GCCS System Integration Support

GCCS Studies and Reports:
Global Command and Control System (GCCS)
Common Desktop Environment (CDE)
Integration Specification
(Version 1.0)

April 1, 1996

Prepared for:

DISA/JIEO/JEJ
ATTN: Claire Burchell
45335 Vintage Park Plaza
Sterling, VA 20166-6701

Contract Number: DCA 100-94-D-0014
Delivery Order Number: 204, Task 2
CDRL Number: A005

Prepared by:

Computer Sciences Corporation
Defense Enterprise Integration Services
Four Skyline Place
5113 Leesburg Pike, Suite 700
Falls Church, VA 22041

THIS DOCUMENT IS UNCLASSIFIED

Table of Contents

Section		Page
1.0	PURPOSE	1-1
1.1	Background	1-1
1.2	Referenced Documents	1-1
2.0	GCCS APPLICATION REGISTRATION	2-1
2.1	Features Provided by Registering an Application	2-1
2.2	Purpose of Application Registration	2-2
2.3	General Steps for Registering an Application	2-2
2.3.1	Modifying Font and Color Resources	2-2
2.3.1.1	Modifying Font Resources	2-2
2.3.1.2	Modifying Color Resources	2-3
2.3.2	Creating the Desktop Application Root	2-3
2.3.3	Creating the Registration Package Directories	2-3
2.3.3.1	Registration Package Contents	2-3
2.3.3.2	To Create the Registration Package	2-5
2.3.4	Creating the Actions and Data Types for the Application	2-5
2.3.4.1	Location for Action and Data Type Definition Configuration Files	2-5
2.3.4.2	Ways to Create Actions and Data Type Files	2-6
2.3.5	Putting the Help Files in the Registration Package	2-6
2.3.6	Creating Icons for the Application	2-7
2.3.6.1	Icons Required for the Desktop	2-7
2.3.7	Creating the Application Group	2-8
2.3.7.1	Creating the Application Group Directory	2-8
2.3.7.2	Configuring the Application Group To Use a Unique Icon	2-9
2.3.7.3	Creating the Contents of the Application Group	2-10
2.3.7.4	Creating the Action File (Application Icon)	2-11
2.3.7.5	Read Me Files	2-11
2.4	Example of Creating a Registration Package	2-11
2.4.1	Example: Integration Steps for Registering "GSORTS"	2-12

APPENDICES

APPENDIX A:	EXAMPLE ACTION FILE TEMPLATE	A-1
APPENDIX B:	PRINTING API'S	B-1
APPENDIX C:	PRINT ACTION FILE	C-1

List of Figures

Figure Page

2-1. Application Groups at the Top Level of Application Manager 2-1
2-2. Registration Package Beneath an Application Root Directory 2-4
2-3. The *appmanager* Directory 2-9

Tables

Tables Page

2-1. Application Resources 2-3
2-2. Categories of Configuration Fields 2-4
2-3. Desktop Icon Dimensions and Naming Conditions 2-8

1.0 PURPOSE

This document describes the technical requirements for integrating software components into the Global Command and Control System (GCCS) Desktop. It provides implementation details that describe, from a software development point of view, the following:

- € The requirements for integration of applications under the Common Desktop Environment (CDE),
- € How to correctly write action and data types files,
- € How to correctly build and name icons and icon files,
- € How to integrate printing under CDE, and
- € The required directory structure for application registration.

1.1 Background

Hewlett-Packard (HP) Company, IBM Corporation, Novell, Inc., and SunSoft, Inc. have jointly developed and defined a consistent set of application programming interfaces (API) for a CDE that can be implemented on operating systems that support an X-Window desktop and Open Software Foundation (OSF) Motif. This CDE provides end users with a consistent graphical user interface (GUI) across workstations, X terminals, and PCs; and it provides software developers with a single set of programming interfaces for HP-UX, IBM AIX, Solaris, UnixWare and other UNIX system-based platforms. This advanced environment will enable users to transparently access data and applications from anywhere in the network.

The CDE specification incorporates and integrates existing technology from participating vendors and has been designed to support distributed, enterprise-wide applications. As such, it will scale across a range of client/server platforms, support small workgroups to large enterprises, and support simple text and data uses as well as advanced collaborative multimedia applications.

CDE incorporates elements from HP's Visual User Environment (HP VUE), IBM's Common User Access model, SunSoft's OPEN LOOK DeskSet productivity tools, and Novell's UnixWare Client components. Specific technologies used by the developing companies include the X-Window System (X11R5), OSF Motif toolkit, and SunSoft's ToolTalk interapplication communication protocol, with extensions to incorporate HP’s Encapsulator protocol. Since most of this environment exists today, the companies have integrated key technologies available in the open marketplace and have deviated only where appropriate to give users and software developers a consistent UNIX desktop environment.

1.2 Referenced Documents

- € Advanced User’s and System Administrator’s Guide (TriTeal Enterprise Desktop (TED™) 4.0
- € Common Desktop Environment 1.0: User’s Guide
- € Common Desktop Environment 1.0: Advanced User’s and System Administrator’s Guide
- € Common Desktop Environment 1.0: Programmer’s Overview
- € Common Desktop Environment 1.0: Programmer’s Guide

- Common Desktop Environment 1.0: Help System Author's and Programmer's Guide
- Common Desktop Environment 1.0: Desktop Kornshell User's Guide.

2.0 GCCS APPLICATION REGISTRATION

Application registration is a set of required tasks that must be performed for registration of a GCCS application under the CDE. When an application is fully registered onto the desktop, it has the following properties:

- € Its own application group at the top level of the Application Manager.
- € An action to start the application. The action is represented by an icon in the application group (see Figure 2-1).
- € Data types for the application's own data files.

2.1 Features Provided by Registering an Application

Registering an application under CDE provides a graphical way for users to:

- € Locate the application for execution.

  Once an application is installed, the application is registered into the Application Manager and has its own group.

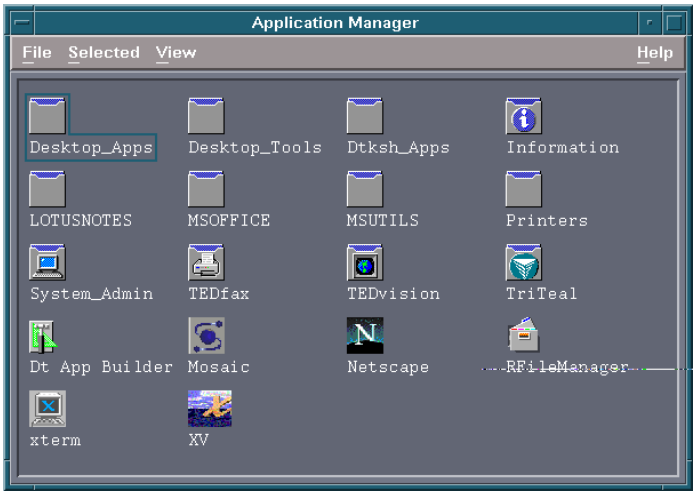

Figure 2-1. Application Groups at the Top Level of Application Manager

- € Identify and manipulate data files (i.e., Printing, Opening).
- € Perform other operations, such as mailing or viewing data.

2.2 Purpose of Application Registration

A registered application must create certain configuration files used by the desktop to provide an interface to the application. The following types of configuration files are a must for the GCCS CDE environment:

- Action and data type definition files,
- Icon image (pixmap or bitmap) files,
- A directory and files that create the application group, and
- Desktop Help files (optional).

For these files to be recognized and used by CDE, they must be in certain directories specified by the desktop's search paths. The desktop allows an application to keep all its desktop configuration files gathered under a single directory. This grouping of files is referred to as a *registration package*.

2.3 General Steps for Registering an Application

NOTE: For a detailed example that uses these steps to create an application package, see "Example of Creating a Registration Package" in the *CDE Advanced User's and System Administrator's Guide*.

2.3.1 Modifying Font and Color Resources

The desktop provides mechanisms for setting and manipulating interface fonts and window colors. For an application to use these mechanisms properly, you may have to modify the application's *app-defaults* file.

2.3.1.1 Modifying Font Resources

NOTE: This subsection applies to applications created using OSF/Motif 1.2 (TM) (or later versions). Style Manager cannot set interface fonts for applications written using earlier versions of OSF/Motif. The desktop Style Manager will set interface fonts for applications created using OSF/Motif 1.2 (or later versions) if the application does not specify application-specific interface fonts. Style Manager provides two fonts:

- A system font used by system areas such as labels, menus, and buttons; and
- A user font used for editable areas such as text fields.

Each font is provided in seven sizes, labeled 1 through 7 in the Fonts dialog box. The Style Manager fonts are connected to actual fonts on the system through Style Manager resources set in */usr/dt/app-defaults/language/Dtstyle*.

If you want the application to use the Style Manager fonts, remove any application resources that interface specific fonts. The desktop automatically sets the application's resources appropriately, as shown in Table 2-1.

Table 2-1. Application Resources

System Resource	Description
FontList	Set to system font
XmText*FontList	Set to user font
XmTextField*FontList	Set to user font

2.3.1.2 Modifying Color Resources

Style Manager provides the ability to change application colors dynamically. The application must be an OSF/Motif 1.1 or 1.2 client. Clients written with other toolkits cannot change color dynamically; color changes take effect when the client is restarted.

The easiest way to use the dynamic colors provided by the desktop is to remove any application color resources for background and foreground color.

2.3.2 Creating the Desktop Application Root

The registration package files for the application are grouped beneath a directory called the application root, or *app_root*. The *app_root* directory used for the desktop configuration files can be the same directory as the application's installation *app_root* or some other directory.

2.3.3 Creating the Registration Package Directories

NOTE: For an example of creating the registration package directories for an application, see Step 3 of "Example of Creating a Registration Package" in the *CDE Advanced User's and System Administrator's Guide.*

The registration package is the group of desktop configuration files used by the desktop to provide a graphical interface for the application.

2.3.3.1 Registration Package Contents

The desktop configuration files include:

- Action and data type definition files;
- Icon image files;
- An application group directory and its contents; and
- Help data files and a Front Panel configuration file (optional).

As described in Paragraph 2.3.2, the registration package is gathered under a top-level directory called the application root. Figure 2-2 illustrates how this would look using the structure from Paragraph 5.2.1 of the DII COE I&RTS Specification.

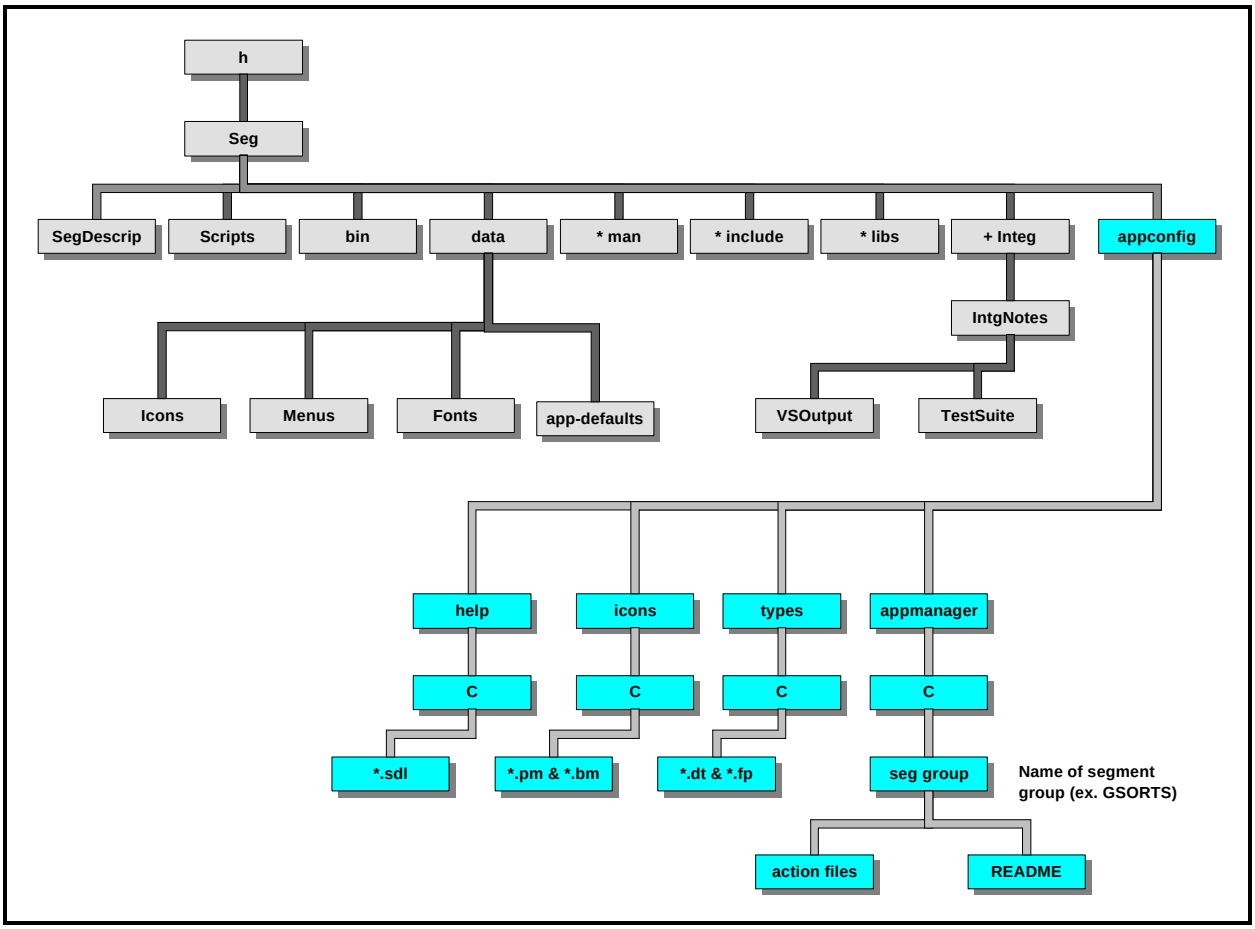

Figure 2-2. Registration Package Beneath an Application Root Directory

The major categories of configuration fields under the */h/Seg/appconfig* directory are shown in Table 2-2.

Table 2-2. Categories of Configuration Fields

Subdirectory	**Contents**
types	Action and data type definitions.
help	Desktop help files.
icons	Bitmap and pixmap image files used by the applications actions and data types.
appmanager	Directory and contents that create the application group.

Each of the major categories has subdirectories for language-dependent files. Default-language files are placed in the *C* directory.

2.3.3.2 To Create the Registration Package

Create these directories. If you are providing language-dependent configuration files, create a separate directory for each language. If you are supplying only one language, put the files in the *C* directory.

NOTE: For GCCS, the “language” part of the directory is “C”. Also “Seg” would be replaced by your segment name, such as GSORTS.

- € /h/Seg/appconfig/types/language
- € /h/Seg/appconfig/help/language
- € /h/Seg/appconfig/icons/language
- € /h/Seg/appconfig/appmanager/language/appgroup_name, where “appgroup_name” is the name of the application group.

2.3.4 Creating the Actions and Data Types for the Application

---

NOTE: For an example of creating the actions and data types for an application, see Step 4 of "Example of Creating a Registration Package" in the *CDE Advanced User’s and System Administrator’s Guide.*

- € **Actions and data types provide a user interface for the application**.

  - Actions provide a user interface for the command to launch the application.
  - Data types provide customized appearance and behavior for the application's data files.

- € **Actions and data types required by an application.**

  - An action that opens the application.
  - A data type for the data files of your application. If you create a data type, you will also want to create:
    - -- An Open action for the data files of your application.
    - -- A Print action for the data files of your application.
  - A data type for the application group.

2.3.4.1 Location for Action and Data Type Definition Configuration Files

Actions and data types are defined in configuration files. The only naming requirement for files containing action and data type definitions is that they must have a *.dt* suffix. By convention, you may want to name the file *action_name.dt* or *application_name.dt*.

Place files containing actions and data types under the application root in the directory /h/Seg/appconfig/types/language. The default language is C.

2.3.4.2 Ways to Create Actions and Data Type Files

In GCCS, action and data types files will be created manually to work under the GCCS Integration Standard. Creating definitions manually requires that you learn the syntax for creating the definitions, but provides access to the full range of functionality. You must also:

- € Create a configuration file containing the action and data type definitions for the application.
- € Follow the naming convention *name.dt* for action and data type definition files.

You can place all your action and data type definitions in one file or distribute them among multiple files. Action and data type names must be one word (no embedded spaces). You can use an underscore character. By convention, the first letter of the action or data type name is capitalized. Do not use an existing action name or file name. Use a name that advanced users and system administrators will easily connect with your application.

If you want the application's icon labeled with a different name than the action name, include a LABEL field in the action definition.

For more information about creating actions and data types, see *"The Advanced User's and System Administrator's Guide"*:

- € Chapter 8, "Introduction to Actions and Data Types."
- € Chapter 9, "Creating Actions and Data Types Using Create Action."
- € Chapter 10, "Creating Actions Manually."
- € Chapter 11, "Creating Data Types Manually."

2.3.5 Putting the Help Files in the Registration Package

If the application includes a desktop help volume (a help volume created with the desktop Help Developer's Kit), the help volume master file (**.sdl*) should be placed in the directory */h/Seg/appconfig/help/language*.

Graphics used by the help files are usually placed in a graphics subdirectory. The graphics must be located in the same directory relative to the master help volume (**.sdl*) file as when the help volume was created.

If the application does not provide a help volume, you can create one if you have the Help Developer 's Kit. There are two levels of integration of a help volume:

- € Full integration: When desktop help is fully integrated, the help volume can be accessed from the application--for example, by on-item help and the Help menu. Full integration involves modification to the application's executables.

€ Partial integration: When desktop help is partially integrated, it is available from the top level of the Help Manager. However, you cannot access the help volume from the application's windows. You can also provide an action to access the help from the application group. The following example action displays the help volume located in the help master file *MyApp.sdl*:

```
ACTION OpenMyAppHelp
{
   LABEL          MyAppHelp
   ARG_COUNT      0
   TYPE           COMMAND
   WINDOW_TYPE    NO_STDIO
   EXEC_STRING    /usr/dt/bin/dthelpview -helpVolume MyApp
   DESCRIPTION    Displays help for the MyApp application.
}
```

2.3.6 Creating Icons for the Application

The desktop provides default icons for actions, data types, and application groups. However, you will probably want to create unique icons for the application.

Icons are placed in the directory */h/Seg/appconfig/icons/C* <language>.

2.3.6.1 Icons Required for the Desktop

The application uses these icon images on the desktop:

€	Action icon:	This is the icon the user double-clicks to start an application (actions). It is referenced in the ICON field of the action that launches the application. Supply three sizes: tiny, medium, and large.
€	Data type icon:	This icon is used to represent the application's data files in File Manager. It is referenced in the ICON field of the data type definition. If your application supports multiple data types, you should provide a different icon for each data type. Supply two sizes: tiny and medium.
€	Application group icon:	This is the icon representing the directory at the top level of the Application Manager. It is referenced in the ICON field of the data type definition for the application gr oup. Supply two sizes: tiny and medium.

You may need to supply both pixmap and bitmap versions of each icon to support color (eight-bit and larger) and monochrome (fewer than eight bits) displays.

Table 2-3 shows the recommended pixel dimensions for desktop icons and the file naming convention to be used.

Table 2-3. Desktop Icon Dimensions and Naming Conditions

Icon Size	Bitmap Name*	Pixmap Name*
16 by 16 (tiny)	name.t.bm	name.t.pm
24 by 24 (small)	name.s.bm	name.s.pm
32 by 32 (medium)	name.m.bm	name.m.pm
48 by 48 (large)	name.l.bm	name.l.pm

* where *name* is the application name

If you do not provide bitmap files, the desktop maps the color specifications of the pixmap files into black and white. However, this mapping may not produce the appearance you want.

For more information about icons, see "Icon Image Files" in the *CDE Advanced User's and System Administrator's Guide*.

2.3.7 Creating the Application Group

Once you have created the action and data type definitions for the application, you must create the configuration files responsible for creating what the user actually sees--the application group and its contents.

The application group is a directory at the top level of the Application Manager.

There are three steps to creating the application group:

a. Create the application group directory in the registration package.

b. Configure the application group to use a unique icon (optional). This involves creating the data type definition for the application group directory.

c. Create the contents of the application group.

2.3.7.1 Creating the Application Group Directory

To create an application group, create the directories in the registration package under *appmanager*, as shown in Figure 2-3.

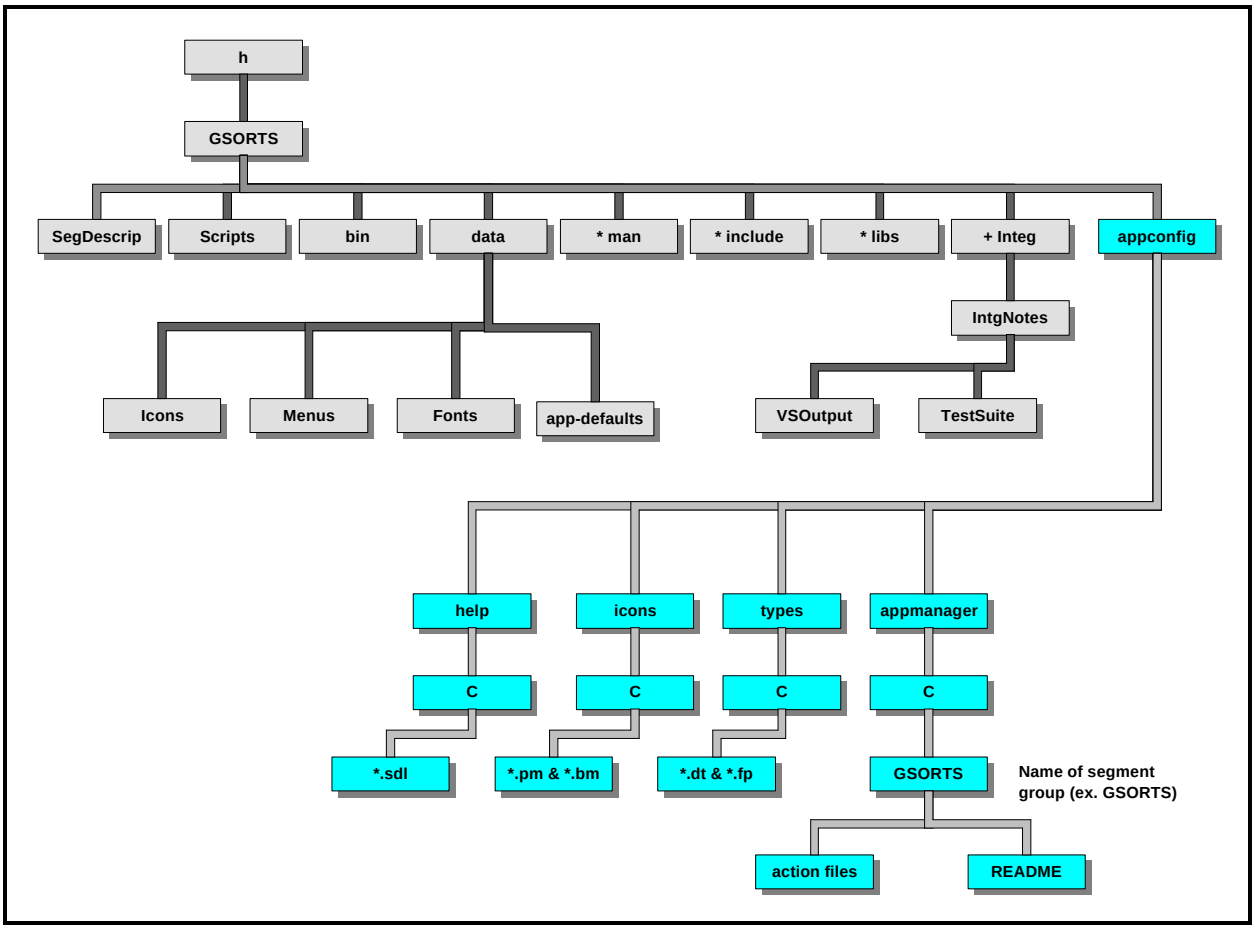

Figure 2-3. The *appmanager* Directory

2.3.7.2 Configuring the Application Group To Use a Unique Icon

The desktop provides a default application-group icon. However, you will probably want to provide a custom icon.

If you want to provide a unique icon for the application group, you must create:

- € A data type for the directory that appears at the top level of Application Manager.
- € Open and Print actions for the data type.

For example, suppose you want to create an application group named *Media_Tools*. The following data type definition, placed in a file */h/Seg/appconfig/types/language/name.dt*, assigns a unique icon to the application group icon.

```
DATA_ATTRIBUTES    Media_ToolsAppgroup
{
   ACTIONS         OpenInPlace,OpenNewView
   ICON            MediaTools
   DESCRIPTION     Double-click to open the Media_Tools application group
}
```

```
DATA_CRITERIA     Media_ToolsAppgroupCriteria1
{
   DATA_ATTRIBUTES_NAME     Media_ToolsAppgroup
   MODE                     d
   PATH_PATTERN             */appmanager/*/Media_Tools
}
```

The Attributes section of the definition specifies the icon to be used. The Criteria section of the definition specifies that the data type be defined to any directory named *Media_Tools* that is a subdirectory of a directory named *appmanager*.

You should also create an Open and Print action for the application group data type:

```
ACTION Open
{
   ARG_TYPE       Media_ToolsAppGroup
   TYPE           MAP
   MAP_ACTION     OpenAppGroup
}

ACTION Print
{
   ARG_TYPE       Media_ToolsAppGroup
   TYPE           MAP
   MAP_ACTION     PrintAppGroup
}
```

OpenAppGroup and PrintAppGroup actions are built-in actions defined in */usr/dt/appconfig/types/language/dtappman.dt.*

2.3.7.3 Creating the Contents of the Application Group

The most important item in the application group is an icon to start the application (an action icon). If the application group contains a suite of applications, there is usually an icon for each application.

In addition to one or more action icons, the application group may contain:

- One or more README files.
- One or more sample data files.
- Templates.
- An icon the user can double-click to view help information.
- A man page.
- A specialized Front Panel control.

The application group can contain subdirectories.

2.3.7.4 Creating the Action File (Application Icon)

The application group should contain an icon that launches the application. If the group supplies a suite of applications, there should be an icon for each one. These icons are called application icons, or action icons, since they represent an underlying action. An action icon is created by creating an executable file with the same name as the action it will run:

/h/Seg/appconfig/appmanager/Seg_group/action_name

The file is called an action file, because its purpose is to create a visual representation of the underlying action. For example, if you've created an action named GSORTS that runs the GSORTS application, you would create an executable file named GSORTS. In File Manager and the Application Manager, the action file will use the icon image specified in the action definition. A sample action file template is provided in Appendix A.

2.3.7.5 Read Me Files

The desktop provides a README data type that you can use for your application's README files. Use one of these naming conventions:

- € README
- € readme
- € README.*
- € Read.*.Me
- € read.*.me
- € READ.*.ME

2.4 Example of Creating a Registration Package

The steps in the following subsections create a registration package for an existing, non-desktop smart application named GSORTS.

The example assumes the following facts about the GSORTS application:

- € It was installed into the directory */h/GSORTS*.
- € The user 's session language is the default value, C.
- € The command line to start GSORTS is: run_gsorts -no_rsh.

GSORTS creates its own window--that is, it does not run inside a terminal emulator window. The existing, non-desktop app-defaults files for GSORTS contain resources for interface fonts and foreground and background colors. An online help volume for GSORTS was created using the desktop Help Developer 's Kit. hen the online help volume was built, it used the following source files:

.../GSORTSHelp.htg
.../graphics/GSORTS1.xwd
.../graphics/GSORTS2.xwd

and generated the file *.../GSORTSHelp.sdl*.

2.4.1 Example: Integration Steps for Registering "GSORTS"

The following procedure will use the mission application GSORTS as an example of how a registration package is done.

a. Modify font and color resources. In GSORTS app-defaults file, remove resources that set:

- € Fonts for text.
- € Colors for foreground and background.

b.. Create the application root. Create the directory:

```
/h/GSORTS
```

If you are integrating an existing application, you should create the application root directory elsewhere than in the installation location for the application; otherwise, the configuration files you create may be removed when you update the application.

c. Create the registration package directories. Create these directories:

```
/h/GSORTS/appconfig/types/C
/h/GSORTS/appconfig/help/C
/h/GSORTS/appconfig/icons/C
/h/GSORTS/appconfig/appmanager/C/GSORTS
```

d. Create the actions and data types for the application:

1. Create the configuration file for the action and data type definitions:

```
/h/GSORTS/appconfig/types/C/GSORTS.dt
```

2. Create the action definition for running GSORTS:

```
ACTION   GSORTS
{
   WINDOW_TYPE       NO_STDIO
   ICON              gsorts
   DESCRIPTION       Double-click this icon or drop a GSORTS
   EXEC_STRING       /h/GSORTS/data/run_gsorts -no_rsh
}
```

3. Create the data type for *.GSORTS files:

```
DATA_ATTRIBUTES   GSORTSDataFile
{
   DESCRIPTION       Gsorts data file.
   ICON              GSORTSData
   ACTIONS           Open,Print
}

DATA_CRITERIA     GSORTSDataFileCriteria1
{
   DATA_ATTRIBUTES_NAME    GSORTSDataFile
   NAME_PATTERN            *.GSORTS
   MODE                    f
}
```

4 Create the data type for *.tpl files:

```
DATA_ATTRIBUTES   GSORTSTemplateFile
{
   DESCRIPTION       GSORTS template file.
   ICON              GSORTSTempl
   ACTIONS           Open
}

DATA_CRITERIAL    GSORTSTemplateFileCriteria1
{
   DATA_ATTRIBUTES_NAME    GSORTSTemplateFile
   NAME_PATTERN            *.tpl
   MODE                    f
}
```

5. Create the Open action for *.GSORTS files:

```
ACTION   Open
{
   ARG_TYPE    GSORTSDataFile
   TYPE        MAP
   MAP_ACTION  GSORTS
}
```

6. Create the Print action for *.GSORTS files.

NOTE:Print API's have been configured for the following graphics formats (see Appendix B for more detail): XWD, JPEG, GIF, PPM, XBM, and PostScript.

Following are simple Print actions that will print the data files. A sample print action file is also provided in Appendix C. These actions require a value for the LPDEST environment variable and ignore the -s print option. (If LPDEST isn't set, the action may fail.)

```
ACTION    Print
{
   ARG_TYPE       GSORTSDataFile
   TYPE           MAP
   MAP_ACTION     GSORTSPrintData
}

ACTION   GSORTSPrintData
{
   WINDOW_TYPE    NO_STDIO
   EXEC_STRING    GSORTSPrint -d $LPDEST %Arg_1%
}
```

Following is another version of the GSORTSPrintData action and an accompanying script. Together, they handle situations where LPDEST is not set or if silent printing is requested.

```
ACTION   GSORTSPrintData
{
   WINDOW_TYPE    NO_STDIO
   EXEC_STRING    /h/GSORTS/Scripts/GSORTSenvprint \
                  %(File)Arg_1%
}
```

The contents of the */h/GSORTS/Scripts/GSORTSenvprint* script is:

```
# GSORTSenvprint
#!/bin/sh
DEST=""
SILENT=""
if   [ $LPDEST ] ; then
DEST="-d $LPDEST"
fi
GSORTSPrint $DEST SILENT $1
```

7. Create the Open action for *.tpl files:

```
ACTION   Open
{
   ARG_TYPE       GSORTSTemplateFile
   TYPE           MAP
   MAP_ACTION     GSORTS
}
```

8. Create the Print action for *.tpl files:

```
ACTION   Print
{
   ARG_TYPES      GSORTSTemplateFile
   TYPE           MAP
   MAP_ACTION     NoPrint  \
                  (If you do not want them to be printed)
}
```

NoPrint is a built-in action that displays a dialog box telling the user the file cannot be printed.

e. Put the help files into the registration package.

1. Place the help files in the following locations:

   /h/GSORTS/appconfig/help/C/GSORTSHelp.sdl
   /h/GSORTS/appconfig/help/C/graphics/GSORTS1.xwd
   /h/GSORTS/appconfig/help/C/graphics/GSORTS2.xwd

2. Create the file:

   /h/GSORTS/appconfig/types/C/GSORTShelp.dt.

   Put the following action definition in the file:

```
ACTION   GSORTSHelp
{
   WINDOW_TYPE    NO_STDIO
   EXEC_STRING    /usr/dt/bin/dthelpview -helpVolume \
                  GSORTSHelp.sdl
   DESCRIPTION    Opens theGSORTS help volume.
}
```

f. Create icons for the application. Use Icon Editor to create the icons. Use the size guidelines provided in Table 2-3:

Create these icon files in the directory */h/GSORTS/appconfig/icons/C*:

To represent:	**Create Icon File**:
The action that runs the application:	gsorts.t.pm, gsorts.m.pm, gsorts.1.pm
*.GSORTS files:	GSORTSData.t.pm, GSORTSData.m.pm
*.tpl files:	GSORTSTempl.t.pm, GSORTSTempl.m.pm
The application group:	GSORTSApp.t.pm, GSORTSApp.m.pm

g. Create the application group.

1. If you haven't already done so, create the directory.

```
/h/GSORTS/appconfig/appmanager/C/GSORTS
```

2. This step is optional. It provides a unique icon for the application group icon by creating a data type and associated actions for the application group. If you omit this step, the application group will use the default icon.

   Add the following data type and action definitions to the file

```
/h/GSORTS/appconfig/types/C/GSORTS.dt
```

   The data type specifies the icon to be used by the GSORTS application group. The actions provide the same Open and Print behavior as the built-in application groups.

```
DATA_ATTRIBUTES    GSORTSAppGroup
{
   ACTIONS         OpenInPlace,OpenNewView
   ICON            GSORTSApp
}

DATA_CRITERIA      GSORTSAppGroupCriterial
{
   DATA_ATTRIBUTES_NAME     BestTextEditorAppGroup
   MODE                     d
   PATH_PATTERN             */appmanager/*/GSORTS
}

ACTION   Open
{
   ARG_TYPE        GSORTSAppGroup
   TYPE            MAP
   MAP_ACTION      OpenAppGroup
}

ACTION   Print
{
   ARG_TYPE        GSORTSAppGroup
   TYPE            MAP
   MAP_ACTION      PrintAppGroup
}
```

3. Create an icon in the application group that will start the application. To do this, create the file:

```
/h/GSORTS/appconfig/appmanager/C/GSORTS/GSORTS
```

   and make the file executable.

4. Create the action file in the application group that will open the help volume. To do this, create the file:

   `/h/GSORTS/appconfig/appmanager/C/GSORTS/GSORTSHelp`

   and make the file executable.

5. Put other files into the application group; for example, "read me" files, sample data and template files.

APPENDIX A

EXAMPLE ACTION FILE TEMPLATE

APPENDIX A: EXAMPLE ACTION FILE TEMPLATE

Data types definition file template:

```
##############################################################################
#
#     Name.dt
#
#     Action and DataType definitions for the Common Desktop Environment
#     <Application name>
#
#
##############################################################################
set DtDbVersion=1.0

#################################################################
#
# Data Attributes
#
#################################################################
```

This is just an example of what goes here.

```
ACTION Print
{
      LABEL           Print
      ARG_TYPE        DtcmAppointmentAttrs
      TYPE            MAP
      MAP_ACTION      NoPrint
}

#################################################################
#
# Actions
#
#################################################################
```

This is just an example of what goes here.

```
ACTION Dtcm
{
      LABEL           Calendar
      ICON            DtCM
      TYPE            COMMAND
      EXEC_STRING     /usr/dt/bin/dtcm
      WINDOW_TYPE     NO_STDIO
      DESCRIPTION     The Calendar (Dtcm) action starts the desktop Calendar \
                      Manager.
}

########################### EOF ##############################################
```

The following is a template for the action file:

```
#######################################################################
# This file represents a @(#)TriTeal Enterprise Desktop (TED) action.  The
# contents of the file does not matter; the action will work even if the file
# is empty.  If this file is executable and the name of the file matches an
# entry in the action database (*.dt files), the TED File Manager will treat
# this file as an action.
#
# For more information on actions, see the TED manuals.
#
# <Application Name>
#
#######################################################################
```

APPENDIX B

PRINTING API'S

APPENDIX B: PRINTING API'S

```
##############################################################################
#
#     datatypes.dt
#
#     TriTeal Enterprise Desktop
#
#     Action and DataType definitions for the TriTeal Enterprise Desktop
#     (TED) DT components.
#
#
#     $Revision: 1.5 $
#
##############################################################################
set DtDbVersion=1.0

#####################################################################
#
# Data Attributes
#
#####################################################################
DATA_ATTRIBUTES   BM
{
      ACTIONS           Open,Print
      ICON              Dtbitmp
      NAME_TEMPLATE     %s.bm
      MIME_TYPE         text/plain
      SUNV3_TYPE        xbm-file
      DESCRIPTION       This is a file containing data in the X11 bitmap \
                        format.  Its data type is named BM. BM files have \
                        names ending with '.bm' or '.xbm'.
}

DATA_CRITERIA     BM1
{
      DATA_ATTRIBUTES_NAME    BM
      MODE              f
      NAME_PATTERN      *.bm
}

DATA_CRITERIA     BM2
{
      DATA_ATTRIBUTES_NAME    BM
      MODE              f
      NAME_PATTERN      *.xbm
}
ACTION      Open
{
      LABEL             Open
      ARG_TYPE          BM
      TYPE              MAP
      MAP_ACTION        Dticon
}
```

```
ACTION      Print
{
      LABEL             Print
      ARG_TYPE          BM
      TYPE              MAP
      MAP_ACTION        PrintXbm
}

#######################################################

DATA_ATTRIBUTES   PM
{
      ACTIONS           Open
      ICON              Dtpixmp
      NAME_TEMPLATE     %s.pm
      MIME_TYPE         text/plain
      SUNV3_TYPE        xpm-file
      DESCRIPTION       This is a pixmap file containing a multicolor image. \
                        Its data type is named PM. PM files have names \
                        ending with '.pm' or '.xpm', or have the characters \
                        *! XPM2" in their contents.
}

DATA_CRITERIA     PM1
{
      DATA_ATTRIBUTES_NAME    PM
      MODE              f
      NAME_PATTERN      *.pm
}

DATA_CRITERIA  PM2
{
      DATA_ATTRIBUTES_NAME    PM
      MODE              f
      NAME_PATTERN      *.xpm
}

DATA_CRITERIA  PM3
{
      DATA_ATTRIBUTES_NAME    PM
      MODE              f
      CONTENT           0 string ! XPM2
}

ACTION      Open
{
      LABEL             Open
      ARG_TYPE          PM
      TYPE              MAP
      MAP_ACTION        Dticon
}
```

```
ACTION      Print
{
      LABEL             Print
      ARG_TYPE          PM
      TYPE              MAP
      MAP_ACTION        NoPrint
}

#######################################################

DATA_ATTRIBUTES   PCL
{
      ACTIONS           Open,Print
      ICON              Dtpcl
      NAME_TEMPLATE     %s.pcl
      MIME_TYPE         application/octet-stream
      DESCRIPTION       This is a file containing data in the format of the \
                        Printer Control Language (PCL). Its data type is \
                        named PCL.  PCL files have names ending with '.pcl'.
}

DATA_CRITERIA  PCL1
{
      DATA_ATTRIBUTES_NAME    PCL
      CONTENT           0 byte 033 0105
      MODE              f&!x
}

DATA_CRITERIA  PCL2
{
      DATA_ATTRIBUTES_NAME    PCL
      NAME_PATTERN      *.pcl
      MODE              f&!x
}

ACTION   Open
{
      LABEL             Open
      ARG_TYPE          PCL
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dterror.ds \
                        "Cannot open - No PCL viewer available." \
                        "Information" \
                        "OK"
      DESCRIPTION       Your system does not provide a viewer for PCL files.\
                        Attempting to open this file displays an error \
                        dialog box.
}

ACTION   Print
{
      LABEL             Print
      ARG_TYPE          PCL
      TYPE              MAP
      MAP_ACTION        PrintRaw
}
```

```
#######################################################

DATA_ATTRIBUTES   POSTSCRIPT
{
      ACTIONS           Open,Print
      ICON              Dtps
      NAME_TEMPLATE     %s.ps
      MIME_TYPE         application/postscript
      SUNV3_TYPE        postscript-file
      DESCRIPTION       This file contains postscript data. Its data type \
                        is named PS.  PS file have names ending with '.ps' \
                        or '.PS', or contain the characters "%!".
}

DATA_CRITERIA  POSTSCRIPT1
{
      DATA_ATTRIBUTES_NAME  POSTSCRIPT
      MODE              f&!x
      NAME_PATTERN      *.ps
}

DATA_CRITERIA  POSTSCRIPT2
{
      DATA_ATTRIBUTES_NAME  POSTSCRIPT
      MODE              f&!x
      NAME_PATTERN      *.PS
}

DATA_CRITERIA  POSTSCRIPT3
{
      DATA_ATTRIBUTES_NAME  POSTSCRIPT
      CONTENT           0 string %!
      MODE              f&!x
}

ACTION   Open
{
      LABEL             Open
      ARG_TYPE          POSTSCRIPT
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       sh -c ' \
                        case "`uname -s`" in \
                           SunOS)imagetool "%Arg_1%" ;;\
                           *) /usr/dt/bin/dterror.ds \
                              "Cannot open - No Postscript viewer available." \
                              "Information" \
                              "OK" ;; esac'
      DESCRIPTION       Your system does not provide a postscript viewer. \
                        Attempting to open this file displays an error \
                        dialog box.
}

ACTION   Print
{
      LABEL             Print
      ARG_TYPE          POSTSCRIPT
      TYPE              MAP
```

```
      MAP_ACTION        PrintRaw
}
#######################################################

DATA_ATTRIBUTES   AUDIO
{
      ACTIONS           Open,Play
      ICON              Dtaudio
      NAME_TEMPLATE     %s.au
      MIME_TYPE         audio/basic
      SUNV3_TYPE        audio-file
      DESCRIPTION       This file contains audio (sound) data.  Its data type \
                        is named AUDIO.  AUDIO file have names ending with \
                        '.snd', '.wav', or '.au', or contain the \
                        characters ".snd".
}

DATA_CRITERIA  AUDIO1
{
      DATA_ATTRIBUTES_NAME  AUDIO
      MODE              f
      NAME_PATTERN      *.snd
}

DATA_CRITERIA  AUDIO2
{
      DATA_ATTRIBUTES_NAME  AUDIO
      MODE              f
      NAME_PATTERN      *.wav
}

DATA_CRITERIA  AUDIO3
{
      DATA_ATTRIBUTES_NAME  AUDIO
      MODE              f
      NAME_PATTERN      *.au
}

DATA_CRITERIA AUDIO4
{
      DATA_ATTRIBUTES_NAME  AUDIO
      CONTENT           0 string .snd
      MODE              f
}

ACTION   Open
{
      LABEL             Open
      ARG_TYPE          AUDIO
      TYPE              MAP
      MAP_ACTION        Play
}
```

```
ACTION   Play
{
      LABEL             Play
      ARG_TYPE          AUDIO
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       sh -c ' \
                        if [ "$DTXSERVERLOCATION" = "local" ]; \
                        then \
                           case "`uname -s`" in \
                              HP-UX) if [ -f /usr/audio/bin/AudioCP ]; \
                                     then \
                                      /usr/audio/bin/AudioCP "%(File)Arg_1%"; \
                                     else \
                                        CDE_INSTALLATION_TOP/bin/dterror.ds \
                    "Cannot play %(File)Arg_1% - Could not find audio player." \
                                        "Information" \
                                        "OK"; \
                                     fi \
                                     ;; \
                              SunOS) audiotool "%(File)Arg_1%"  \
                                     ;; \
                                     *) CDE_INSTALLATION_TOP/bin/dterror.ds \
                    "Cannot play %(File)Arg_1% - No audio player available." \
                                     "Information" \
                                     "OK" \
                                     ;; \
                           esac; \
                        else \
                           /usr/dt/bin/dterror.ds \
                           "X server is remote.  Cannot play %(File)Arg_1%." \
                           “Information" \
                           "OK"; \
                        fi'
      DESCRIPTION       Attempts to play this file using the platform's audio
                        player.
}

ACTION   Print
{
      LABEL             Print
      ARG_TYPE          AUDIO
      TYPE              MAP
      MAP_ACTION        NoPrint
}

##########################################################
```

```
DATA_ATTRIBUTES TIFF
{
      ACTIONS           Open,Print
      ICON              Dttif
      NAME_TEMPLATE     %s.tif
      MIME_TYPE         image/tiff
      SUNV3_TYPE        tiff-file
      DESCRIPTION       This file contains a graphics image in TIFF format. \
                        Its data type is named TIFF.   TIFF files have names \
                        ending with '.TIFF', '.TIF', '.tiff', or '.tif'.
}

DATA_CRITERIA  TIFF1
{
      DATA_ATTRIBUTES_NAME  TIFF
      MODE              f
      NAME_PATTERN      *.TIFF
}

DATA_CRITERIA TIFF2
{
      DATA_ATTRIBUTES_NAME  TIFF
      MODE              f
      NAME_PATTERN      *.TIF
}

DATA_CRITERIA     TIFF3
{
      DATA_ATTRIBUTES_NAME  TIFF
      MODE              f
      NAME_PATTERN      *.tiff
}

DATA_CRITERIA  TIFF4
{
      DATA_ATTRIBUTES_NAME  TIFF
      MODE              f
      NAME_PATTERN      *.tif
}
```

```
ACTION   Open
{
      LABEL             Open
      ARG_TYPE          TIFF
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       sh -c '
                        case "`uname -s`" in \
                           SunOS) imagetool "%Arg_1%" ;;\
                           HP-UX) /usr/bin/X11/imageview "%Arg_1%" ;; \
                           *) /usr/dt/bin/dterror.ds \
                              "Cannot open - No TIFF file viewer available." \
                              "Information" \
                              "OK" ;; \
                        esac'
      DESCRIPTION       Your system does not provide a TIFF viewer. \
                        Attempting to open this file displays an error \
                        dialog box.
}

ACTION   Print
{
      LABEL             Print
      ARG_TYPE          IFF
      TYPE              MAP
      MAP_ACTION        NoPrint
}

##########################################################

DATA_ATTRIBUTES JPEG
{
      ACTIONS           Open,Print
      ICON              Dtjpegi
      NAME_TEMPLATE     %s.jpg
      MIME_TYPE         image/jpeg
      SUNV3_TYPE        jpeg-file
      DESCRIPTION       This file contains a graphics image in JPEG \
                        Interchange File Format.   Its data type is named \
                        JPEG. JPEG files have names ending with '.JPEG', \
                        '.jpg', '.jpeg', or '.JPG'.
}

DATA_CRITERIA  JPEG1
{
      DATA_ATTRIBUTES_NAME  JPEG
      MODE              f
      NAME_PATTERN      *.JPEG
}

DATA_CRITERIA  JPEG2
{
      DATA_ATTRIBUTES_NAME  JPEG
      MODE              f
      NAME_PATTERN      *.jpg
}
```

```
DATA_CRITERIA  JPEG3
{
      DATA_ATTRIBUTES_NAME  JPEG
      MODE                  f
      NAME_PATTERN          *.jpeg
}

DATA_CRITERIA  JPEG4
{
      DATA_ATTRIBUTES_NAME  JPEG
      MODE                  f
      NAME_PATTERN          *.JPG
}

ACTION   Open
{
      LABEL                 Open
      ARG_TYPE              JPEG
      TYPE                  COMMAND
      WINDOW_TYPE           NO_STDIO
      EXEC_STRING sh        -c ' \
                            case "`uname -s`" in \
                               SunOS) imagetool "%Arg_1%" ;; \
                               HP-UX)/usr/bin/X11/imageview "%Arg_1%" ;; \
                               *) /usr/dt/bin/dterror.ds \
                                  "Cannot open - No JPEG file viewer available." \
                                  "Information" \
                                  "OK" ;; \
                            esac'
      DESCRIPTION           Your system does not provide a JPEG viewer.   \
                            Attempting to open this file displays an \
                            error dialog box.
}

ACTION   Print
{
      LABEL                 Print
      ARG_TYPE              JPEG
      TYPE                  MAP
      MAP_ACTION            PrintJpg
}

##########################################################

DATA_ATTRIBUTES   PPM
{
      ACTIONS               Open,Print
      ICON                  Dtdata
      NAME_TEMPLATE         %s.ppm
      MIME_TYPE             image/ppm
      SUNV3_TYPE            ppm-file
      DESCRIPTION           This file contains a graphics image in PPM \
                            Interchange File Format.  Its data type is named \
                            PPM.  PPM files have names ending with '.PPM', \
                            '.ppm', '.pnm' or '.PNM'.
}

DATA_CRITERIA  PPM1
```

```
{
      DATA_ATTRIBUTES_NAME  PPM
      MODE                  f
      NAME_PATTERN          *.PPM
}

DATA_CRITERIA  PPM2
{
      DATA_ATTRIBUTES_NAME  PPM
      MODE                  f
      NAME_PATTERN          *.ppm
}

DATA_CRITERIA  PPM3
{
      DATA_ATTRIBUTES_NAME  PPM
      MODE                  f
      NAME_PATTERN          *.pnm
}

DATA_CRITERIA  PPM4
{
      DATA_ATTRIBUTES_NAME  PPM
      MODE                  f
      NAME_PATTERN          *.PNM
}

ACTION   Open
{
      LABEL                 Open
      ARG_TYPE              PPM
      TYPE                  COMMAND
      WINDOW_TYPE           NO_STDIO
      EXEC_STRING           sh -c ' \
                            case "`uname -s`" in \
                               SunOS)imagetool "%Arg_1%" ;; \
                               HP-UX)/usr/bin/X11/imageview "%Arg_1%" ;; \
                               *)/usr/dt/bin/dterror.ds \
                                  "Cannot open - No JPEG file viewer available." \
                                  "Information" \
                                  "OK" ;; \
                            esac'
      DESCRIPTION           Your system does not provide a JPEG viewer. \
                            Attempting to open this file displays an error \
                            dialog box.
}

ACTION   Print
{
      LABEL                 Print
      ARG_TYPE              PPM
      TYPE                  MAP
      MAP_ACTION            PrintPpm
}

#########################################################
```

```
DATA_ATTRIBUTES   XWD
{
      ACTIONS            Open,Print
      ICON               Dtxwd
      NAME_TEMPLATE      %s.xwd
      MIME_TYPE          image/xwd
      SUNV3_TYPE         xwd-file
      DESCRIPTION        This file contains a graphics image in XWD \
                         Interchange File Format.   Its data type is named \
                         XWD.  XWD files have names ending with '.XWD', \
                         '.xwd'.
}

DATA_CRITERIA  XWD1
{
      DATA_ATTRIBUTES_NAME  XWD
      MODE               f
      NAME_PATTERN       *.XWD
}

DATA_CRITERIA  XWD2
{
      DATA_ATTRIBUTES_NAME  XWD
      MODE               f
      NAME_PATTERN       *.xwd
}

ACTION   Open
{
      LABEL              Open
      ARG_TYPE           XWD
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO
      EXEC_STRING        sh -c ' \
                         case "`uname -s`" in \
                            SunOS)   imagetool "%Arg_1%" ;; \
                            HP-UX) /usr/bin/X11/imageview "%Arg_1%" ;; \
                            *) /usr/dt/bin/dterror.ds \
                            "Cannot open - No XWD file viewer available." \
                            "Information" \
                               "OK" ;; \
                         esac'
      DESCRIPTION        Your system does not provide a XWD viewer. \
                         Attempting to open this file displays an error \
                         dialog box.
}
```

```
ACTION   Print
{
      LABEL             Print
      ARG_TYPE          XWD
      TYPE              MAP
      MAP_ACTION        PrintXwd
}

#######################################################

DATA_ATTRIBUTES   MPEG
{
      ACTIONS           Open
      ICON              Dtmpeg
      NAME_TEMPLATE     %s.mpg
      MIME_TYPE         video/mpeg
      DESCRIPTION       This file contains a graphics image movie in MPEG \
                        Interchange File Format.   Its data type is named \
                        MPEG. MPEG files have names ending with '.MPEG', \
                        '.mpg', '.mpeg', or '.MPG'.
}

DATA_CRITERIA  MPEG1
{
      DATA_ATTRIBUTES_NAME  MPEG
      MODE              f
      NAME_PATTERN      *.MPEG
}

DATA_CRITERIA  MPEG2
{
      DATA_ATTRIBUTES_NAME  MPEG
      MODE              f
      NAME_PATTERN      *.mpg
}

DATA_CRITERIA  MPEG3
{
      DATA_ATTRIBUTES_NAME  MPEG
      MODE              f
      NAME_PATTERN      *.mpeg
}

DATA_CRITERIA  MPEG4
{
      DATA_ATTRIBUTES_NAME  MPEG
      MODE              f
      NAME_PATTERN      *.MPG
}
```

```
ACTION   Open
{
      LABEL             Open
      ARG_TYPE          MPEG
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dterror.ds \
                        "Cannot open - No MPEG player available." \            "
                                                                               I
                                                                               n
                                                                               f
                                                                               o
                                                                               r
                                                                               m
                                                                               a
                                                                               t
                                                                               i
                                                                               o
                                                                               n
                                                                               "
                                                                               \
                           "OK"
      DESCRIPTION       Your system does not provide a MPEG viewer.   \
                        Attempting to open this file displays an error \
                        dialog box.
}

ACTION   Print
{
      LABEL             Print
      ARG_TYPE          MPEG
      TYPE              MAP
      MAP_ACTION        NoPrint
}

##########################################################

DATA_ATTRIBUTES   GIF
{
      ACTIONS           Open,Print
      ICON              Dtgif
      NAME_TEMPLATE     %s.gif
      MIME_TYPE         image/gif
      SUNV3_TYPE        gif-file
      DESCRIPTION       This file contains a graphics image in GIF format.  \
                        Its data type is named GIF.   GIF files have names \
                        ending with .gif or .GIF, or contain the characters \
                        "GIF87a" or "GIF89a".
}

DATA_CRITERIA  GIF1
{
      DATA_ATTRIBUTES_NAME  GIF
      MODE              f
      NAME_PATTERN      *.gif
}

DATA_CRITERIA  GIF2
```

```
{
      DATA_ATTRIBUTES_NAME  GIF
      MODE                  f
      NAME_PATTERN          *.GIF
}

DATA_CRITERIA  GIF3
{
      DATA_ATTRIBUTES_NAME  GIF
      CONTENT               0 string GIF87a
      MODE                  f
}

DATA_CRITERIA  GIF4
{
      DATA_ATTRIBUTES_NAME  GIF
      CONTENT               0 string GIF89a
      MODE                  f
}

ACTION   Open
{
      LABEL                 Open
      ARG_TYPE              GIF
      TYPE                  COMMAND
      WINDOW_TYPE           NO_STDIO
      EXEC_STRING           sh -c ' \
                            case "`uname -s`" in \
                            SunOS) imagetool "%Arg_1%" ;; \
                            HP-UX) /usr/bin/X11/imageview "%Arg_1%" ;; \
                            *) /usr/dt/bin/dterror.ds \
                                  "Cannot open - No GIF file viewer available." \
                                  "Information" \
                                  "OK" ;; \
                            esac'
      DESCRIPTION           Your system does not provide a GIF viewer.    \
                            Attempting to open this file displays an error \
                            dialog box.
}

ACTION   Print
{
      LABEL                 Print
      ARG_TYPE              GIF
      TYPE                  MAP
      MAP_ACTION            PrintGif
}

##########################################################

DATA_ATTRIBUTES   README
{
      ACTIONS               Open,Print
      ICON                  DtRdMe
      IS_TEXT               true
      NAME_TEMPLATE         README
      MIME_TYPE             text/plain
```

```
      DESCRIPTION       This file is a text file containing information about \
                        the files in the current folder.  Its data type is \
                        named README.  README files have names beginning with \
                        READ and ending with ME, with 0 or more characters in \
                        between. Letters can be uppercase or lowercase.
}

DATA_CRITERIA  README1
{
      DATA_ATTRIBUTES_NAME  README
      MODE              f
      NAME_PATTERN      README
}
DATA_CRITERIA  README2
{
      DATA_ATTRIBUTES_NAME  README
      MODE              f
      NAME_PATTERN      README.*
}

DATA_CRITERIA  README3
{
      DATA_ATTRIBUTES_NAME  README
      MODE              f
      NAME_PATTERN      Read.*.Me
}

DATA_CRITERIA  README4
{
      DATA_ATTRIBUTES_NAME  README
      MODE              f
      NAME_PATTERN      read.*.me
}

DATA_CRITERIA  README5
{
      DATA_ATTRIBUTES_NAME  README
      MODE              f
      NAME_PATTERN      READ.*.ME
}

DATA_CRITERIA  README6
{
      DATA_ATTRIBUTES_NAME  README
      MODE              f
      NAME_PATTERN      readme
}

#######################################################


DATA_ATTRIBUTES   HTML
{
      ACTIONS           Open,Print
      ICON              Dtdata
      IS_TEXT           true
      NAME_TEMPLATE     %s.html
      MIME_TYPE         text/html
```

```
      DESCRIPTION        This file is a text file containing Hypertext Markup \
                         Language from the World Wide Web.   Its data type is \
                         named HTML. HTML files have names ending with \
                         '.html', or contain the characters "<HTML>"   or \
                         "<html>".
}

DATA_CRITERIA  HTML1
{
      DATA_ATTRIBUTES_NAME  HTML
      MODE               f&!x
      NAME_PATTERN       *.html
}

DATA_CRITERIA  HTML2
{
      DATA_ATTRIBUTES_NAME  HTML
      MODE               f&!x
      CONTENT            0 string <HTML>
}

DATA_CRITERIA  HTML2
{
      DATA_ATTRIBUTES_NAME  HTML
      MODE               f&!x
      CONTENT            0 string <html>
}

ACTION   Print
{
      LABEL              Print
      ARG_TYPE           HTML
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO
      EXEC_STRING        /usr/dt/bin/dtlp "%Arg_1%"
}

##########################################################

DATA_ATTRIBUTES   RICHTEXT
{
      ACTIONS            Open,Print
      ICON               Dtdata
      IS_TEXT            true
      NAME_TEMPLATE      %s.rt
      MIME_TYPE          text/richtext
      DESCRIPTION        This file is a text file containing multiple font \
                         definitions as well as plain text.  Its data type is \
                         named RICHTEXT.   RICHTEXT files have names  ending \
                         with '.rt'.
}

DATA_CRITERIA  RICHTEXT1
{
      DATA_ATTRIBUTES_NAME  RICHTEXT
      MODE               f&!x
      NAME_PATTERN       *.rt
}
```

```
DATA_CRITERIA  RICHTEXT2
{
      DATA_ATTRIBUTES_NAME  RICHTEXT
      MODE                  f&!x
      NAME_PATTERN          *.rtf
}

ACTION   Print
{
      LABEL                 Print
      ARG_TYPE              RICHTEXT
      TYPE                  COMMAND
      WINDOW_TYPE           NO_STDIO
      EXEC_STRING           /usr/dt/bin/dtlp "%Arg_1%"
}

#########################################################

DATA_ATTRIBUTES   UIL
{
      ACTIONS               Open,Print
      ICON                  Dtuil
      NAME_TEMPLATE         %s.uil
      MIME_TYPE             text/plain
      DESCRIPTION           This is a text file containing source code that can \
                            be used by a UIL interpreter to build a Motif user \
                            interface.  UIL files have names in '.uil' or '.UIL'.
}

DATA_CRITERIA  UIL1
{
      DATA_ATTRIBUTES_NAME    UIL
      MODE                  !d
      NAME_PATTERN          *.uil
}

DATA_CRITERIA  UIL2
{
      DATA_ATTRIBUTES_NAME  UIL
      MODE                  !d
      NAME_PATTERN          *.UIL
}


#########################################################

DATA_ATTRIBUTES   MSDOS_EXE
{
      ACTIONS               (None)
      ICON                  Dtexec
      IS_EXECUTABLE         true
      NAME_TEMPLATE         %s.exe
      MIME_TYPE             application/octet-stream
      SUNV3_TYPE            msdos-file
      DESCRIPTION           This is an an executable file that runs on \
                            MSDOS- compatible personal computers.  Its data type \
                            is named MSDOS_EXE.  MSDOS_EXE files have names \
                            ending with '.exe' or '.EXE'.
```

```
}

DATA_CRITERIA  MSDOS_EXE1
{
      DATA_ATTRIBUTES_NAME  MSDOS_EXE
      MODE                  f&x
      NAME_PATTERN          *.exe
}

DATA_CRITERIA  MSDOS_EXE2
{
      DATA_ATTRIBUTES_NAME  MSDOS_EXE
      MODE                  f&x
      NAME_PATTERN          *.EXE
}

ACTION   Print
{
      LABEL                 Print
      ARG_TYPE              MSDOS_EXE
      TYPE                  MAP
      MAP_ACTION            NoPrint
}

########################### EOF ##############################################
```

APPENDIX C

PRINT ACTION FILE

APPENDIX C: PRINT ACTION FILE

```
##############################################################################
#
#     print.dt
#
#     TriTeal Enterprise Desktop
#
#     Action and DataType Definitions for the printing
#
#     (c) Copyright 1994, 1995   TriTeal Corporation
#     (c) Copyright 1993, 1994, 1995 Hewlett-Packard Company
#     (c) Copyright 1993, 1994, 1995 International Business Machines Corp.
#     (c) Copyright 1993, 1994, 1995 Sun Microsystems, Inc.
#     (c) Copyright 1993, 1994, 1995 Novell, Inc.
#
#     $Revision: 1.4.2.1 $
#
#     The printer model for TED is centered on the 'Print' action.
#
#     You can define multiple 'Print' actions, one per data type.
#     If no arguments are supplied to the 'Print' action, then the dtprintinfo
#     command is invoked to show the printer and job status.
#
#     'dtprintinfo -populate' is an administration tool used to create default
#     printer actions of the form   '<printer name>_Print'.
#
#     When a new file type is added to the system, a file-type specific 'Print'
#     action may be created that the desktop action engine will automatically
#     use instead of the default.   This new action may use 'dtlp' or rely on
#     its own print facilities to gather application-specific arguments.
#
##############################################################################
set DtDbVersion=1.0

##############################################################################
#
# WARNING:  This file may be overwritten in subsequent installations of
#     the TriTeal Enterprise Desktop (TED).   Consequently, any system wide
#     changes should be made to an equivalent database file in
#     /etc/dt/types and not in this file.
#
##############################################################################

#################################################################
#
# Data Attributes
#
#################################################################

DATA_ATTRIBUTES   PRINTER_UNKNOWN
{
      ACTIONS            PrinterUnconfigured
      ICON               DtPrtun
      IS_EXECUTABLE      true
      DESCRIPTION        This icon represents a printer that is no longer \
                         registered on your desktop.
}
```

```
DATA_CRITERIA     PRINTER_UNKNOWN1
{
      DATA_ATTRIBUTES_NAME  PRINTER_UNKNOWN
      MODE                  f&x
      NAME_PATTERN          *_Print
}

#
# The default print action, invoked for generic ARG_TYPEs; that is,
# for files that have no other print action defined.
#

ACTION   Print
{
      LABEL                 Print
      ARG_TYPE              *
      TYPE                  COMMAND
      WINDOW_TYPE           NO_STDIO
      EXEC_STRING           /usr/dt/bin/dtlp %(File)Arg_1%
      DESCRIPTION           This is the default print action.  When it is run \
                            with a file argument, it gathers printer-specific \
                            options and then prints the file.  When it is run \
                            with no arguments, it displays the WHAT IS THIS \
                            WINDOW CALLED window.
}

ACTION   PrintRaw
{
      TYPE                  COMMAND
      WINDOW_TYPE           NO_STDIO
      EXEC_STRING           /usr/dt/bin/dtlp -w %(File)Arg_1%
}

#
#     If no file arguments are provided, invoke the Dtprintinfo
#     action.
#

ACTION   Print
{
      LABEL                 Print
      ARG_COUNT             0
      TYPE                  MAP
      MAP_ACTION            Dtprintinfo
}

#
#     The Print Manager (DtPrintManager) action invokes the
#     'dtprintinfo -all' command.
#
```

```
ACTION DtPrintManager
{
      LABEL             Print Manager
      ARG_COUNT         0
      ICON              FpPrtmg
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dtprintinfo -all
      DESCRIPTION       The Print Manager (DtPrintManager) action enables you \
                        to determine the status of and view print jobs of \
                        multiple printers. It also allows you to drag \
                        printers to Install Icon controls on sub-panels.
}

ACTION   DtPrintManager
{
      LABEL             Print Manager
      ARG_COUNT         >0
      TYPE              MAP
      MAP_ACTION        DtPrint
}

ACTION   PrintManager
{
      LABEL             Print Manager
      ICON              FpPrtmg
      TYPE              MAP
      MAP_ACTION        DtPrintManager
}


#
#     The Print Jobs (Dtprintinfo) action invokes the 'dtprintinfo'
#     command or the 'dtprintinfo -p printer' command.
#

ACTION   Dtprintinfo
{
      LABEL             Print Jobs
      ICON              Fpprnt
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dtprintinfo -p %(String)Arg_1%
      DESCRIPTION       The Print Jobs (Dtprintinfo) action enables you to \
                        determine the status and view print jobs of a printer.
}

ACTION   Dtprintinfo
{
      LABEL             Print Jobs
      ARG_COUNT         0
      ICON              Fpprnt
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dtprintinfo
      DESCRIPTION       The Print Jobs (Dtprintinfo) action enables you to \
                        determine the status and view print jobs of a printer.
}
```

```
#################################################################
#
#     Actions for the Default Printer
#
#################################################################

#
#     If a file argument is provided, invoke the Print
#     action.
#

ACTION DtPrint
{
      LABEL             Default Printer
      ARG_TYPE          *
      ICON              Fpprnt
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dtaction Print %(File)Arg_1%
      DESCRIPTION       This is the default print action.   When it is run \
                        with a file argument, it gathers printer-specific \
                        options and then prints the file.   When it is run \
                        with no arguments, it displays the WHAT IS THIS \
                        WINDOW CALLED window.
}

#
#     If no file arguments are provided, invoke the Dtprintinfo
#     action.
#

ACTION   DtPrint
{
      LABEL             Default Printer
      ARG_COUNT         0
      ICON              Fpprnt
      TYPE              MAP
      MAP_ACTION        Dtprintinfo
      DESCRIPTION       The Print Jobs (Dtprintinfo) action enables you to \
                        determine the status and view print jobs of a printer.
}
```

```
#
#     Here are actions and data attributes for unknown and unconfigured
#     printer objects.
#

ACTION PrinterUnconfigured
{
      LABEL             PrinterUnconfigured
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dterror.ds \
                        "This printer control is not configured.\\nTo \
                        configure it, press 'Help' and follow \\nthe \
                        instructions." "Unconfigured Printer" \
                        "OK" \
                        "HELP"
      DESCRIPTION       The PrinterUnconfigured action displays a \
                        dialog which states that the printer has not \
                        yet been configured.
}

ACTION   Printer_Trash
{
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       /usr/dt/bin/dterror.ds \
                        "Putting a printer in the trash does not\\n\
                        delete it from the system.  Refer to your\\n\
                        system administration documentation to\\n\
                        delete a printer from the system." \
                        "Delete Printer" \
                        "OK"
      DESCRIPTION       The Printer_Trash action displays a \
                        dialog which states that putting a printer in the \
                        trash can will not delete that printer from your \
                        system.
}

########################### EOF ##############################################

ACTION   PrintXwd
{
      TYPE              COMMAND
      WINDOW_TYPE       NO_STDIO
      EXEC_STRING       sh -c ' \
                        XWDFILE=%Arg_1% ; \
                        `xpr -device ps -gray 2 -out /tmp/${PFILE}.out %Arg_1% ;\
                        dtaction Print /tmp/${PFILE}.out ; \
                        rm -f /tmp/${PFILE}.out`
}
```

```
ACTION   PrintJpg
{
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO
      EXEC_STRING        sh -c ' \
                         JPEGFILE=%Arg_1% ; \
                         /h/COTS/TED/Tools/Image/djpeg -gif %Arg_1% | \
                         /h/COTS/TED/Tools/Image/giftopnm 2>/dev/null | \
                         /h/COTS/TED/Tools/Image/pnmtoxwd 2>/dev/null | \
                         xpr -device ps -gray 2 -out /tmp/${JPEGFILE}.out ; \
                         dtaction Print /tmp/${JPEGFILE}.out ; rm -f \
                         /tmp/${JPEGFILE}.out`
}

ACTION   PrintTif
{
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO

}

ACTION   PrintGif
{
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO
      EXEC_STRING        sh -c ' \
                         GIFFILE=%Arg_1% ; \
                         /h/COTS/TED/Tools/Image/giftopnm %Arg_1% 2>/dev/null |\
                         /h/COTS/TED/Tools/Image/pnmtoxwd 2>/dev/null | xpr \
                         -device ps -gray 2 -out/tmp/${GIFFILE}.out ; dtaction \
                         Print /tmp/${GIFFILE}.out ; rm -f /tmp/${GIFFILE}.out`
}

ACTION   PrintPpm
{
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO
      EXEC_STRING        sh -c ' \
                         PPMFILE=%Arg_1% ; \
                         /h/COTS/TED/Tools/Image/pnmtoxwd %Arg_1% 2>/dev/null |\
                         xpr -device ps -gray 2 -out /tmp/${PPMFILE}.out ; \
                         dtaction Print /tmp/${PPMFILE}.out ; rm -f \
                         /${PPMFILE}/xpr.out`
}

ACTION   PrintXbm
{
      TYPE               COMMAND
      WINDOW_TYPE        NO_STDIO
      EXEC_STRING        sh -c ' \
                         XBMFILE=%Arg_1% ; \
                         /h/COTS/TED/Tools/Image/xbmtopbm %Arg_1% 2>/dev/null |\
                         /h/COTS/TED/Tools/Image/pbmtolps 2>/dev/null \
                         >/tmp/${XBMFILE}.out ; dtaction Print \
                         /tmp/${XBMFILE}.out ; rm -f /${XBMFILE}/xpr.out`
}
```