV

Reference: Chapter 31-13,17

Table: LEGAL_SUBSET

CONTROLLED ENTITY DEFINITIONS

Name	Class	Description
Drawing	C	Drawing entity

Name: The name of the entity label to be defined.

Class: The type of entity label to be defined. Valid entries are:
C (Controlled), R(Relational) and H(Hierarchical).

Description: A description of the entity label.

Menu: IM-EN

Reference: Chapter 32

Table: CE_TYPE

CONTROLLED ENTITY COMPONENT DEFINITIONS

CE type	Entity Component Name	Seqno	Field Usage
Drawing	dwg_no	1	key
	dwg_sheet	2	key
	dwg_title	3	
	dwg_auth	5	rule
	dwg_prj_st	6	rule
	dwg_essential	7	rule
	dwg_comments	13	
	dwg_current	14	rule
	dwg_notify	15	
	dwg_reldate	20	
	dwg_cadfile	17	
	dwg_cadcode	18	
	dwg_rev	19	

CE type: The entity being formed.

Comp. Name: The attrib. name defined in IM-AT that is being linked to the controlled entity structure.

Seqno: The relative position within the record in which the attrib. is to be located.

Field Usage: The usage of the field within the controlled entity label.
Valid entries are: KEY, RULE and DESCRIPTION.
Note: If the attrib. is a key and a rule field, two entries are required.

Menu: IM-EN-CE/EX Reference: Chapter 32-11,14 Table: ???

PROCESS CONTROL RULES

CE type	Group ID	Menu Functions						Pr Ty	In	Project ID	Current State	Desired State	
		1	2	3	4	5	6						
DRAWING	CDMS		LA	FO	CO	*	*	*	+	+	*	RELEASE	*
	CDMS		LA	FO	QY	*	*	*	+	+	*	RELEASE	*
	CDMS		LA	QY	*	*	*	*	+	+	*	RELEASE	*
	CDMS		VW	*	*	*	*	*	+	+	*	RELEASE	*
	CM		LA	FO	AO	*	*	*	+	+	*	NULL	*
	CM		LA	FO	CA	*	*	*	+	+	*	NULL	*
	CM		LA	FO	PU	*	*	*	+	+	*	NULL	*
	CM		LA	FO	QY	*	*	*	+	+	*	NULL	*
	CM		LA	IS	*	*	*	*	+	+	*	NULL	*
	CM		LA	QY	*	*	*	*	+	+	*	NULL	*
	CM		LA	ST	IS	*	*	*	+	+	*	NULL	*
	CM		RE	*	*	*	*	*	+	+	*	NULL	*
	KEH		LA	AD	*	*	*	*	+	+	*	NULL	*
	KEH		LA	CA	*	*	*	*	+	+	*	NULL	*
	KEH		LA	CR	*	*	*	*	+	+	*	RELEASE	*
	KEH		LA	DE	*	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	CA	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	CI	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	CO	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	DF	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	MO	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	PU	*	*	*	+	+	*	NULL	*
	KEH		LA	FO	QY	*	*	*	+	+	*	NULL	*
	KEH		LA	QY	*	*	*	*	+	+	*	NULL	*
	KEH		LA	UP	*	*	*	*	+	+	*	NULL	*
	KEH		RE	*	*	*	*	*	+	+	*	NULL	*
	KEHESSN		LA	CR	*	*	*	*	+	+	*	RELEASE	*

(KEH entries have design authority rule field set to KEH)

(repeat KEH entries for other companies: WHC, PNL)

CE type: The controlled entity for which the process control rule applies.
 Group ID: The id of the group for which the process control rule applies.
 Menu Fcn 1-6: The menu function path which describes the rule. '*' is wild.
 Process Type: '+' gives the group the process control rule.
 Inherited: '-' denies the group the process control rule.
 Project ID: '+' allows the rule to be inherited by the group's children.
 Current State: '-' denies any inheritance.
 Project ID: The project id for which this rule applies.
 Current State: The code to identify the current life cycle state of the CE.
 Desired State: The code to identify the state the CE would move to if a state change request was approved.

Menu: CO-PC

Reference: Chapter 24

Table: AUTHORITIESxx

MENU ACCESS RULES

Group ID	Menu Functions						Pr Ty	In
	1	2	3	4	5	6		
*	DL	AS	PM	*	*	*	-	+
*	DL	CS	PM	*	*	*	-	+
*	DL	DT	PM	*	*	*	-	+
CDMS	CH	*	*	*	*	*	+	+
CDMS	LA	FO	CO	*	*	*	+	+
CDMS	LA	FO	QY	*	*	*	+	+
CDMS	LA	QY	*	*	*	*	+	+
CDMS	LO	*	*	*	*	*	+	+
CDMS	ME	*	*	*	*	*	+	+
CDMS	VW	*	*	*	*	*	+	+
CM	LA	FO	AO	*	*	*	+	+
CM	LA	FO	CA	*	*	*	+	+
CM	LA	FO	PU	*	*	*	+	+
CM	LA	FO	QY	*	*	*	+	+
CM	LA	IS	*	*	*	*	+	+
CM	LA	QY	*	*	*	*	+	+
CM	LA	ST	IS	*	*	*	+	+
CM	LA	UP	*	*	*	*	+	+
CM	RE	*	*	*	*	*	+	+
CM	VW	*	*	*	*	*	+	+
CMADMIN	OD	*	*	*	*	*	+	+
DS	LA	FO	CO	*	*	*	+	+
DS	LA	FO	QY	*	*	*	+	+
DS	LA	QY	*	*	*	*	+	+
DS	RE	*	*	*	*	*	+	+
KEH	LA	AD	*	*	*	*	+	+
KEH	LA	CA	*	*	*	*	+	+
KEH	LA	CR	*	*	*	*	+	+
KEH	LA	DE	*	*	*	*	+	+
KEH	LA	FO	CA	*	*	*	+	+
KEH	LA	FO	CI	*	*	*	+	+
KEH	LA	FO	CO	*	*	*	+	+
KEH	LA	FO	DF	*	*	*	+	+
KEH	LA	FO	MO	*	*	*	+	+
KEH	LA	FO	PU	*	*	*	+	+
KEH	LA	FO	QY	*	*	*	+	+
KEH	LA	QY	*	*	*	*	+	+
KEH	LA	UP	*	*	*	*	+	+
KEH	RE	*	*	*	*	*	+	+
KEHESSN	LA	CR	*	*	*	*	+	+
(repeat for other design authorities: PNL,WHC)								
SUPER_USER	*	*	*	*	*	*	+	+
SUPER_USER	DD	*	*	*	*	*	-	+
SUPER_USER	RI	*	*	*	*	*	-	+
SUPER_USER	SY	*	*	*	*	*	-	+

Group ID: The id of the group for which the menu access rule applies.
Menu Fcn 1-6: The menu function path which describes the rule. '*' is wild.
Process Type: '+' gives the group access to this menu fcn.
Process Type: '-' denies the group access to this menu fcn.

Inherited: '+' allows the rule to be inherited by the group's children.
'-' denies any inheritance.

Menu: CO-ME Reference: Chapter 22 Table: MENU_ACCESS

APPROVER RULES

CE type	Group ID	Tab Method	Pr Ty	In	Project ID	Current State	Desired State	Orig By	Set #	Reqd
(none)										

CE type: The controlled entity for which the rules apply.
 Group ID: The id of the group for which the rules apply.
 Tab Method: The code to determine the method of counting votes. Valid entries are: ANY, ANYALL, AUTOREQ, EVERY, EVERYALL, MAJMINUS and MAJPLUS.
 Process Type: '+' gives the group the voting privilege.
 '-' denies the group the voting privilege.
 Inherited: '+' allows the rule to be inherited by the group's children.
 '-' denies any inheritance.
 Project ID: The project id for which this rule applies.
 Current State: The code to identify the current life cycle state of the CE.
 Desired State: The code to identify the state the CE would move to if a state change request was approved.
 Orig by: The id of the user that will initiate the approval process in order for this approval rule to be in effect. '*' is wild.
 Set #: A positive number used for identifying serial approval sets. Lowest numbered sets vote first.
 Reqd: '+' This user's vote is required.
 '-' This user's vote is not required.

Menu: CO-AP

Reference: Chapter 19

Table: APPROVERSxx

LIFE CYCLE RULES

CE type	Project ID	Pr Ty	Current State	Desired State	Reject State
Drawing	*	+	NULL	RELEASE	NULL

CE type: The controlled entity for which the rules apply.
 Project ID: The project id for which this rule applies.
 Process Type: '+' This rule is used to determine the next state.
 '-' This rule defines an exception to the rule.
 Current State: The code to identify the current life cycle state of the CE.
 Desired State: The code to identify the state the CE would move to if a state change request was approved.
 Reject State: The code to identify the state the CE would move to if a state change request is denied.

Menu: CO-LF

Reference: Chapter 21

Table: LIFECYCLE_STATExx

NOTIFICATION RULES

CE type	Event Type	Not Lvl	Pr Ty	In	Project ID	Current State	Desired State	Reject State	Orig. By
(none)									

CE type: The controlled entity for which the rules apply.
 Event Type: The id of the group to be notified.
 Not Lvl: 'X' indicates that the rule applies to the specified CE type.
 Process Type: '+' indicates the group is to be notified.
 '- ' indicates the group is not to be notified.
 Inherited: '+' allows the rule to be inherited by the group's children.
 '- ' denies any inheritance.
 Project ID: The project id for which this rule applies.
 Current State: The code to identify the current life cycle state of the CE.
 Desired State: The code to identify the state the CE would move to if a state change request was approved.
 Reject State: To Be Determined.
 Orig. by: The id of the user that initiates the event in order for this notification message to be sent out. '*' is wild.

Menu: CO-NO Reference: Chapter 23 Table: NOTIFICATIONSxx

6.0 REFERENCE DOCUMENTS

- WHC CM-4-2 - Quality Assurance
- WHC CM-3-10 - Software Practices
- WHC CM-2-6 - Data Administration Standards
- WHC-SD-GN-CDR-30002 - CDMS Concept Document
- WHC-SD-GN-PMP-30003 - CDMS System Project Management Plan
- WHC-SD-GN-CSRS-30005 - CDMS System Requirements Specification

STRUCTURAL DYNAMICS RESEARCH CORPORATION (SDRC) MANUALS

- P-0628 - DMCS Planning, Implementation and Customization Guide
- P-0632 - DMCS Software Installation and System Administrator's Guide

APPENDIX A CDMS Hardware Environment **CDMS Hardware Description**

CDMS1: Sun SPARCstation 10 model 40, 96 MB RAM, internal 400 MB hard disk, external 1GB hard disk, SunOS 4.1.3.

This workstation will have the DMCS software loaded on it and will provide all DMCS services.

CDMS2: Sun SPARCstation 10 model 40, 96 MB RAM, internal 400 MB hard disk, external 1GB hard disk, 5GB 8mm tape drive, SunOS 4.1.3.

This workstation will controll the mass storage system (FLEXSTOR) and provide a backup for CDMS1 if hardware problems develop.

FLEXSTOR: The mass storage system consisting of an HP model 40T optical disk library with two 1.3 GB HP optical drives and 30 1.3 GB read/write optical disks. The driving software will be loaded on CDMS2.

CDMFS: A dedicated NT file server used to facilitate drawing file transfers from DOS to UNIX.

APPENDIX B CDMS LAN INTERFACES

CDMS IN THE LAN TOPOLOGY

CDMS1 provides all of the DMCS services. Communications with CDMS1 will be through the Unix sockets interface. This interface is provide by the Windows Sockets library (winsock.dll) for DOS computers. The CDMS interface can also be accessed through Telnet.

The data files will be stored on the FLEXSTOR system which is controlled by CDMS2. The only computer accessing CDMS2 in any way will be CDMS1 which will initiate file transfers between CDMS2 and itself or between CDMS2 and other computers that provide FTP (File Transfer Protocol) services.

CDMS DATA PATHS

The CDMFS file server will provide interim storage for data files coming from or going to DOS computers. Access from CDMS1 or CDMS2 to CDMFS will be through an FTP server running on CDMFS. The CDMFS server will be using the NT operating system.

(See CDMS Data Paths diagram on next page)

APPENDIX B (Cont'd)

APPENDIX C GUI/PCI INTERFACE

The GUI/PCI interface field usage is shown in the table below.
 R=Read, W=Write/update, Q=Query. Fields marked with * are sent positional and not by name.

Field	Login Form	Search Form	Query Result	Drawing List	Drawing Form	Submit Form
dwg no		Q		R	R	W
dwg sheet		Q		R	R	W
dwg rev				R	R	W
dwg title					R	W
dwg cadfile					R	W
dwg comments					R	W
dwg auth		Q			R	W
dwg notify					R	W
dwg cadcode					R	W
dwg prj st					R	W
dwg essential					R	W
project id		Q			R	W
approval date					R	
from application file					R	W
file description					R	W
owner id					R	
auth group	R					
default aspace	R					
sh1	R					
sh2	R					
sh3	R					
*User id	Q					
*Password	Q					
*Record number						

APPENDIX D UNIX System and DMCS Customization

Directories

Several important directories were created and/or used during development. Each of the following directories can be found in the /home/dmcs directory and a discussion of each follows.

control/

The control directory contains subdirectories that will be used by the test/training and development knowledge bases as their "control space". This is the main repository of the files that DMCS manages.

dmcs_support/

The dmcs support directory contains information and source files received from SDRC's DMCS Support Center.

exe/

The exe directory holds the DMCS executables that have been rebuilt. They should eventually be moved to the version directory, replacing the executables from the original DMCS distribution.

notes/

The notes directory contains files that are notes about or instructions on using DMCS.

obj/stubs/

The obj/stubs directory contains object files that are added to the DMCS libuser.a library file which can then be linked to the DMCS executables. This directory contains some files from the DMCS distribution that have been modified, as well as, some custom source files that were created to be used by the DMCS process graph (see the Transaction Modeling Chapter in the DMCS Planning, Implementation and Customization Guide for more information).

The custom files will be discussed individually later in this appendix.

This directory also contains a makefile to be used in rebuilding the DMCS libuser.a library and executables. Any time a file in this directory is modified, the 'make' program should be invoked from this directory. To rebuild the executables, type: make exe.

For example, if the usermcpa.c file were modified, the developer should type the following:

```
% make          <= this rebuilds the libuser.a library
% make exe      <= rebuilds the DMCS executables
```

If new source files are created in this directory, they must be added to the makefile. Also, if the filename is added to proto.files, the functions in the new source file will be added to proto.h which is included in most of the other source files.

The script 'makeproto' in this directory makes the new proto.h file.

sql/

The sql directory contains useful scripts that can be invoked by simply typing @scriptname from the sql command line. This directory is included in the SQLPATH.

src/cdmspci

This directory contains all the source for the cdmspci program. This program provides the external interface to DMCS to other programs that connect to the CDMS system.

The makefile will rebuild cdmspci if any file has changed. Once built, it can be tested from this directory. To make the new executable available to the external process, type:

```
% make install
```

If new source files are added to cdmspci, the makefile must be edited to add these files. Also, if the filename is added to proto.files, the functions in the new source file will be added to proto.h which is included in most of the other source files.

The script 'makeproto' in this directory makes the new proto.h file.

src/cdmsrv/

The src/cdmsrv directory contains the source for a daemon that could be used to monitor the CDMS port in place of inetd.

src/class/

The src/class directory contains source created during the Advanced DMCS Customization class and can be used for examples.

tbl/

The tbl directory contains all the files used to build the DMCS process graph. Several of these files have been modified and their modifications will be discussed later in this appendix.

inetd

The Unix inetd service is used to monitor a set port for CDMS. Connections to this port start up the cdmspci program which allows batch and interactive processes to communicate with CDMS. The following denotes the files modified and the lines added to each:

```
/etc/inetd.conf
cdmsrv stream tcp nowait cdmuser /home/cdmuser/cdmsd cdmsd
```

```
/etc/services
cdmsrv          5000/tcp
```


scripts

Several shell scripts (both Bourne and C-shell) were written and/or to perform special tasks for CDMS or aid in its development. They reside in the following directories, as indicated, and are described below:

/home/dmcs/bin

::::::::::::

backup

::::::::::::

#!/bin/sh

#

backup

#

DM Brown

6 March 1991

#

Create a backup of the current user's directory. Find is used to limit which files are backed up. Tar is used to create the backup file, which can then be copied to some backup media or a backup host. A script, 'backcp' is checked for in \$HOME/bin or /usr/local/bin. If it exists, it is called with the name of the backup file passed to it as an argument.

#

The script uses the current date and day of the week to determine what type of backup to perform. A daily backup is performed each day of the week, except Fridays, when the weekly backup is performed instead. On the last Friday of the month a monthly backup is performed.

#

The monthly backup performs a complete backup of the user's home directory (except for files and directories find has been told to ignore). The weekly backs up back to the last weekly or monthly and the daily backs up to the last backup performed (daily, weekly or monthly).

#

The first time the script is run, it checks to see if the \$HOME/.backup directory exists. If it doesn't exist, it is created and the last.daily and last.weekly files created. The script then defaults to running a monthly backup, regardless of the day of the week or date.

#

It can be placed in a user's crontab to be executed automatically on a set schedule. For example:

#

30 23 * * * fullpathname/backup

#

::::::::::::

backup_db

::::::::::::

#!/bin/sh

backup_db

DM Brown

13 September 1994

```

#
# export the cdms database (knowledge base) to a file to be
# backed up with the daily backup.

::::::::::::
dbg_dmcs
::::::::::::
#!/bin/csh
# Starts up DMCS in debugging mode. Extra information is
# written to the dmcs.trc file.

::::::::::::
mkexe
::::::::::::
#!/bin/sh
This is a standard DMCS script file for rebuilding DMCS executables. It was
modified to include a library of ftp routines.

::::::::::::
proc
::::::::::::
#!/bin/sh
This is a standard DMCS script file for building PRO*C programs. It was
modified to correct a problem in its success/fail logic.

::::::::::::
reportb
::::::::::::
#!/bin/sh
This is a copy of a standard DMCS script for creating reports from the Oracle
database. It was modified to allow it to run in a batch mode. If run as user
dmcs, it will use the oradcd program to discover the Oracle username/password.

::::::::::::
runit
::::::::::::
#!/bin/csh
# DM Brown
# 20 Sep 1994
#
# Establish csh environment for dmcs scripts that need it,
# then run them. (scripts like reports/nightly.sh and
# bin/backup_db)

::::::::::::
sdmcs_xt
::::::::::::
#!/bin/csh
Start up the DMCS TUI in an Xterm window.

::::::::::::
stopdmcs
::::::::::::
#!/bin/sh
#
# stopdmcs

```

```
# DM Brown
# 17 March 1994
#
# shutdown all processes associated with dmcs: oracle, sqlnet,
# sched

:::::::::::::
stpsched_d
:::::::::::::
#!/bin/sh
This is a copy of a standard DMCS script that is used to stop the dmcs
scheduler. It has been modified to enable it to run in a batch mode.
```

```
/home/cdmuser
```

```
:::::::::::::
cdmsd
:::::::::::::
#!/bin/csh
# cdmsd
# DM Brown
# 23 May 1994
#
# start up cdms through inetd. Without an argument, it
# starts up the cdms KB. With 'train' or 'dev' as an
# argument, it will start up the cdmstrain or cdmsdev
# KB.
#
```

```
/home/dmcs/reports
```

```
:::::::::::::
nightly.sh
:::::::::::::
#!/bin/sh
# nightly.sh
#
# DM Brown
# 9 Sept 1994
#
# Script to run reports on a scheduled basis through cron.
# The file nightly.list holds a list of the reports that wil
# be created. The script 'distr_reports.sh' will distribute
# them if it exists.
```

oracle customization

The table 'aspace sharename' has been added to the cdms Oracle tablespace. This table is defined as follows and holds information for the DOS/Windows User Interface to be able to relate the DMCS defined application space to the DOS sharename required to make a connection to the file server directory.

```
SQL> describe aspace_sharename
Name                               Null?    Type
-----
```

TABLE_NAME	NOT NULL	CHAR(30)
LOCATION_NAME	NOT NULL	CHAR(100)
LOCATION_TYPE		CHAR(1)
SHARE_NAME	NOT NULL	CHAR(25)
SHARE_PW		CHAR(8)
SHARE_SUBDIR		CHAR(100)

An SQL script, /home/dmcs/sql/mk_local_tables.sql, has been established to re-create this table, if necessary.

Knowledge Bases

cdms: The main knowledge base for the CDMS.

cdmstrain: Used for training and test.

cdmsdev: Used for future development.

cron

The Unix cron facility is used to perform periodic tasks on a regular basis. The following cron entries have been made for the dmcs user and will perform the required system backups and create reports.

```
00 23 * * * $HOME/bin/backup
00 21 * * * $HOME/bin/runit reports/nightly.sh >/dev/null
00 22 * * * $HOME/bin/runit bin/backup_db 2>&1 >/dev/null
```

Miscellaneous Files Modified

/home/dmcs/tbl/

cltn_sort_list.dat: this file tells dmcs how to sort queries.

dmcs_tbls.<kbname>: these files set up the DMCS internal environment for each knowledge base.

Modifications to the Process Graph

The DMCS process graph is modified through files in the /home/dmcs/tbl directory. The appropriate mcpbin file must then be rebuilt. For a full explanation of this process, see the Transaction Modeling Chapter in the DMCS Planning, Implementation and Customization Guide.

Once any of the mcpt*.tbl files in this directory are modified, the process graph will need to be rebuilt with the 'mkcdms' script. This script will create the file mcpbin.cdmsdev. Once testing is complete, this file should be copied to mcpbin.cdms and a 'share' performed to load it into shared memory. For example:

```
% mkcdms
% cp mcpbin.cdmsdev mcpbin.cdms
% share 2
```

The following are the files that were added or modified.

```
:::::::::::::::
mcpt01.tbl
```

.....

Added code to maintain a usage log.

.....

mcpt05.tbl

.....

Added form field initializations for fields specific to the CDMS drawing entity.

.....

mcpt10.tbl

.....

Added code to maintain the dwg_rev field. This field is incremented when a create revision is performed.

.....

mcpt21.tbl

.....

Added crc check functions beffore and after file xfer (a244) calls.

.....

mcpt23.tbl

.....

Added code that ensures the user will only retrieve current records. Previous revisions of drawing labels are not retrieved in the query. This is done by setting a database flag in the labels01 table, dwg current, for the label that is the current released revision, and resetting this flag when the label is replaced with a new revision.

.....

mcpsub.tbl

.....

This file was modified to fix an oversight (bug) from SDRC. This oversight didn't allow a PCI program to be able to create a revision of a label.

.....

udwg.tbl

.....

/*

DRAWING NULL label transactions

DMB 30 Dec 1993

Allow the user to query only on NULL/RELEASE state drawings.

*/

.....

uprocess.tbl

.....

Added include for udwg.tbl.

```

:
:
ustarts.tbl
:
:

```

Added entries for udwg.tbl transaction entry points. These entry points are accessed through the MAIN.RE and MAIN.VW menu picks in the DMCS TUI.

Source Files in obj/stubs

The following headers were extracted out of the source files that were added or modified to add functionality to the process graph. These files are in /home/dmcs/obj/stubs and are added to libuser.a.

```

:
:
checkcrc.c
:
:

```

```

/* checkcrc.c
   DM Brown
   26 July 1994

```

This file includes all the functions used in the process graph to generate and compare crc's.

Compile with:

```

mkhere checkcrc.c
mk_stub_lib
mkexe dmcs
mkexe server
mkexe tcpsrvr
mkexe -X dmcs (if you have motif libraries)

```

*/

```

:
:
ftp_crc.c
:
:

```

```

/* ftp_crc.c
   DM Brown
   26 July 1994

```

Generate a crc on a remote file. Portions taken from crc16.c and fcp.c. Link with libftp.a.

*/

```

:
:
inc_numfield.c
:
:

```

```

/* inc_numfield.c
   DM Brown
   14 July 1994

```

Increment a numeric field by one. This function is called from usermcpa.c and is intended for use with the dwg_rev field in cdms.

Compile with:

```
mkhere inc numfield.c
mk_stub lib
mkexe dmcs
mkexe server
mkexe tcpsrvr
mkexe -X dmcs (if you have motif libraries)
```

*/

locations.h

locations.h

```
/* locations.h
   DM Brown
   25 July 1994
```

Structures and definitions for working with dmcs location information.

*/

locations.pc

locations.pc

```
/* locations.pc
   DM Brown
   25 July 1994
```

Functions for working with dmcs file location information.

*/

my_msgs.c

my_msgs.c

```
/* my_msgs.c
   DM BROWN
   18 July 1994
```

provide some special message functions for a100 routines.

*/

myora.h

myora.h

```
/* myora.h
   DM Brown
   19 May 1994 (original 12 December 1989)
```

macros and definitions I find useful when working with oracle pro*c.

*/

ora_pc.h

ora_pc.h

```
/* ora_pc.h
   DM Brown
```

31 May 1994

Include file for all the pro*c stuff

*/

:::::::::::::

oradcd.c

:::::::::::::

/* get oradcd.c

DM Brown

15 September 1994

get the oracle userid and password from oradcd.
return a pointer to a static buffer with the uid/passwd string
or NULL.

Compile with:

mkhere get oradcd.c

mk_stub lib

mkexe dmcs

mkexe server

mkexe tcpsrvr

mkexe -X dmcs (if you have motif libraries)

*/

:::::::::::::

oralogon.pc

:::::::::::::

/* oralogon.pc

DM Brown

1 February 1990

provide standard logon and error reporting functions.

15 Sep 1994

added oradcd() to get the userid/passwd for the current KB.

*/

:::::::::::::

proto.h

:::::::::::::

/* proto.h

DM Brown

15 Sep 94

This file is a list of prototypes gleaned from the source
files listed in proto.files. It was automatically generated
by the 'makeproto' script.

*/

:::::::::::::

savedval.c

:::::::::::::

/* savedval.c

DM Brown

14 July 1994

Created from my original modifications to usermcpa.c.

These functions save a value, then allow that value to be assigned to a field in dmcs. They are called from al00() through usermcpa.c. They were written to facilitate saving a value to be copied into the current state field during a LA-QY transaction. This will allow the user's query to be held to a specific state depending on the actions they are performing.

Compile with:

```
cc -c -DUNIX usermcpa.c -I/home/dmcs/include/dmcs
or mkhere usermcpa.c
mk_stub_lib
mkexe dmcs
mkexe server
mkexe tcpsrvr
mkexe -X dmcs    (if you have motif libraries)
```

*/

```
:::::::::::::
storecrc.pc
:::::::::::::
/* storecrc.pc
   DM Brown
   1 June 1994
```

Template for pro*c code.

*/

```
:::::::::::::
stubmail.c
:::::::::::::
/* stubmail.c
```

Taken from unixmail.example, see that file for copyright, etc.

To use this new routine place yourself in the obj/stubs directory and do the following:

- 1) mkhere stubmail.c (to build new object)
- 2) mk_stub lib (to insert new object into user lib.)
- 3) mkexeall (rebuild all executables with new

lib.)

*/

```
:::::::::::::
upd_current.c
:::::::::::::
/* upd_current.c
   DM Brown
   18 July 1994
```

update the dwg_current field and set it to '-'. To be called during a label issue, to reset the current drawing flag. This flag is used

to present only the latest released version of a dataset to the user.

Compile with:

```
mkhere upd_current.c
mk_stub_lib
mkexe dmcs
mkexe server
mkexe tcpsrvr
mkexe -X dmcs    (if you have motif libraries)
```

*/

```
:::::::::::::::
upd_dwg_cur.pc
:::::::::::::
/* upd_dwg_cur.pc
   DM Brown
   18 July 1994
```

Pro*c code to support modifying the dwg_current flag on the previous revision of the label.

*/

```
:::::::::::::::
userlog.c
:::::::::::::
/* userlog.c
   DM Brown
   2 August 1994
```

function to log user information to \$DMCS_LOG/userlog. This function is called from a100 passing the second argument (string) to userlog(). The date, time,uid, and string are logged to the file.

Compile with:

```
mkhere userlog.c
mk_stub_lib
mkexe dmcs
mkexe server
mkexe tcpsrvr
mkexe -X dmcs    (if you have motif libraries)
```

*/

```
:::::::::::::::
usermcpa.c
:::::::::::::
/*
 * FILENAME:    usermcpa.c
 *
 * PURPOSE:     This function is set up to call user written functions
 *
 * MODULES:
 *
 *      user_mcp_action - execute an action based on action codes
 */
```

usermcpa.c is the file that provides C programming hooks into the DMCS process graph. For more information, see the Transaction Modeling chapter in the DMCS Planning, Implementation, and Customization Guide.

Source for CDMSPCI

The following headers were extracted out of the source files for the cdmspci program. These source files are in the src/cdmspci directory.

```

::::::::::::
build_label.c
::::::::::::
/* build_label.c
   DM Brown
   1 April 1994

   parse fin looking for records of the form:
       (
         at_name : at_value
         :
       )
   and put into the supplied ufls.
   This routine checks the attribute against the list of
   attributes maintained by dmcs. the functions check_atrib()
   is used.
*/

::::::::::::
cancel_owner.c
::::::::::::
/* cancel_owner.c
   DM Brown
   29 May 1994

   cancel ownership of the indicated file item to the name passed
   in the new_owner parameter. if new_owner = "", ownership is
   cancelled.
*/

::::::::::::
caown.c
::::::::::::
/* caown.c
   DM Brown
   19 May 1994

   cancel the user's ownership on files in the indicated
   label from the current query.
*/

::::::::::::
carv.c
::::::::::::
/* carv.c
   DM Brown
   19 May 1994

```

cancel the revision of the indicated label from the current query.

*/

::::::::::

cdms_cleanup.c

::::::::::

/* cdms_cleanup.c

DM Brown

31 March 1994

All cleanup routines should be called from here. This routine will be called before logging off from dmcs.

*/

::::::::::

cdms_commands.h

::::::::::

/* commands.h

DM Brown

Command definitions for interpreter.

*/

::::::::::

cdmspci.c

::::::::::

/* cdmspci.c

DM Brown

Main portion of cdmspci.

*/

::::::::::

cdmspci.h

::::::::::

/* cdmspci.h

DM Brown

25 March 1994

Standard definitions and externals for cdmspci.

*/

::::::::::

cdmsstd.h

::::::::::

/* cdmsstd.h

DM Brown

25 March 1994

Standard include files for cdmspci.

*/

::::::::::

change_owner.c

::::::::::

```
/* change_owner.c
   DM Brown
   29 May 1994
```

assign ownership of the indicated file item to the name passed in the new_owner parameter. if new_owner = "", ownership is cancelled. If the cancel flag is set, this function invokes the LA.FO.CA process graph transactions instead of the LA.FO.AO.

```
*/
```

```
:::::::::::::::
check_atrib.c
```

```
:::::::::::::::
```

```
/* check_atrib.c
   DM Brown
   31 March 1994
```

Get a list of attributes for the indicated entity from the database and check that the named attribute is on the list.

```
*/
```

```
:::::::::::::::
check_auth.pc
```

```
:::::::::::::::
```

```
/* check_auth.pc
   DM Brown
   1 July 1994
```

A function to check a user's menu authorization in the CDMS System.

```
*/
```

```
:::::::::::::::
chown.c
```

```
:::::::::::::::
```

```
/* chown.c
   DM Brown
   19 May 1994
```

cancel the revision of the indicated label from the current query.

```
*/
```

```
:::::::::::::::
chpw.c
```

```
:::::::::::::::
```

```
/* chpw.c
   DM Brown
   22 March 1994
```

This routine allows the users to change their password.

```
*/
```

```
:::::::::::::::
display_label.c
```

```
.....  
/*  display_label.c  
    DM Brown  
    29 April 1994
```

Display the label information passed in the ufcb in the format:

```
(
  at_name: at_value
  :
  :
)
```

*** /**

```

.....
env.c
.....
/* env.c
        DM Brown
        28 March 1994

```

Provide the ability to store and retrieve values, similar to an environment variable.

*** /**

```

.:.:.:.:.:.:.:.:.:.:
env.h
.:.:.:.:.:.:.:.:.:.:
/*  env.h
        DM Brown
        28 March 1994

```

prototypes for the environment functions.

*** /**

```

::::::::::::
foci.c
::::::::::::
/* foci.c
    DM Brown
    26 April 1994

```

This function gets the information required to copy a file in to the dmcs system from the user's application space. the replace flag is used to determine if an existing file is to be replaced in the folder.

*** /**

```

::::::::::::
foco.c
::::::::::::
/*  foco.c
    DM Brown
    25 April 1994

```

This function gets the information required to copy a file out of the dmcs system to the user's application space. the mod flag is

used to determine if the file is to be checked out for modification of just for reference.

*/

:::::::::::::
fode.c

:::::::::::::

/* fode.c
DM Brown
3 August 1994

delete a folder item.

*/

:::::::::::::
folder_copyin.c

:::::::::::::

/* folder_copyin.c
DM Brown
26 April 1994

Initiate the copy in a file from the application file space to dmcs.

This function requires that the queryitem and folder item be identified and that the application space and source file name be given. The replace_flag determines whether an existing file in the folder will be replaced.

If the replace flag is not set, fitem is not required, but the description is. If it is set, the opposite is true.

*/

:::::::::::::
folder_copyout.c

:::::::::::::

/* folder_copyout.c
DM Brown
25 April 1994

Initiate the copy of a file out of dmcs to the application file space for the user program to pick up.

This function requires that the queryitem and folder item be identified and that the application space and destination name be given. The mod_flag determines whether the modify or copyout function will be called.

*/

:::::::::::::
folder_delete.c

:::::::::::::

/* folder_delete.c
DM Brown
3 August 1994

delete/purge the indicate folder item from the folder.

*/

```

:::::::::::::
folder_query.c
:::::::::::::
/* folder_query.c
   DM Brown

```

collect the folder items for the indicated label.

*/

```

:::::::::::::
fopu.c
:::::::::::::
/* fopu.c
   DM Brown
   5 August 1994

```

purge a folder item. This operation deletes all folder items that are older than the one indicated.

*/

```

:::::::::::::
foqy.c
:::::::::::::
/* foqy.c
   DM Brown
   11 April 1994

```

Cause folder_query() to get folder information for the desired label.

*/

```

:::::::::::::
get_ent_ufc.c
:::::::::::::
/* get_ent_ufc.c
   DM Brown
   31 March 1994

```

build a ufc with fields retrieved from dmcs for the desired entity. This ufc can be used for comparison while building other related ufc's on the fly, or as a starting place.

*/

```

:::::::::::::
get_folder.c
:::::::::::::
/* get_folder.c
   DM Brown
   18 April 1994

```

Get the folder item metadata from a previous query.

*/

• • • • •

• • • • •

```
get_label.c
```

• • • • •

```
/* get label.c
```

DM Brown

4 April 1994

Get a label from a previous query.

*** /**

```
get_uauth.pc
```

• • • • •

```
/* get uauth.pc
```

DM Brown

2 June 1994

This routine displays all the groups the indicated user belongs to. This is an indication of the authorizations the user has. There must be a better way to do this, perhaps another routine which interprets the authorization, but that would be fairly non-portable.

*/

A decorative graphic consisting of two rows of dots. The top row has 14 dots and the bottom row has 13 dots, arranged in a staggered pattern.

```
get uinf.pc
```

• • • • •

```
/* get uinf.pc
```

D $\bar{M}$ Brown

13 June 1994

build a ufcb of information about the user passed to the routine.

***/**

• • • • •

help.c

• • • • •

```
/* help.c
```

DM Brown

15 June 1994

```

, search the help file for a section beginning with the desired
topic.  Dump this section to the output.

```

*/

• • • • •
• • • • •

```
interpret.c
```

• • • • •

```
/* interpret.c
```

DM Brown

Interpret the command line.

*/

• • • • •

```

is_aspace.pc
::::::::::::
/* is_aspace.pc
   DM Brown
   1 June 1994

   returns TRUE if the indicated name is a valid location_name in dmcs,
   otherwise returns FALSE.
*/

::::::::::::
is_user.pc
::::::::::::
/* is_user.pc
   DM Brown
   1 June 1994

   returns TRUE if the indicated name is a valid group_id in dmcs,
   otherwise returns FALSE.
*/

::::::::::::
laad.c
::::::::::::
/* SCCS ID @(#)laad.c  6500.1 11/15/93  */
/*
 * File:      laad
 *
 * This program will create a ufls, populate the ufls
 * by reading from the file handle passed in, and do a LA.AD
 *
 * Changed to build_label().
*/

::::::::::::
label_add.c
::::::::::::
/* label_add.c
   DM Brown
   29 March 1994

   perform label add and get collection name
*/

::::::::::::
label_owner.pc
::::::::::::
/* label_owner.pc
   DM Brown
   3 August 1994

   query the database for label info where a user has
   objects checked out of the folder.
*/

::::::::::::

```

```
label_query.c
::::::::::::
/* label_query.c
   DM Brown
   25 March 1994

   perform label query and get collection name
*/

::::::::::::
label_update.c
::::::::::::
/* label_update.c
   DM Brown
   4 May 1994

   perform label update
*/

::::::::::::
lacr.c
::::::::::::
/* lacr.c
   DM Brown
   5 May 1994

   creates a revision of the indicated label from the current
   query.
*/

::::::::::::
lais.c
::::::::::::
/* lais.c
   DM Brown
   5 May 1994

   performs a label issue on the indicated label from
   the current query.
*/

::::::::::::
laown.c
::::::::::::
/* laown.c
   DM Brown
   3 August 1994

   list the labels the user has objects checked out of the
   folder for.
*/

::::::::::::
laqy.c
::::::::::::
/* laqy.c
```

DM Brown
29 March 1994

perform label query using information parsed from fin (stdin)

Changed to build_label().

*/

:::::::::::::

larj.c

:::::::::::::

/* larj.c

DM Brown

30 June 1994

send notification to the dwg_notify user that the label
has been rejected.

added mail function because it was easy.

*/

:::::::::::::

laup.c

:::::::::::::

/* laup.c

DM Brown

4 May 1994

This program will create a ufls, populate the ufls
by reading from the file handle passed in, and do a LA.UP

*/

:::::::::::::

list_aspace.pc

:::::::::::::

/* list_aspace.pc

list all the currently available data locations (application spaces)
where files can be copied to/from.

*/

:::::::::::::

list_project.pc

:::::::::::::

/* list_project.pc

list all the currently available projects CDMS knows about.

*/

:::::::::::::

list_users.pc

:::::::::::::

/* list_users.pc

list all the currently active users. any group_id that has its
password disabled (NULL) is skipped.

```
*/
```

```
::::::::::::
```

```
login.c
```

```
::::::::::::
```

```
/* login.c
```

```
DM Brown
```

Prompt the user for an id and password, then: establish dmcs connection, rcid and rtid.

```
*/
```

```
::::::::::::
```

```
myio.c
```

```
::::::::::::
```

```
/* myio.c
```

```
DM Brown
```

```
14 April 1994
```

Provide a central set of routines to handle input and output.

5-23-94

changed show_error() and show_text() to use varargs.

8-9-94

changed get_word() to return multiple word encased in quotes. When a beginning single or double quote is found, it searches until it finds a matching quote or the eoln and returns all the words found.

```
*/
```

```
::::::::::::
```

```
myio.h
```

```
::::::::::::
```

```
/* myio.h
```

prototypes for myio routines.

```
*/
```

```
::::::::::::
```

```
ora_pc.h
```

```
::::::::::::
```

```
/* ora_pc.h
```

```
DM Brown
```

```
31 May 1994
```

Include file for all the pro*c stuff

```
*/
```

```
::::::::::::
```

```
oradcd.c
```

```
::::::::::::
```

```
/* get oradcd.c
```

```
DM Brown
```

```
15 September 1994
```

get the oracle userid and password from oradcd.
 return a pointer to a static buffer with the uid/passwd string
 or NULL.

*/

:::::::::::::

oralogon.pc

:::::::::::::

/* oralogon.pc

DM Brown

1 February 1990

provide standard logon and error reporting functions.

16 Sept 1994

added get_oradcd() to get the userid/passwd for the current KB.

*/

:::::::::::::

proto.h

:::::::::::::

/* proto.h

DM Brown

21 Sep 94

This file is a list of prototypes gleaned from the source
 files listed in proto.files. It was automatically generated
 by the 'makeproto' script.

*/

:::::::::::::

purge_all.c

:::::::::::::

/* purge_all.c

DM Brown

16 August 1994

Search the entire folder for folder items that have been revised.
 Add those to a list and call folder_delete() with the purge flag
 set for each item.

*/

:::::::::::::

set_uinf.c

:::::::::::::

/* set_uinf.c

DM Brown

3 August 1994

Set user environment information, user & default aspace. Get the
 default aspace from the KB.

*/

:::::::::::::

setdefaults.c

:::::::::::::::

/* setdefaults.c

DM Brown

1 April 1994

Set default local environment information for the interpreter.

*/

:::::::::::::::

share_info.pc

:::::::::::::::

/* share_info.pc

DM Brown

2 August 1994

given an application space name, return the dos share information about that location. Relies on the table aspace_share which must be created and populated seperately.

SQL> describe aspace_sharename

Name	Null?	Type
TABLE_NAME	NOT NULL	CHAR(30)
LOCATION_NAME	NOT NULL	CHAR(100)
LOCATION_TYPE		CHAR(1)
SHARE_NAME	NOT NULL	CHAR(25)
SHARE_PW		CHAR(8)
SHARE_SUBDIR		CHAR(100)

*/

:::::::::::::::

shin.c

:::::::::::::::

/* shin.c

DM Brown

3 August 1994

Get the share information about an application space.
See share_info.pc.

*/

:::::::::::::::

user_auth.c

:::::::::::::::

/* user_auth.c

DM Brown

2 June 1994

This routine displays all the groups the indicated user belongs to. This is an indication of the authorizations the user has. There must be a better way to do this, perhaps another routine which interprets the authorization, but that would be fairly

non-portable.

1 July 1994

moved to a .c and created get_uauth.pc which builds a ufcdb
with the information.

*/

::::::::::::

user_info.c

::::::::::::

/* user_info.c

DM Brown

2 June 1994

list the user information for the indicated user.

*/

::::::::::::

userlog.c

::::::::::::

/* userlog.c

DM Brown

2 August 1994

function to log user information to \$DMCS_LOG/userlog. This function
is called from a100 passing the second argument (string) to userlog().
The date, time,uid, and string are logged to the file.

*/

::::::::::::

version.h

::::::::::::

/* version.h

DM Brown

13 Sep 1994

Maintain version information about cdmspci.

*/

::::::::::::

wildcards.c

::::::::::::

/* wildcards.c

DM Brown

19 Sep 1994

scan all the fields set in the query ufcdb for wildcards and replace
them with complex queries.

*/