U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 5141

National PDES Testbed Report Series

Data Probe User's Guide

David A. Sauder

NATIONAL PDES TESTBED

March 1993

NIST

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 5141

National PDES Testbed

NATIONAL PDES TESTBED

Data Probe User's Guide

David A. Sauder

March 1993

Data Probe User's Guide

David A. Sauder

Abstract

Data Probes are software tools built using the emerging international Standard for the Exchange of Product Model Data -- commonly referred to as STEP. A Data Probe is easily created from any information model written in the Express specification language. It is used to view information from the information model for which it was created and is also used to create, edit, or view data that conforms to that information model.

Data Probe was built at the National Institute of Standards and Technology to assist in the development of STEP and is part of the set of tools that are used in the National PDES Testbed. A Data Probe may, however, be used by anyone that wishes to create and edit data that conforms to an information model. It is available as public domain software.

This document explains what Data Probes are, discusses what is involved in creating a Data Probe, explains how to run a Data Probe, and provides a detailed explanation of the commands needed to use a Data Probe.

List Of Figures

Data Probe User’s Guide

David A. Sauder

1 Introduction

Data Probes are software tools built using the emerging international Standard for the Exchange of Product Model Data[1] -- commonly referred to as STEP. STEP addresses the need to share data in a complex computer environment by providing a basis for common understanding and communication of product data. STEP includes definitions of information models[2] for various application areas. These information models communicate the structure and the semantics of the data necessary to inter-operate between different computer systems.

STEP specifies a mechanism for information modeling and mechanisms for the sharing of related data. STEP defines a specification language called Express [ISO11] that is used to represent the information models within STEP. STEP specifies a mechanism known as the Exchange File format [ISO21] for data exchange via computer data files called STEP exchange files. STEP exchange files contain data which is meaningful in the context of an associated information model written in Express. Data Probe tools are created from information models written in Express and use the STEP exchange file format for reading and writing data files.

A Data Probe is created for any information model written in Express. It is used to view information from the information model for which it was created. It is also used to create, edit, or view data corresponding to the information model for which it was created. It performs validation of data when reading STEP exchange files so it may also be used to validate STEP exchange files.

The software used to create Data Probe tools was built at the National Institute of Standards and Technology to assist in the development of STEP.[3] It is written in the C++ programming language on top of the X Window System [Scheifler89] using the InterViews [Linton91] user interface toolkit. It uses only public domain software and is available as public domain software. It is currently supported only for use on a Sun SPARCstation[4] workstation.

1. The Standard for the Exchange of Product Model Data (STEP) is a project of the International Organization for Standardization (ISO) Technical Committee on Industrial Automation Systems (TC184) Subcommittee on Industrial Data and Global Manufacturing Programming Languages (SC4). For an overview of the standard refer to *Part 1: Overview and Fundamental Principles* [ISO1].

2. An information model describes the data for an application area without all the details needed for computer implementation.

3. Funding for the work described has been provided by the Department of Defense’s Computer Aided Acquisition and Logistic Support (CALS) Office. The work is funded by the United States Government and is not subject to copyright.

A Data Probe can be created and used by anyone who wishes to create or edit data that conforms to an information model written in Express. Data Probes are used by developers of STEP and are part of the set of tools [Clark90] that are used in the National PDES Testbed[5].

This document explains what Data Probes are, discusses what is involved in creating a Data Probe, explains how to run a Data Probe, and provides a detailed explanation of the commands needed to use a Data Probe. Section 2 provides background information discussing the reasons for developing the Data Probe software. Section 3 explains how to create and run a new Data Probe. Section 4 provides an overview of the Data Probe and Data Probe windows. Section 5 explains the basics of dealing with the Data Probe user interface. Sections 6 through 10 provide detailed information on each of the types of the Data Probe windows. Section 11 is a tutorial for walking a user through the basics of using a Data Probe. Appendix A defines and discusses terms that are used throughout the document. Appendix B contains an information model written in Express which was used to create the Data Probe tool used in the figures and in the tutorial.

2 Background

The Data Probe software was written to assist in preparation of product data to be used in STEP validation testing of an information model. The software supports the editing, browsing, and formatting of product data, and browsing of the structure of information models. Further details of the methodology used for validation and requirements for automating that process are described in other documents [Mitch91][Morris91a][Morris91b][Mitch92][Morris92a][Morris92b].

The National PDES Testbed has been used for STEP validation testing since 1989. The early software which supported the testing process, the *interim system,* consisted of a set of tools which were not integrated. The tools in this system were collected from a variety of sources, and, as a result, they operated on a variety of hardware platforms and in a variety of software environments. In the interim system the user needed to be familiar with a number of executable programs and their interfaces. In addition, the method of sharing data throughout the testing process was through exchange of data files between the different tools. The data was represented in memory in one of a number of different types of data structures depending on the tool.

The Data Probe integrates the functions supported by the set of tools in the interim system. The integration of the functions more efficiently automates the testing process. The Data Probe software improves on the existing system in the following areas:

- the number of errors is reduced by reducing the frequency of data translation and human intervention;
- the amount of time needed for the testing process is reduced by improved performance and automation of the workflow;

4. No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied

5. PDES, Product Data Exchange using STEP, is the U.S. activity in support of the international STEP standard. The National PDES Testbed is located at NIST and is funded by the Department of Defense’s Computer-Aided Acquisition and Logistic Support (CALS) Office.

- the time needed for learning to use the software is reduced by providing a single user interface to the system;
- inconsistencies in the data are reduced by providing more sophisticated support for data editing and creation; and
- the potential for errors is reduced through better and more extensive error checking.

The interim system required data translation every time data was moved between activities in the testing process. The translation process introduced errors or inconsistencies, and the associated manual steps, such as importing and exporting the data, were time consuming. With respect to data editing, the interim system did not track which portions of the data were complete and which needed further development. Manual support for this function was extremely difficult, given the amount of data, and led to inconsistencies in the data. In addition, the manual configuration of the different versions of all the intermediate data files was error prone. The Data Probe software allows users to operate through a single interface, rather than with each tool separately.

3 Getting Started

This section describes the process of creating and running a Data Probe tool. If you create a Data Probe from the Express schema found in Appendix B your Data Probe tool will match the examples in this guide.

3.1 Creating a Data Probe from an Information Model

A Data Probe is built for a particular information model. The information model is specified using the Express specification language. This section provides an overview of the Data Probe creation process for an information model specific Data Probe. More details of the process are included with the Data Probe release.

Several software libraries [Morris92b] are required to create a new Data Probe. These libraries are listed below:

- Information model specific library - this library needs to be created for each information model specific Data Probe. This library contains information relating to the entity types that are within the information model and constructs for representing entity instances of those types. The software used to create this library is generated automatically by supplying an information model written in Express to a software application that translates Express into C++ programming language constructs.
- STEP core and editor library - the STEP core part of the library contains the C++ constructs that the library representing the information model is built on top of. The editor part of the library contains the data structures used to represent the editing information. These are conceptually two libraries but are physically one library.
- VTS user interface library - contains the VTS specific user interface portion of the Data Probe software as well as generic extensions to the InterViews library.
- Utility library - contains generic utility constructs and functions.
- gnu C++ library - a general purpose set of C++ classes.
- InterViews libraries - a generic X Window System user interface toolkit.

All of the libraries above excluding the first library are created once and reused for each Data Probe that is built. An information model specific Data Probe is built by creating the first library above and linking it together with the other libraries.

Included with the release of the Data Probe is a shell[6] script named *mkProbe* which will create a new Data Probe given an Express file and an information model name. The format for running the script is as follows:

mkProbe *express-file information-model-name*

Where *express-file* is the file containing the information model written in Express used to create the information model specific Data Probe and *information-model-name* is the name of the associated information model which is used as the name of a new directory that is created and is used in generating the name of the new Data Probe executable.

A new directory is created under the directory from which the script was run. This directory will have the same name as the *information-model-name* argument supplied when running the script. The new directory will contain the following:

1) the source code containing the C++ classes generated by fedex_plus from the information model written in Express found in the file supplied as the first argument, *express-file*,

2) the object files generated from the source code above,

3) the information model specific library created from the object files above,

4) your new Data Probe executable i.e., under the directory from where the script was run you will find the executable:

information-model-name/dp_*information-model-name*

The mkProbe script will let you know if it is successful or if it fails at creating a new Data Probe from the information model written in Express in *express-file*.

3.2 Running the Data Probe

Here is the command line format for running a Data Probe:

Data-Probe-executable-name [-saveFile *backup-file-name*] [-saveFreq *integer*]

Where *Data-Probe-executable-name* is the name of the Data Probe executable; *backup-file-name* is a name for the editing session backup file that overrides the default backup file name; and *integer* is a number which determines how often the editing session backup file is written. The brackets should not be typed. They indicate that the command line argument is optional.

The optional command line arguments affect the writing of the editing session backup file that the Data Probe regularly writes in the form of a working file. The first command line argument specifies the name of the backup file that is written where *backup-file-name* indicates the name of the backup file optionally including a directory path. There is another method for choosing a backup file name or path. It may be done using the “Choose Backup File” menu option from the “File Management” menu on the Data Probe window (see section 6.2.3). The name of the backup file defaults to dpAu-

6. the Unix shell csh

toSave.wf and is written to the path where the Data Probe was run. The second command line argument specifies how often the backup file will be written where *integer* is some number greater than zero. This number defaults to 30 and represents the number of X Window System events that should happen before a backup file is written.

Exiting the Data Probe is done using the Quit Menu accessed through the Quit menu bar located on the left of the Data Probe window (for more information see section 6.1).

4 Data Probe Overview

This section provides an overview of the Data Probe including an explanation of the concept of an information model specific Data Probe, a description of the windows found within a Data Probe, validation of instance data, editing states associated with instances, and the data files associated with a Data Probe.

4.1 A Data Probe is Information Model Specific

A collection of software [Morris92b] is used to build an information model specific Data Probe. The information model must be written in Express and must be compiled by a software application called fedex_plus. Fedex_plus generates code that is used with the other software to create a Data Probe tool. A Data Probe tool will be referred to as simply -- a Data Probe. All the software needed to create a Data Probe is included in the distributed release.

A Data Probe contains an *entity type list* and an *entity instance list* that are specific to the information model for which it was created. The information model defines entity types that represent objects within the model. A Data Probe has an *entity type list* which contains the list of entity types that were defined in the information model. Data Probe also provides functionality for displaying information about the entity types that are in its list of entity types. The entity types defined in the information model provide a framework around which entity instances have meaning just as the instances defined in Figure 2 of Appendix A have meaning based on the entity types defined in Figure 1 of Appendix A.

Data Probe maintains a list of entity instances called the *entity instance list* that correspond to the entity types known to it. It provides functionality for manipulating the individual instances and the list of instances. Entity instances may be created interactively within a Data Probe from its list of entity types or read from a file. The instances may then be viewed or edited from within the Data Probe and written to a file.

4.2 Data Probe Windows

A Data Probe contains five types of windows: (identified in Figure 1)

1. Data Probe window - contains menus of commands for quitting the Data Probe, reading and writing files, and other functions. It also contains a list of the Express schema names the Data Probe was created to work with, quick reference information, and a message area for communicating results of commands to the user.
2. Entity Type List window - contains a scrollable list of all the entity types that were defined within the information model for which the Data Probe was created.

3. Entity Instance List window - contains a scrollable list of entity instances. These instances may be created from the list of entity types in the Entity Type List window or read in from a file.
4. STEP Entity Descriptor window - used for displaying information about an entity type from the information model.
5. STEP Entity Editor window - used for editing an entity instance. There may be several of these windows open for separate instances.

The first three windows appear on the screen when the Data Probe is run. The Data Probe window appears centered at the top of the screen. The Entity Instance List window and the Entity Type List window appear at the bottom right and bottom left corners of the screen respectively. Several of the STEP Entity Editor windows and STEP Entity Descriptor windows may exist at any time. Figure 1 shows the start of a Data Probe editing session with a STEP Entity Editor window on top of the Entity Instance List window and a STEP Entity Descriptor window on top of the Entity Type List window. The Data Probe shown was created from the information model found in Appendix B.

The Entity Type List window provides commands for creating new entity instances. This window also allows a user to display information about an entity type by creating a STEP Entity Descriptor window. Several STEP Entity Descriptor (SED) windows may be open at one time each displaying information for a separate entity type.

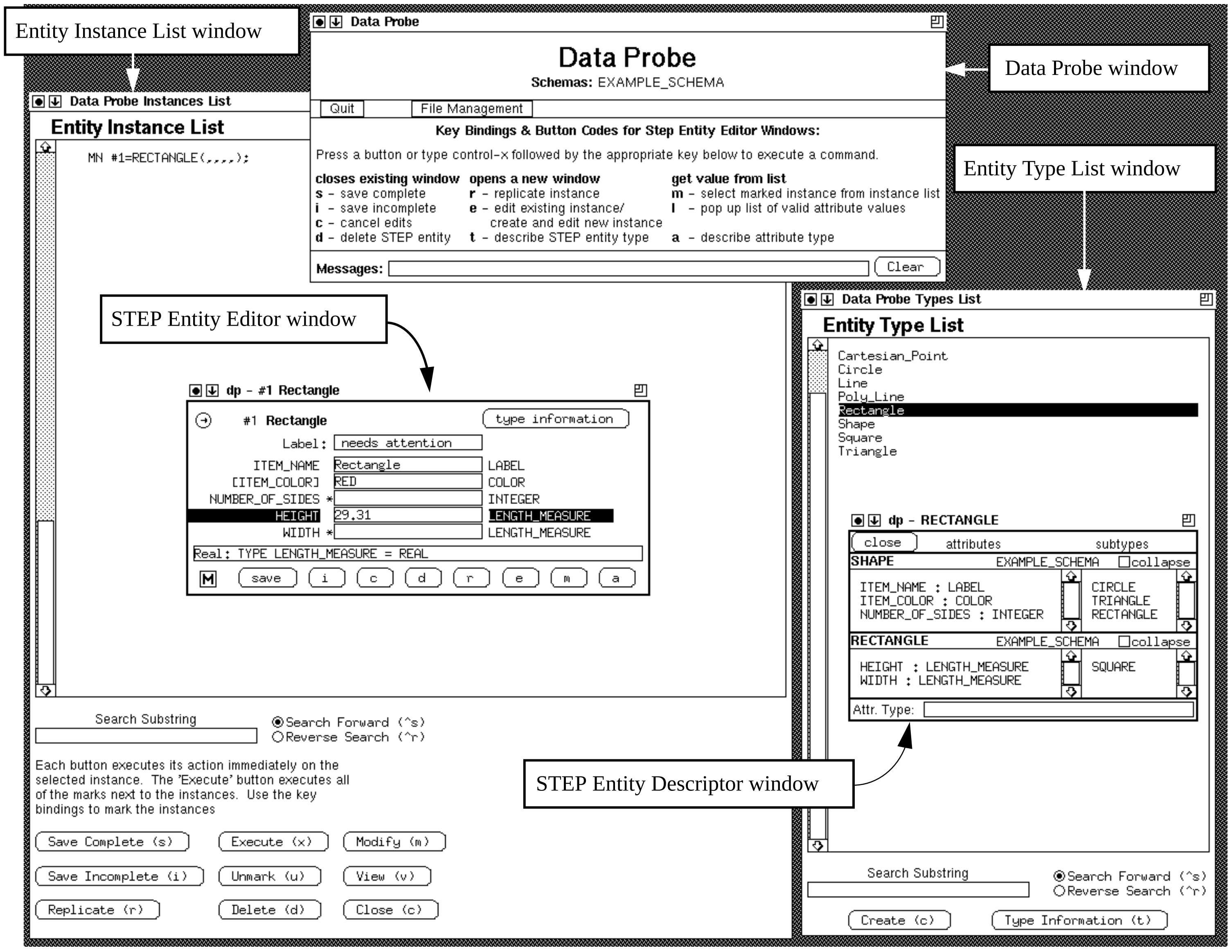
The Entity Instance List window contains a scrollable list of entity instances. The format of this list is based on the STEP exchange file format [ISO21]. There are marks next to each instance displayed in this window. The marks show information about the editing state of the instance or show intended commands to be executed on the instance. This window allows the user to issue commands that affect entity instances. One of the commands is to edit the instance by creating a STEP Entity Editor window. Several STEP Entity Editor (SEE) windows may be on the screen at one time. The STEP Entity Editor window allows a user to edit the values of attributes for an instance. It validates the attribute values (see the next section) and informs the user of incorrect data. It has many features which help the user in the editing process.

One more window that has an important use during a Data Probe editing session is the window from which the Data Probe was run. This window is used by the Data Probe to communicate detailed warnings and error messages to the user and supplements the short messages communicated within the Data Probe.

4.3 Instance Validation

Each entity type has associated with it zero or more attributes. The value of a particular attribute within an entity instance is defined in Express to be from a certain domain of values. An attribute’s type represents the domain of values that are valid for the attribute. For example, an attribute which is of type integer will require that its value be an integer. Other values would be considered invalid. The Data Probe assists the user in keeping attribute values valid by validating them against the attribute’s type.

Attributes are also defined in Express to be either optional or required attributes. Required attributes must contain a value or the attribute is said to be invalid. Optional attributes are just that -- optional -- meaning the attribute may or may not contain a value. In addition to validating based on attribute type, the Data Probe also validates



MN #1=RECTANGLE(,,,,);

Data Probe
Schemas: EXAMPLE_SCHEMA

Key Bindings & Button Codes for Step Entity Editor Windows:

Press a button or type control-x followed by the appropriate key below to execute a command.

closes existing window	opens a new window	get value from list
s – save complete	**r** – replicate instance	**m** – select marked instance from instance list
i – save incomplete	**e** – edit existing instance/ create and edit new instance	**l** – pop up list of valid attribute values
c – cancel edits		
d – delete STEP entity	**t** – describe STEP entity type	**a** – describe attribute type

Messages:

#1 **Rectangle**

Label:	needs attention	
ITEM_NAME	Rectangle	LABEL
[ITEM_COLOR]	RED	COLOR
NUMBER_OF_SIDES *		INTEGER
HEIGHT	29.31	LENGTH_MEASURE
WIDTH *		LENGTH_MEASURE

Real: TYPE LENGTH_MEASURE = REAL

Cartesian_Point
Circle
Line
Poly_Line
Rectangle
Shape
Square
Triangle

Each button executes its action immediately on the selected instance. The 'Execute' button executes all of the marks next to the instances. Use the key bindings to mark the instances

SHAPE EXAMPLE_SCHEMA

ITEM_NAME : LABEL
ITEM_COLOR : COLOR
NUMBER_OF_SIDES : INTEGER

CIRCLE
TRIANGLE
RECTANGLE

RECTANGLE EXAMPLE_SCHEMA

HEIGHT : LENGTH_MEASURE
WIDTH : LENGTH_MEASURE

SQUARE

Attr. Type:

FIGURE 1 Start of a Data Probe Editing Session

based on an attribute being optional or required. Any attribute value that exists must match its required type. Any attribute that is required must contain a value. The Data Probe will only save edited attribute values within the instance that are valid[7]. Invalid attribute values will not be permitted to be saved.

7. The Data Probe at present does not do validation on the basis of WHERE clauses in Express.

4.4 Instance Editing States

Associated with each instance being managed by the Data Probe is an editing state. The editing state provides information about the instance as follows:

- Complete- All required attributes within the instance exist and all attribute values are valid.
- Incomplete - Required attributes within the instance possibly do not have values and all attributes that have values are valid. Having this state allows attribute values to be saved even though the entity instance as a whole is incomplete. It also provides a way for a user to mark an instance as needing further attention even though all required attributes values exist.
- Delete - The entity instance is marked for deletion.
- New - The entity instance is newly created from from the entity type list. If any attribute values have been entered into a SEE window for this entity instance they have not been saved. Instances retain the editing state of *new* until they are deliberately changed from a new state (e.g. by saving the instance after editing it).

4.5 Data Probe Files

There are three types of files associated with the Data Probe. These are summarized below:

- STEP Exchange files - These files require that all entity instances be valid. All required attributes in the entity instances must contain values and all attribute values must be valid.
- Working Session files - These files allow a user to save work in progress. They require all attribute values that exist to be valid but allow required attributes in entity instances to be missing. Along with each instance, Working Session files save the editing state information defined above. When this type of file is read, instances that are marked for deletion will be deleted and references to these instances by other instances will be removed.
- a backup file - A backup file of the current entity instance list is written regularly during a Data Probe editing session in the form of a Working Session file. This is done in order to permit a recovery of the work done in the event of mishap. The name of the backup file defaults to dpAutoSave.wf and is written to the directory where the command to run Data Probe was issued.

4.5.1 STEP Exchange Files and Working Session Files

The Data Probe provides commands for writing STEP exchange files and Working Session files. It will only allow STEP exchange files to be written when all of the entity instances are complete and valid. A list of entity instances that contain instances with missing required attributes can be written to a Working Session file only. A command for validating the entity instances in the list of instances is provided that will print out appropriate warnings and error messages in the window where the Data Probe was run.

When reading and writing STEP exchange files and Working Session files appropriate warnings and error messages will be printed out in the window where the Data Probe was run. When reading a STEP exchange file, instances that are invalid will be read but will have an incomplete editing state assigned to them and will be marked as such in the list of instances. As many attribute values as possible will be read for invalid instances when reading a STEP exchange file.

4.5.2 Appending Files

The Data Probe provides commands for replacing its current list of entity instances by reading in a new list from a STEP exchange file or Working Session file. It also provides a command to append a list of instances to its current list by reading a list from a STEP Exchange file or Working Session file. Each entity instance has associated with it a unique numeric instance identifier. When appending a list of entity instances from a file, the instance identifiers for the instances being read are adjusted by a numeric constant in order to avoid conflict with identifiers that exist in the current list of instances. Any attribute values for the entity instances being read that reference other entity instances are adjusted by the same numeric constant to keep the references correct.

4.5.3 Backup File

The Data Probe writes regular backup files in the form of a working file. The name of the backup file defaults to *dpAutoSave.wf* and is written to the directory from where the command to run the Data Probe was issued. The user may specify a different name or path for the backup file using two different methods. First the Data Probe may be run with two command line arguments that affect the writing of the backup file. Either, none, or both of these options may be specified following the Data Probe executable when running the Data Probe:

- -saveFile <backup file name>
where <backup file name> is the new name for the backup file.

- -saveFreq <integer>
where <integer> is a number which determines how often the backup file is written. This number defaults to 30 and represents the number of X Windows events that should happen before a backup file is written.

The other method for choosing a backup file name or directory path is using the “Choose Backup File” menu option from the “File Management” menu as described in section 6.2.3.

5 User Interface Basics

This section describes the basics of the Data Probe user interface including command basics, using the mouse, pressing buttons, choosing commands from a menu, editing text, and dealing with lists.

5.1 Data Probe Command Interface

Much of the Data Probe user interface offers two methods of issuing a command. The first method is using keystrokes and the other method is using the mouse to press buttons or select a command from a menu. Some of the commands will execute immediately while others will execute in batch mode (a group of commands will be executed when told to do so). Commands executed using the mouse to press buttons, or choose menu options will execute immediately. Some commands issued from a menu have an additional step involved like choosing a file name. When another step is involved, a chance will be given to quit the command without executing it.

Keystroke commands may be single character commands or character sequences. Control characters are involved in multiple character commands. A control character is typed by simultaneously pressing the key labeled *control* and the other character involved. Control characters are identified like this:

^<character>

where <character> is another character from the keyboard. For example the command to save an entity instance when editing the instance from a STEP Entity Editor window is done using the key sequence ^x s. This command is executed by pressing and holding *control* and while still holding *control* pressing *x* then releasing all characters and pressing *s*.

5.2 Using the Mouse

In addition to providing keystroke commands, the Data Probe user interface is built to take advantage of the intuitive use of a mouse. The mouse may be used to select entries in lists, press buttons in a window, choose commands from a menu, scroll lists or text, place the cursor within text, and highlight text to be deleted.

Located on the mouse are three buttons. These buttons will be referred to as the left, middle, or right mouse button as they are physically located on the left, middle, and right side of the mouse. Pressing, selecting, or clicking with the mouse refers to putting the mouse cursor over an object in a Data Probe window and pressing one of the mouse buttons. Any of the three mouse buttons may be used to press a button found in a Data Probe window or select a command from a menu. It is a good idea to get used to selecting with the left mouse button so it will not conflict with scrolling which is implemented using the middle and right mouse buttons.

Associated with the mouse is a cursor on the computer screen which is referred to as the mouse cursor. This cursor on the screen follows the movements of the mouse. The shape of the mouse cursor changes depending on what command is being executed, what action is being done with the mouse, and where the mouse is. Here are the mouse cursors and what they identify:

- arrow pointing northwest - this is how the mouse cursor usually looks.
- hour glass - this means a command is waiting to finish (like waiting for a file to be read).
- menu - the mouse cursor has been placed over a menu.
- hand - the mouse cursor changes to a hand when it moves over a button or when scrolling a list or single line of text using grab scrolling (see sections 5.5 and 5.6) with the middle mouse button.
- double triangle pointing up or down - the mouse cursor changes to a double triangle pointing up or down when scrolling a list vertically using rate scrolling (see section 5.6) with the right mouse button. It points up when scrolling a list to display entries that are higher up in the list. It points down when scrolling a list to display entries that are lower in the list.

- double triangle pointing left or right - the mouse cursor changes to a double triangle pointing left or right when scrolling a single line of text horizontally using rate scrolling (see section 5.5) with the right mouse button. It points left when scrolling the text to display text that precedes the current text. It points right when scrolling the text to display text that follows the current text.

Another very important use of the mouse is selecting the window in which you are planning to work. When you decide to work in a different window than the one you are currently in you must move the mouse cursor to that window and click inside that window. This is called changing window focus. Some areas within a window demand focus also and may require that you click with the mouse inside of that area of the window and leave the mouse cursor inside of that area.

5.3 Pressing Buttons

There are screen buttons that are located on the computer screen within a window of the Data Probe and there are buttons that are physically located on the mouse. Referring to a button will refer to a screen button that is part of the Data Probe user interface that appears on the computer screen. Referring to a mouse button will refer to one of the three buttons that are physically located on the mouse.

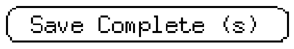
A button should be identifiable by sight. A button is also identified by looking at the mouse cursor. The mouse cursor changes to a hand when it is placed over a button.

Pressing a button in a window of the Data Probe activates the function associated with the button. Pressing a *mouse* button means clicking one of the three physical buttons that are located on the mouse. A screen button is pressed using the following sequence:

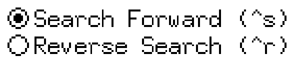
1. place the mouse cursor over the button in a Data Probe window.
2. press a mouse button.
3. with the mouse cursor still inside of the button release the mouse button.

Any mouse button may be clicked to press the button. If after pressing a mouse button when the mouse cursor is inside a screen button you decide you don't want to press the screen button, simply remove the mouse cursor from inside the screen button and release. The button will not be pressed. All of the buttons in the Data Probe will change how they look when being pressed.

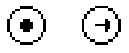
There are three types of buttons found in Data Probe windows:



Text Buttons - these are buttons with text inside of them that are pressed and released.



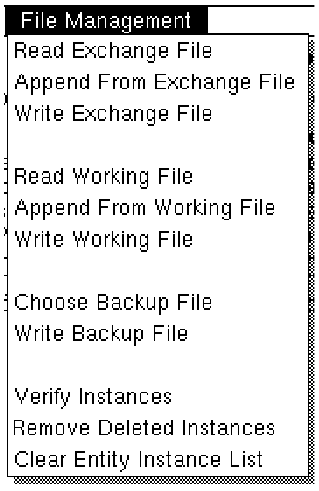
Radio Buttons - these types of buttons exist in groups of two or more where only one may be turned on at a time like the buttons on an old car radio. These buttons are usually used to set a state.



Toggle Buttons - these types of buttons toggle (switch) between two values (or states) when pressed. They redraw themselves to indicate their current state. The example to the left shows a 'pin' toggle button drawn as it would look if *stuck in* followed by the button as it would look if redrawn as *pulled out*.

5.4 Choosing Commands from Menus

The Data Probe uses pull-down menus. A pull-down menu is accessed from a box with the name of the menu inside it called a menu bar. When the mouse cursor is placed over the menu bar it changes to look like a menu. When a mouse button is pressed and held



down while the mouse cursor is inside the menu bar, the menu pulls down or draws itself beneath the menu bar.

A command may be executed from a menu using the following sequence:

1. Place the mouse cursor over the pull-down menu.

2. Press and hold one of the mouse buttons. The pull-down menu will appear.

3. While still holding the mouse button down, move the mouse cursor over the menu command that you wish to execute. As you move the mouse cursor over the menu commands each command will highlight itself when the mouse cursor is over it indicating that it is the currently selected choice to be executed.

4. Release the mouse button when the menu command you wish to execute is highlighted.

If you decide not to execute a command from the menu simply remove the mouse cursor from inside the menu and release the mouse button.

5.5 Editing Text

Editing is done within an editable portion of a window referred to as an editable box. The keystroke commands correspond to keystroke commands in the emacs editor where possible. The following commands may be used to facilitate the editing of text in editable boxes within the Data Probe:

■ left mouse button - used to initiate editing in an editable box and to move the cursor within an editable box as follows:

1.) Point the mouse cursor inside an editable box in a window.

2.) Click the left mouse button to place the cursor at the desired position inside the text in the editable box.

■ left mouse button - used to select a portion of text to be deleted inside an editable box:

1.) Point the mouse cursor inside an editable box in a window. Click and hold the left mouse button.

2.) Drag the mouse to the left or right to highlight text you wish to delete as appropriate. The text will scroll to display more text as necessary.

3.) To delete the selected text: press the delete key or input a value to replace the selected text.

■ middle mouse button - *horizontal grab scroll* - works like it would if you were to grab and scroll the list with your hand.

1.) Point the mouse inside the text.

2.) Press and hold the middle mouse button.

3.) Slide the mouse left or right to display more of the text preceding or following the viewable text respectively.

■ right mouse button - *horizontal rate scroll*: the farther the mouse is slid from its original position, the faster the rate of scrolling.

1.) Point the mouse inside the text.

2.) Press and hold the right mouse button.

3.) Slide the mouse left to display more of the text preceding the mouse cursor or slide the mouse right to display more of the text following the mouse cursor.

- ^f - move cursor forward.
- ^b - move cursor backward.
- ^e - move cursor to the end.
- ^a - move cursor to the beginning.
- <esc> f - move cursor forward a word.
- <esc> b - move cursor backward a word.
- ^d - delete a character in place.
- <delete> or ^h - delete the previous character.
- ^w - select the remainder of the word following the cursor to be deleted.
- ^k - select everything following the cursor to be deleted (kill rest of line).
- ^u - select the entire text to be deleted.

5.6 Dealing with Lists

The keystroke commands correspond to keystroke commands in the emacs editor where possible.

- left mouse button double click - double clicking the left mouse button has different effects. The effect depends on the functionality defined for it by the receiving list. Double clicking means to quickly click one of the mouse buttons twice over an entry in the list.

5.6.1 Selecting an Entry in a List

- left mouse button - point the mouse cursor over the entry and click the left mouse button.
- ^n - select next entry.
- ^p - select previous entry.
- < - select top entry.
- > - select bottom entry.
- <delete> - unselect selected entry (no entry is selected).
- ^h - unselect selected entry (no entry is selected).

These two apply only to the entity instance list and the entity type list:

- ^s - search forward - selects the entry containing the substring in the search window.
- ^r - reverse search - selects the entry containing the substring in the search window.

5.6.2 Scrolling a List

- click inside the scroll bar to scroll the list. (For lists with scroll bars attached.)
- click (and optionally hold) the up mover or down mover located above and below the scroll bar respectively to scroll the list. (For lists with scroll bars attached.)
- middle mouse button - *vertical grab scroll* - works like it would if you were to grab and scroll the list with your hand.

1.) Point the mouse cursor inside the list (not the scroll bar).

2.) Press and hold the middle mouse button.

3.) Slide the mouse up or down to display more of the list from below or above respectively.

- right mouse button - *vertical rate scroll* - the farther you slide the mouse from its original position, the faster the rate of scrolling.

  1.) Point the mouse cursor inside the list (not the scroll bar).

  2.) Press and hold the right mouse button.

  3.) Slide the mouse down to display more of the list from below the mouse cursor or slide the mouse up to display more of the list from above the mouse cursor.
- ^a - scroll list to beginning.
- ^e - scroll list to end.
- ^b - scroll up (backward).
- ^f - scroll down (forward).
- ^k - scroll up (like the vm editor).
- ^j - scroll down (like the vm editor).
- ^v - page down.
- <space> - page down.
- ^d - half page down.
- ^u - half page up.

5.6.3 Entity Type List and Entity Instance List Search

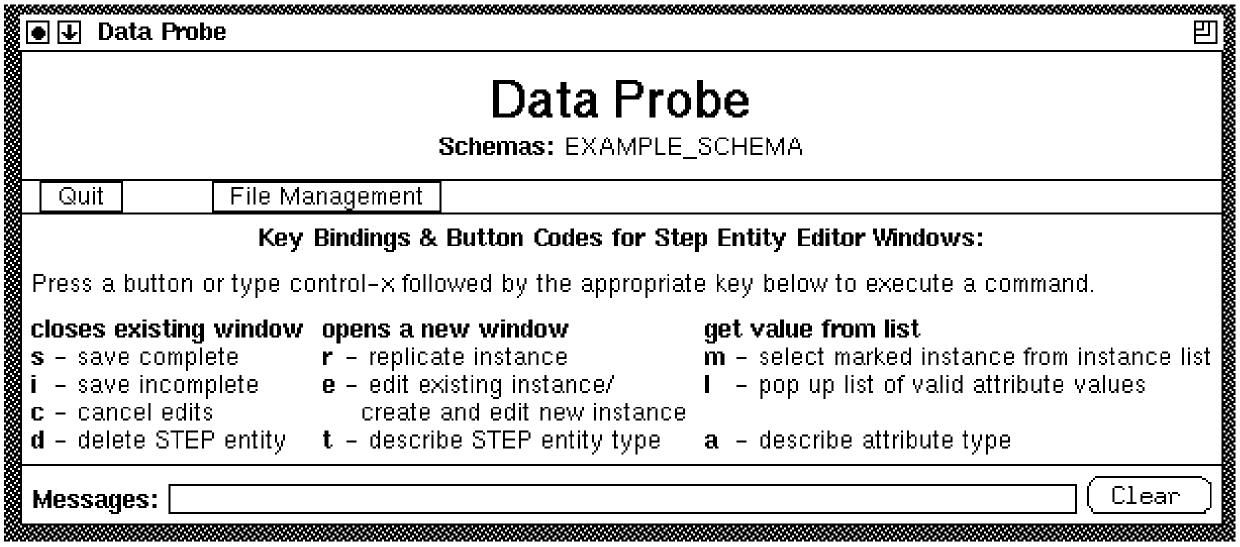
The Entity Instance List window and the Entity Type List window contain a mechanism for case sensitive searching of the entity instance list and entity type list respectively. Searching is implemented via an editable search string box, two radio buttons, and keystroke commands . The radio buttons toggle between search forward and search backward. The keystrokes will only work if the mouse has been clicked and remains inside either the list part of the window or the search string editable box. Three keystrokes cause the search command to be executed from the currently selected entry in the Entity Instance List or Entity Type List:

- return - causes the search to go in the direction of the currently selected search radio button.
- ^s - causes a forward search regardless of the search button selected.
- ^r - causes a backward (reverse) search regardless of the search button selected.

6 Data Probe Window

The Data Probe window contains four sections. Figure 2 shows the Data Probe window for a Data Probe created from the information model found in Appendix B. The section at the top of this window identifies the Data Probe. It contains the name 'Data Probe' in bold letters followed by a list of the schemas that the Data Probe was created for. The section below the schema list contains a menu bar with two pull-down menus described below. The third section explains the use of the buttons and key bindings within STEP

Entity Editor windows. The final section contains a message area for communicating results of commands to the user. It also contains a button labelled ‘clear’ for clearing messages from the message area.



Schemas: EXAMPLE_SCHEMA

Key Bindings & Button Codes for Step Entity Editor Windows:

Press a button or type control-x followed by the appropriate key below to execute a command.

closes existing window	opens a new window	get value from list
s – save complete	**r** – replicate instance	**m** – select marked instance from instance list
i – save incomplete	**e** – edit existing instance/ create and edit new instance	**l** – pop up list of valid attribute values
c – cancel edits		
d – delete STEP entity	**t** – describe STEP entity type	**a** – describe attribute type

Messages:

FIGURE 2 Data Probe Window

6.1 Quitting a Data Probe session -- the Quit Menu

The menu furthest to the left of the Data Probe window entitled “Quit Menu” is used for quitting a Data Probe editing session. There is no warning about saving a file when the Data Probe is quit. When the ‘Quit’ option is chosen the Data Probe application quits.

6.2 The File Management Menu

The menu furthest to the right in this window entitled “File Management” is used for dealing with STEP Exchange files, Working Session files, the editing session backup file, and activities relating to the list of entity instances. The three types of files are described in detail in section 4.5. The “File Management” menu options are described below.

6.2.1 Reading and Writing Exchange Files

The following “File Management” menu options deal with STEP Exchange Files:

- Read Exchange File - Reads in a STEP Exchange file and replaces all the entity instances in the Entity Instance List window with the instances read in from the STEP Exchange file.
- Append From Exchange File - Reads in a STEP Exchange file appending the newly read entity instances to the end of the list of existing instances in the Entity Instance List window. The instance identifiers of the newly read instances and the instance identifiers of the entity references within the newly read instances are all adjusted up by a constant amount so that there will not be referencing conflicts.

- Write Exchange File - Writes out a STEP Exchange file from the entity instance list. Changes to instances that have been made inside STEP Entity Editor windows that have not been saved will not be reflected in the STEP Exchange file that is written.

6.2.2 Reading and Writing Working Session Files

The following "File Management" menu options deal with Working Session Files:

- Read Working File - Reads in a Working Session file and replaces all the entity instances in the Entity Instance List window with the instances read in from the Working Session file. Instances that have an editing state of *Delete* will not be read in and any references by attributes of other instances to instances that are deleted will be made null.
- Append Working File - Reads in a Working Session file appending the newly read entity instances to the end of the list of existing instances in the Entity Instance List window. The instance identifiers of the newly read instances and the instance identifiers of the entity references within the newly read instances are all adjusted up by a constant amount so that there will not be referencing conflicts.
- Write Working File - Writes out a Working Session file from the entity instance list. Changes to instances that have been made inside STEP Entity Editor windows that have not been saved will not be reflected in the Working Session file that is written.

6.2.3 Dealing With the Editing Session Backup File

The following "File Management" menu options deal with the editing session backup file:

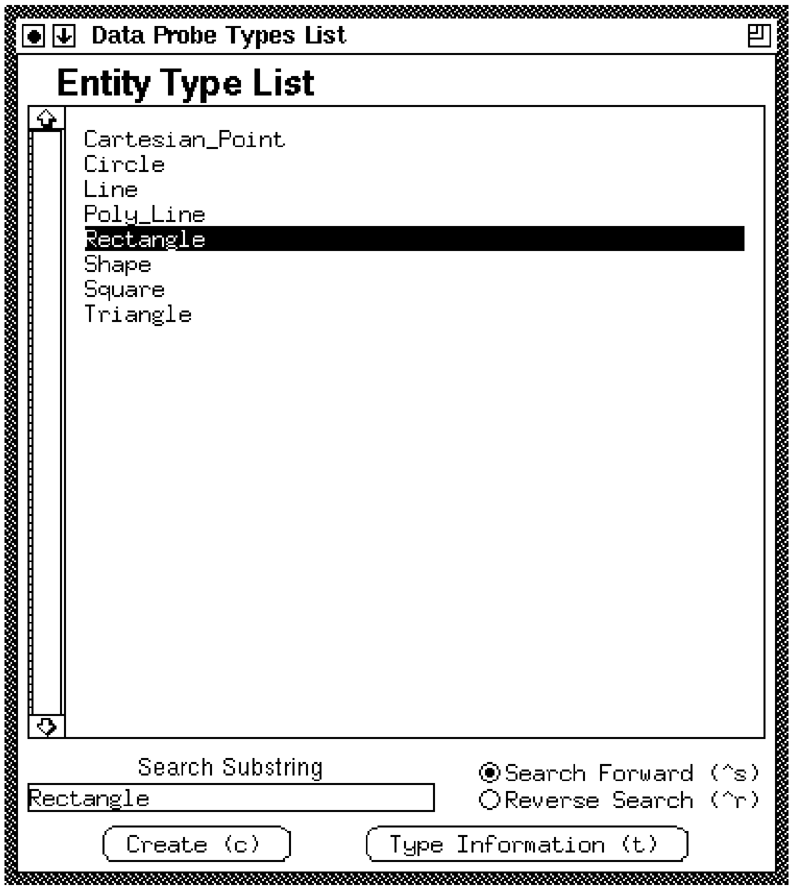
- Choose Backup File -Allows the user to select a new name or path for the backup file. The default backup file name is *dpAutoSave.wf*
- Write Backup File - Allows the user to force a backup file to be saved.

6.2.4 Deleting Instances, Validating or Clearing the Entity Instance List

- Validate Instances - Validates the instances in the Entity Instance List window. A list of the instance identifiers of the invalid instances will be printed in the window where the Data Probe was run and a message that tells how many entity instances were invalid will be sent to the Messages window in the Data Probe window.
- Remove Deleted Instances - Removes all entity instances that have an editing state of *Delete* and removes all references to the deleted entity instances by other instances. This is done by writing and reading a Working Session file with a default name.
- Clear Entity Instance List - Deletes all the entity instances known to the Data Probe and clears them from the entity instance list in the Entity Instance List window.

7 Entity Type List Window

The Entity Type List window contains a list of all the entity types that were defined in the information model written in Express from which the Data Probe was created. This scrollable list, referred to as the *entity type list*, comprises the main part of the window. Figure 3 shows the Entity Type List window from a Data Probe created from the information model found in Appendix B. This window allows a user to create new entity instances and display information about entity types by placing STEP Entity Descriptor windows on the screen.



Cartesian_Point
Circle
Line
Poly_Line
Rectangle
Shape
Square
Triangle

Search Substring: Rectangle

[✓ Search Forward (^s)] Reverse Search (^r)

FIGURE 3 Data Probe Entity Type List Window

7.1 Keystrokes Commands

The entity type list part of this window accepts keystroke commands from two groupings. The first set includes keystroke commands for selecting entity types in the list and navigating the list (see section 5.6). These are the same keystroke commands that apply to the Entity Instance List window. Included also in this group of commands are commands for searching the list (see below). The second set of commands are used for displaying information about the entity types and creating new instances that will appear in the entity instance list in the Entity Instance List window. Commands in this window do not leave marks as they do in the Entity Instance List window.

7.1.1 Search Keystrokes

This window contains a mechanism for case sensitive searching of the entity type list. Searching is implemented via an editable search string box, two radio buttons, and keystroke commands (see Figure 3). The radio buttons toggle between search forward and search backward. The keystrokes will only work if the mouse has been clicked and remains inside either the entity type list part of the window or the search string editable box. Three keystrokes cause the search command to be executed from the currently selected type in the entity type list:

- return - causes the search to go in the direction of the currently selected search radio button.
- ^s - causes a forward search regardless of the search button selected.
- ^r - causes a backward search regardless of the search button selected.

7.1.2 Displaying Entity Type Information and Creating New Instances

The following keystrokes are executed immediately after the keystroke is typed:

- c - create a new entity instance of the selected entity type and place a STEP Entity Editor window on the screen for it. The new instance will appear at the bottom of the entity instance list in the Entity Instance List window. The instance will be preceded by the letter *N* to signify that the instance is newly created.
- t - type information. That is, place a STEP Entity Descriptor window on the screen displaying information from the information model about the selected entity type.

7.2 Command Buttons

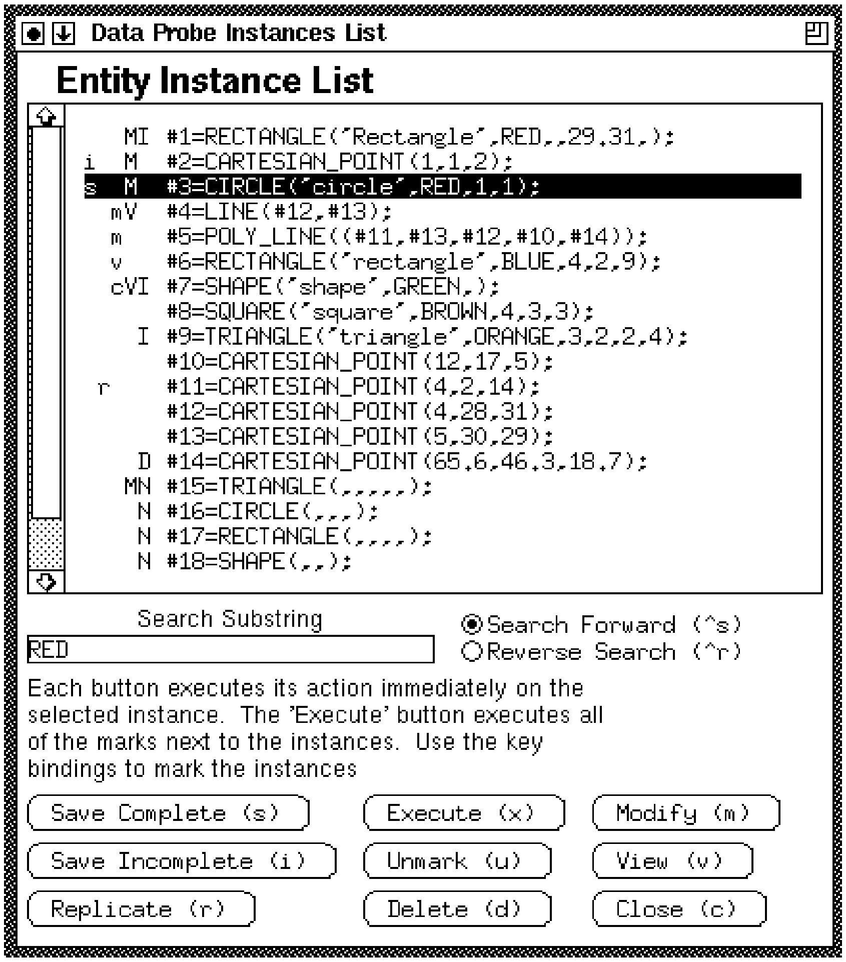
At the bottom of this window are buttons that provide an alternate way of executing commands from the keystroke commands listed above. To execute commands using the buttons the user simply selects an entity type from the entity type list using the mouse (by pointing the mouse cursor on the type and left clicking) and again using the mouse presses the appropriate button. Pressing the button will cause the command to be executed immediately against the selected entity type. Each button has the corresponding keystroke command for it in parenthesis in the button text. The following buttons are available:

- Create (c) - create a new entity instance of the selected type and pop up a STEP Entity Editor window for it. The new instance will appear at the bottom of the Entity Instance List with the letter ‘N’ next to it.
- Type Information (t) - place a STEP Entity Descriptor window on the screen displaying information from the information model about the selected entity type.

Double clicking the left mouse button over an entry for an entity type in the entity type list is like selecting that entity type and pressing the *Create (c)* button. It immediately creates an instance of the selected type, adds the instance to the end of the entity instance list, and places a modifiable SEE window on the screen for the newly created instance.

8 Entity Instance List Window

The Entity Instance List window is used to manage the list of all the entity instances that exist in a Data Probe editing session. Thus, the main part of this window is made up of a scrollable list of these instances called the *entity instance list*. Figure 4 shows the Entity Instance List window from a Data Probe created from the information model found in Appendix B.



```
   MI #1=RECTANGLE('Rectangle',RED,,29,31,);
i  M  #2=CARTESIAN_POINT(1,1,2);
s  M  #3=CIRCLE('circle',RED,1,1);
  mV  #4=LINE(#12,#13);
  m   #5=POLY_LINE((#11,#13,#12,#10,#14));
  v   #6=RECTANGLE('rectangle',BLUE,4,2,9);
  cVI #7=SHAPE('shape',GREEN,);
      #8=SQUARE('square',BROWN,4,3,3);
    I #9=TRIANGLE('triangle',ORANGE,3,2,2,4);
      #10=CARTESIAN_POINT(12,17,5);
 r    #11=CARTESIAN_POINT(4,2,14);
      #12=CARTESIAN_POINT(4,28,31);
      #13=CARTESIAN_POINT(5,30,29);
    D #14=CARTESIAN_POINT(65.6,46.3,18.7);
   MN #15=TRIANGLE(,,,,,);
    N #16=CIRCLE(,,,);
    N #17=RECTANGLE(,,,,);
    N #18=SHAPE(,,);
```

Search Substring: RED

[✓ Search Forward (^s)] Reverse Search (^r)

Each button executes its action immediately on the selected instance. The 'Execute' button executes all of the marks next to the instances. Use the key bindings to mark the instances

FIGURE 4 Data Probe Entity Instance List Window

The Entity Instance List window provides for manipulating instances using either keystroke commands or command buttons. Keystroke commands are not executed immediately. The result of issuing a keystroke command is a lower case letter called a command mark being placed in the entity instance list preceding the selected instance. The command mark is the same as the keystroke command that was issued. The commands associated with the command marks are executed when the *execute (e)* button is pressed or the *execute* keystroke command is issued.

At the bottom of the Entity Instance List window are command buttons that may be executed against the selected instance in the entity instance list. Commands executed as a result of pressing one of the command buttons are executed immediately. These command buttons are discussed at the end of this section. A keystroke command is associated with each of the command buttons. The associated keystroke command is indicated in parenthesis as part of the text found on the button.

Preceding each instance in the entity instance list are five columns. The first three columns hold the command marks referred to above that are the result of keystroke commands being issued against the instance. These keystroke commands are discussed later in this section. The last two columns preceding the instance contain marks indicating editing information about the instance. The marks that appear in the last two columns preceding the instance appear as upper case letters. The editing information marks are discussed next.

8.1 Editing Information Marks

Each entity instance in the entity instance list may have an upper case letter in one of two columns preceding it indicating editing information. The first column of upper case letters contains an indication of whether there is a STEP Entity Editor window on the screen displaying the instance.

A SEE window is created for an instance the first time it is placed on the screen. The SEE window will continue to exist until the instance is deleted. The SEE window may be removed from the screen or placed back on the screen several times during an editing session. Any changes that are made to the instance’s attribute values inside the SEE window will not be reflected in the entity instance list until those changes have been saved. Changes to attribute values inside a SEE window that have not been saved will be left intact inside the SEE window even though the window is removed from the screen. The next time the SEE window is placed on the screen the changes made to the attribute values will still be there and may be saved. The following information marks are mutually exclusive indicating whether a SEE window is on the screen for the instance in the entity instance list:

- no mark - no STEP Entity Editor window is on the screen displaying the instance.
- M : Modify - an editable STEP Entity Editor window displaying the instance is on the screen. Edited attribute values may not have been saved to be reflected in the entity instance list.
- V : View-only - a view-only STEP Entity Editor window displaying the instance is on the screen. Edited attribute values may not have been saved to be reflected in the entity instance list before the STEP Entity Editor window was made view-only.

Once a STEP Entity Editor window is placed on the screen it is under the control of your X Window System window manager. You may use your window manager to manipulate the window as you do other windows on the screen (e.g. you may move or iconify the window). The Entity Instance List window may be used to open and close STEP Entity Editor windows.

The second column of upper case letters indicates the editing state of the instance. These are also the marks that appear after reading or appending a Working Session file. These marks are also mutually exclusive:

- no mark : Complete - the instance has been saved to a complete state meaning that all required attributes exist and all attribute values are valid.
- I : Incomplete - the instance has been saved to an incomplete state meaning that all attribute values are valid with the exception that attribute values may be missing from optional and required attributes instead of just optional attributes. Any attribute value that does not pass validation will not be saved. Having this state allows attribute values to be saved for an instance even though the instance is incomplete. It also provides a way for a user to mark an instance as needing further attention even though all required attributes exist.

- D : <u>D</u>elete - the instance will be deleted when the “Remove Deleted Instances” menu option is chosen from the Data Probe window’s “File Management” menu. If a Working Session file is written these entity instances will be deleted when the Working Session file is read back into the Data Probe.
- N : <u>N</u>ew - the instance is newly created. If the instance was created since a file has been read then there are attribute values associated with the instance in a STEP Entity Editor window that have not been saved. These attribute values may be accessed by looking at the STEP Entity Editor window that is on the screen or placing the one that exists for the instance on the screen again.

8.2 Keystroke Commands

The entity instance list part of this window accepts two kinds of keystroke commands: ones that leave command marks and ones that do not. The first set includes keystroke commands for selecting instances and navigating the list (see section 5.6). These are the same keystroke commands that apply to the entity type list in the Entity Type List window. Included also in commands for selecting instances are commands for searching the list (see below). Commands for selecting instances and navigating the list do not leave command marks. The second set of commands accepted by the entity instance list leave command marks and are used for manipulating instances.

8.2.1 Search Keystrokes

The Entity Instance List window contains a mechanism for case sensitive searching of the entity instance list. Searching is implemented via an editable search string box, two radio buttons, and keystroke commands (see Figure 4). The radio buttons toggle between search forward and search backward. The keystrokes will only work if the mouse has been clicked and remains inside either the entity instance list part of the window or the search string editable box. Three keystrokes cause the search command to be executed from the currently selected instance in the entity instance list:

- return - causes the search to go in the direction of the currently selected search radio button.
- ^s - causes a forward <u>s</u>earch regardless of the search button selected.
- ^r - causes a backward (<u>r</u>everse) search regardless of the search button selected.

8.2.2 Command Markings and Keystrokes for Dealing with Instances

The second set of commands place lower case letters called command marks preceding the selected instance in the entity instance list. Each entity instance that has a lower case letter preceding it will have a command that corresponds to the letter executed against it when the *execute* keystroke (*x* or *X*) is typed or the *execute (e)* button is pressed. Commands executed in this manner are useful when you want to do the same command to several instances. Keystroke commands allow you to work faster on a group of instances than dealing with each instance separately using the mouse. The keystroke commands that leave command marks are explained below. The keystroke command and the command mark left behind are identical.

- x : e<u>x</u>ecute - all instances that are preceded by command marks will have the command associated with the command mark executed against it. After the command associated with the command mark is executed, the command mark is cleared. Thus the result of this command is that all commands represented by command marks in the entity instance list are executed and all command marks are cleared from the entity instance list. This command does not leave a mark.

- u : unmark instance -- remove all keystroke command marks from the selected instance. This will not leave a mark.
- r : replicate instance using a shallow copy. The newly created replicate will have a STEP Entity Editor window placed on the screen for it.

Each of these keystroke commands will replace the command mark left by other keystroke commands in this list:

- s : save instance to a complete state.
- i : save instance to an incomplete state.
- d : mark instance for deletion.

Each of these keystroke commands will replace the command mark left by other keystroke commands in this list:

- m - modify instance by bringing up a Step Entity Editor window in a modifiable state.
- v - view instance by bringing up a Step Entity Editor window in a view-only state.
- c - close Step Entity Editor window.

8.3 Command Buttons

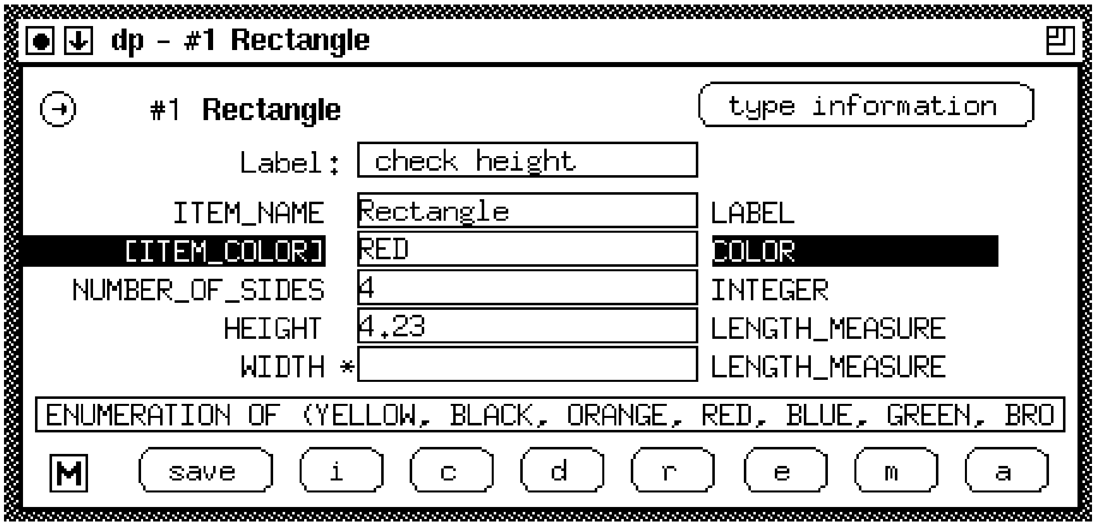
At the bottom of this window are buttons that provide an alternate way of executing commands to the keystroke commands that leave command marks. To execute commands using command buttons the user simply selects an entity instance from the entity instance list using the mouse (by pointing the mouse cursor on the instance and left clicking) and again using the mouse presses the appropriate command button. Pressing the button will cause the command to be executed immediately against the selected instance. Each button has the corresponding keystroke command for it in parenthesis in the button text. The following buttons are available:

- Execute (x) - execute the commands associated with the command marks against the instances that they precede in the entity instance list.
- Save Complete (s) - save instance to a complete state.
- Save Incomplete (i) - save instance to an incomplete state.
- Delete (d) - change the edit state for the instance to delete.
- Modify (m) - modify instance by bringing up a Step Entity Editor window.
- View (v) - view instance by bringing up a Step Entity Editor window in a view only state.
- Close (c) - close Step Entity Editor window.
- Replicate (r) - replicate instance using a shallow copy. Attribute values that reference another instance will have the reference to the instance copied. The instance being referenced will not be replicated. This will bring up a STEP Entity Editor window for the newly created replicate.

Double clicking the left mouse button over an entry for an instance in the entity instance list is like selecting that instance and pressing the *Modify (m)* button. It immediately places a modifiable SEE window on the screen for the instance.

9 STEP Entity Editor Windows

All editing of instance attribute values are done through STEP Entity Editor (SEE) windows. Figure 5 shows a SEE window containing an instance of a rectangle entity being edited within a Data Probe created from the information model found in Appendix B.



dp - #1 Rectangle

#1 Rectangle

Label: check height

```
         ITEM_NAME   Rectangle   LABEL
      [ITEM_COLOR]   RED         COLOR
   NUMBER_OF_SIDES   4           INTEGER
            HEIGHT   4.23        LENGTH_MEASURE
           WIDTH *               LENGTH_MEASURE
ENUMERATION OF (YELLOW, BLACK, ORANGE, RED, BLUE, GREEN, BRO
```

FIGURE 5 STEP Entity Editor (SEE) Window

A SEE window is created for an instance when it is newly created from the entity type list or when it is to be edited for the first time after having been read from a file. Whether or not the SEE window is on the screen, once created it will continue to exist until the instance it was created for is deleted. A SEE window may be removed from the screen or placed back on the screen as much as is desired during an editing session. Any changes that are made to the instance's attribute values inside the SEE window will not be reflected in the entity instance list until those changes have been saved. Changes to attribute values inside a SEE window that have not been saved will be left intact inside the SEE window even though the window is removed from the screen. The next time the SEE window is placed on the screen the changes made to the attribute values will still be there and may be saved.

9.1 Window Description

The SEE window contains a numeric instance identifier for the instance followed by the name of the entity type of the instance in the upper left of the window and also in the title bar provided by your X Window System window manager if your window manager provides a title bar. The middle of the window contains the list of attributes for the entity instance. Each attribute line contains the following in order:

- attribute name - optional attributes have square brackets around the attribute name.
- an optional asterisk - an asterisk indicates that the attribute value is missing and required or the existing value was validated by the Data Probe and was considered invalid (see section 4.3).
- attribute value - the value of the attribute is contained in an editable box that if possible helps with keeping the value valid. It does this in two ways. First, if possible it will not allow invalid values be typed in -- such as not allowing letters to be typed into an integer field. And second, by validating the attribute value when the user presses a terminating character -- such as hitting return to move to the next attribute value field or saving the instance.
- attribute type.

Below the list of attributes there is a message area where error messages are displayed or the results of commands are communicated. This message area is scrollable using either the middle mouse button for grab scrolling or the right mouse button for rate scrolling (see section 5.5).

The SEE window contains a button on the top left and right side of the window and a row of buttons along the bottom of the window. The actions of these buttons are explained below.

9.2 SEE Buttons and Corresponding Keystroke Commands

A SEE window contains several buttons. This section describes the functionality of each of the buttons and where applicable specifies the corresponding keystroke command.

- Pin Button - In the upper left hand corner of the SEE window is a toggle button that contains a pin. The pin toggles between *stuck in* and *pulled out*. If the pin is *pulled out*, the SEE window will be closed when the instance is saved, canceled, or changed to a delete state. If the pin is *stuck in*, the SEE window will stay on the screen when saved, canceled, or changed to a delete state. This button defaults to *pulled out*.
- Type Information - ^x ^t or ^x t - In the upper right hand corner of the SEE window is a text button labelled *type information*. This button pops up a STEP Entity Descriptor window that contains information about the entity type. The STEP Entity Descriptor window includes attribute information including where in the entity inheritance hierarchy (if any) the attribute came from, the schema that each entity in the entity inheritance hierarchy came from, and the subtypes of each entity in the entity inheritance hierarchy. The keystroke commands *^x ^t* or *^x t* may also be used to place the STEP Entity Descriptor window on the screen.
- View/Modify - The bottom left corner of the SEE window contains a toggle button containing an *M* or a *V.* This button toggles the SEE window between a modifiable or view-only state as depicted by the button containing either the letter *M* or *V.*

Along the bottom of the SEE window is a row of text buttons The following shows the button action, followed by the button text, the keystroke command sequence that will execute the command from an attribute editable field, and finally an explanation of the command. The buttons are listed as they appear in the SEE window from left to right.

- Save Complete - save - ^x ^s or ^x s - save the instance to a complete state. If any of the attribute values are invalid the instance will be saved to an incomplete state. The invalid attributes will be marked with an asterisk preceding their attribute editable field. This button will cause the SEE window to be closed unless the pin button is *stuck in*.

- Save Incomplete - i - ^x ^i or ^x i - save the instance to an incomplete state. Any invalid attributes will be marked with an asterisk preceding the attribute editable field. This button will cause the SEE window to be closed unless the pin button is *stuck in*.
- Cancel - c - ^x ^c or ^x c - revert the values of the attribute fields back to the last saved values. This button will cause the SEE window to be closed unless the pin button is *stuck in*.
- Delete - d - ^x ^d or ^x d - change the edit state of this instance to a deletion state. Instances with an editing state of delete will be deleted when the “Remove Deleted Instances” menu option is chosen from the “File Management” menu or when being read from a Working Session file in which they were written. This button will cause the SEE window to be closed unless the pin button is *stuck in*.
- Replicate - r - ^x ^r or ^x r - replicate the entity instance using a shallow copy. Attribute values that reference another instance will have the reference to the instance copied. The instance being referenced will not be replicated. This button will cause a SEE window to be placed on the screen for the newly created replicate instance.
- Edit - e - ^x ^e or ^x e - edit the value of the attribute using additional help. Pressing this button when editing an attribute that is of type enumeration will cause a dialog box called an Enumeration Chooser to be placed on the screen. The Enumeration Chooser contains the valid enumeration choices for the attribute value. Figure 6 shows an Enumeration Chooser for the *item_color* attribute in Figure 5. This Enumeration Chooser contains the enumeration values from the color enumeration defined in Express in the information model found in Appendix B. An enumerated

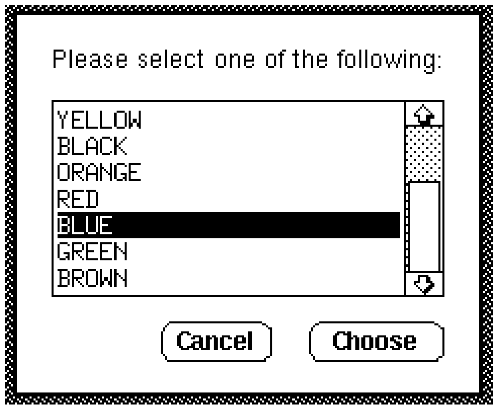


FIGURE 6 STEP Entity Editor (SEE) Enumeration Chooser

value chosen from the Enumeration Chooser will be filled in as the new value of the attribute. Pressing this button when editing an attribute of type entity will cause one of two actions. If the attribute value already contains a reference to an entity instance then the referenced entity instance will have a modifiable SEE window placed on the screen for it. This is a convenient way to bring up a SEE window for editing or viewing an entity instance referenced by another entity instance. If the attribute does not contain an entity instance reference for its value then a new instance of the shown entity type will be created, a SEE window will be placed on the screen for the newly created instance, and the reference to the newly created instance will be filled in as the new attribute value.

- Mark - m - ^x ^m or ^x m - fill in as the value of this attribute a reference to the instance that is selected in the Entity Instance List window. This button only works with attributes of type entity or aggregate of entity. Attributes of type entity will have their value replaced by the selected instance. Attributes of type aggregate of entity will have the value appended to the end of the aggregate list of entity references.
- Attribute Type Information - a - ^x ^a or ^x a - a detailed description of the attribute type is written to the message area of the SEE window.

9.3 Additional Keystroke Commands

Selecting an attribute for editing:

- left mouse button - point the mouse cursor inside one of the attribute editing boxes and click the left mouse button. Leaving an attribute editing box in this manner will bypass the validation of that attribute value. The attribute value will still be validated when the instance is saved.
- <return>, or ^n - select next attribute. This causes the attribute value to be validated.
- ^p - select previous attribute. This also causes the attribute value to be validated.

Editing an attribute in an attribute editing box:

- ^x ^u, ^x u - attribute undo: Revert attribute value back to the last attribute value saved (complete or incomplete).
- All of the commands found in section 5.5 may be used to edit an attribute value.

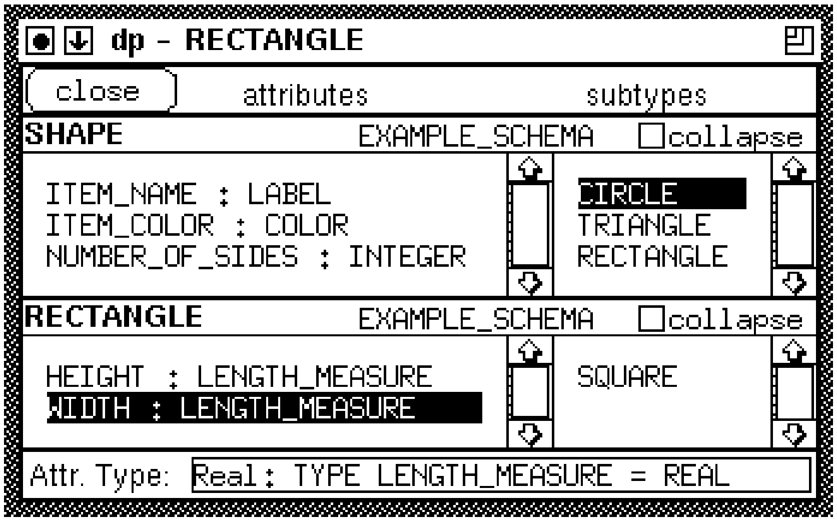
9.4 Opening SEE Windows

A SEE window is opened when a new entity instance is created to allow the user to fill in values for the attributes of the newly created instance. A SEE window may also be opened for editing any existing entity instance. The following summarizes the opening of SEE windows:

- a SEE window is opened when a new entity instance is created from the Entity Type List window (see section 7.1.2 and 7.2)
- a SEE window is opened when a new entity instance is created using the *edit* keystroke command or button inside a SEE window that has an attribute that references another entity instance (see section 9.2).
- a SEE window is opened when the attribute being edited inside a SEE window contains a reference to an entity instance and the *edit* keystroke command is typed or the *edit* button is pressed. (see section 9.2).
- a SEE window may be opened for any existing instance from the Entity Instance List window (see section 8.2.2 and 8.3).

10 STEP Entity Descriptor Windows

The STEP Entity Descriptor (SED) windows display detailed information about the entity types. Figure 7 shows a SED window for a rectangle entity type from a Data Probe created from the information model found in Appendix B.


dp - RECTANGLE

close attributes subtypes

SHAPE EXAMPLE_SCHEMA collapse

ITEM_NAME : LABEL
ITEM_COLOR : COLOR
NUMBER_OF_SIDES : INTEGER

CIRCLE
TRIANGLE
RECTANGLE

RECTANGLE EXAMPLE_SCHEMA collapse

HEIGHT : LENGTH_MEASURE
WIDTH : LENGTH_MEASURE

SQUARE

Attr. Type: Real: TYPE LENGTH_MEASURE = REAL

FIGURE 7 STEP Entity Descriptor (SED) Window

10.1 Window Description

The middle of the SED window is made up of entity description boxes. There is an entity description box for each entity in the entity inheritance hierarchy for the selected entity type. The bottom entity description box displays information local to the selected entity type. Each entity description box is preceded by an entity description box for the entity’s supertype entity type on up to the top entity in the hierarchy for the selected entity type. In other words there is a list of entity description boxes in the order of inheritance starting at the root entity type down to the selected entity type.

Each entity description box contains a title bar and a body. The title bar contains the entity type name in bold letters at the top left, followed to the right by the name of the schema in which the entity type was defined. On the far right of the title bar is a button which will allow the body of the entity description box to be hidden. The body of the entity description box contains two scrollable lists. The list on the left contains the attribute definitions in Express for all of the attributes declared within the entity type. When a user double clicks on one of the attribute definitions a more detailed explanation of the attribute type is displayed in a message area at the bottom of the SED window. The list on the right of the entity description box contains the subtypes of the entity type. When a user double clicks on one of the entity types in the list of subtypes a SED window for that subtype entity type will be opened on the screen. The top of the SED window contains a ‘close’ button for closing the SED window.

10.2 Opening SED Windows

A SED window may be opened for an entity type from the following places:

- The Entity Type List window by selecting the appropriate entity type with the left mouse button and pressing the *Type Information (t)* button or pressing *t* the type information keystroke command.
- A SED window by double clicking with the mouse on an entity type in one of the subtype lists found on a SED window.
- A SEE window displaying an instance of a particular entity type. The SED window opened will be for the entity type matching the type of the instance in the SEE window. The SED window is opened by pressing the *type information* button on the SEE window or by typing *^x t* or *^x ^t* (the type information keystroke command for a SEE window).

11 Data Probe Editing Session Tutorial

This tutorial walks a user through the basics of using a Data Probe. The Data Probe being used in this tutorial is one that was created using the information model found in Appendix B. When this tutorial indicates that a window will appear on the screen the window manager may place the window itself or it may require assistance from the user.

11.1 Run the Data Probe Example

The first thing to do is run the Data Probe. Simply type the name of the Data Probe executable at the command prompt of your workstation. Three windows should have appeared (see Figure 8). The Data Probe window should have appeared at the top middle of your screen. The Entity Instance List and Entity Type List windows should have appeared at the left and right sides of the screen respectively.

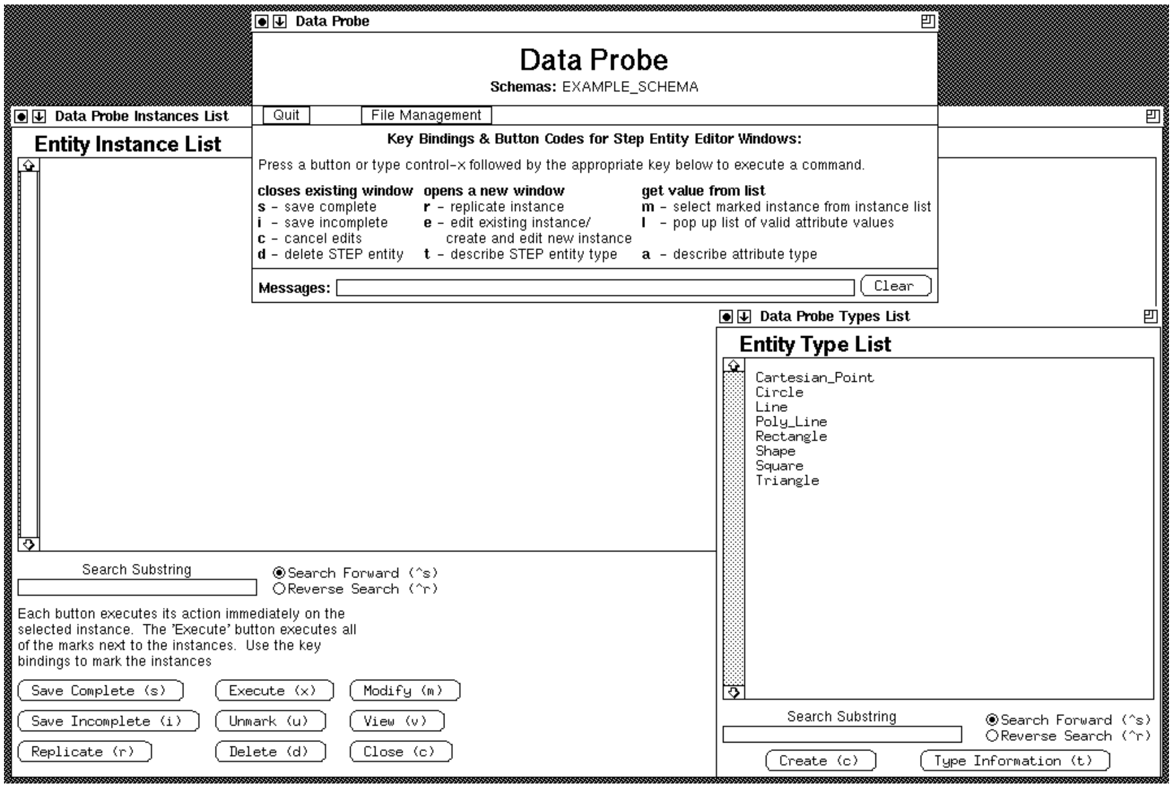
11.2 Exit a Data Probe Editing Session and How to Use a Menu

A menu bar is a box containing text that allows access to a menu. When the mouse cursor is placed over a menu bar it redraws itself as a menu. When the mouse cursor is moved outside of the menu bar it redraws itself as the default mouse cursor.

Follow these steps to quit the Data Probe editing session:

- Place the mouse cursor inside the quit menu bar. The quit menu bar is a box labelled quit located on the left of the Data Probe window. Notice how the cursor has drawn itself to look like a menu.
- With the mouse cursor still inside the menu bar press and hold the left mouse button down. The quit menu will now pull down and draw itself. As long as you hold the left mouse button down the menu will remain open no matter where you move the mouse cursor (see Figure 9).
- While still holding the left mouse button down move the mouse cursor down into the pull down menu so that the first menu entry labelled *quit* highlights itself. If the cursor is moved outside the menu while still holding the left mouse button down the quit menu entry will unhighlight itself and the menu will remain open. To avoid choosing a command from the menu simply release the mouse button outside of the menu. To choose a menu entry place the mouse cursor over the menu entry so that it highlights itself and release the mouse button. Choose the quit menu entry to quit the Data Probe editing session.

Run the Data Probe again to continue the tutorial.



Key Bindings & Button Codes for Step Entity Editor Windows:

Press a button or type control-x followed by the appropriate key below to execute a command.

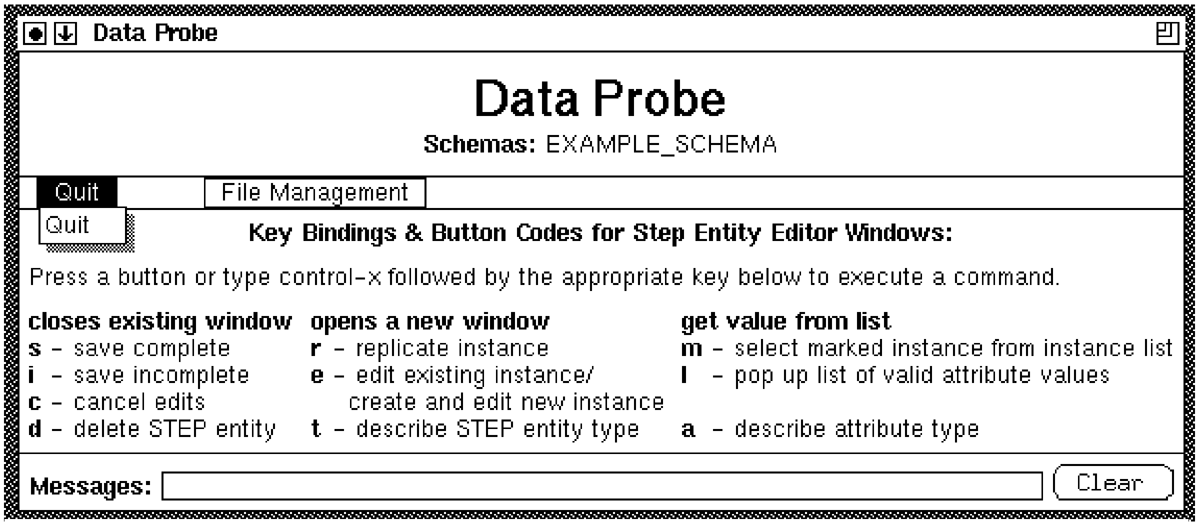
closes existing window	opens a new window	get value from list
s – save complete	**r** – replicate instance	**m** – select marked instance from instance list
i – save incomplete	**e** – edit existing instance/ create and edit new instance	**l** – pop up list of valid attribute values
c – cancel edits		
d – delete STEP entity	**t** – describe STEP entity type	**a** – describe attribute type

Messages:

```
Cartesian_Point
Circle
Line
Poly_Line
Rectangle
Shape
Square
Triangle
```

Each button executes its action immediately on the selected instance. The 'Execute' button executes all of the marks next to the instances. Use the key bindings to mark the instances

FIGURE 8 The Three Data Probe Windows at Start Up



Key Bindings & Button Codes for Step Entity Editor Windows:

Press a button or type control-x followed by the appropriate key below to execute a command.

closes existing window	opens a new window	get value from list
s – save complete	**r** – replicate instance	**m** – select marked instance from instance list
i – save incomplete	**e** – edit existing instance/ create and edit new instance	**l** – pop up list of valid attribute values
c – cancel edits		
d – delete STEP entity	**t** – describe STEP entity type	**a** – describe attribute type

Messages:

FIGURE 9 Quitting the Data Probe

11.3 Create an Entity Instance

The next two steps will be referred to as selecting an entity type from the entity type list.

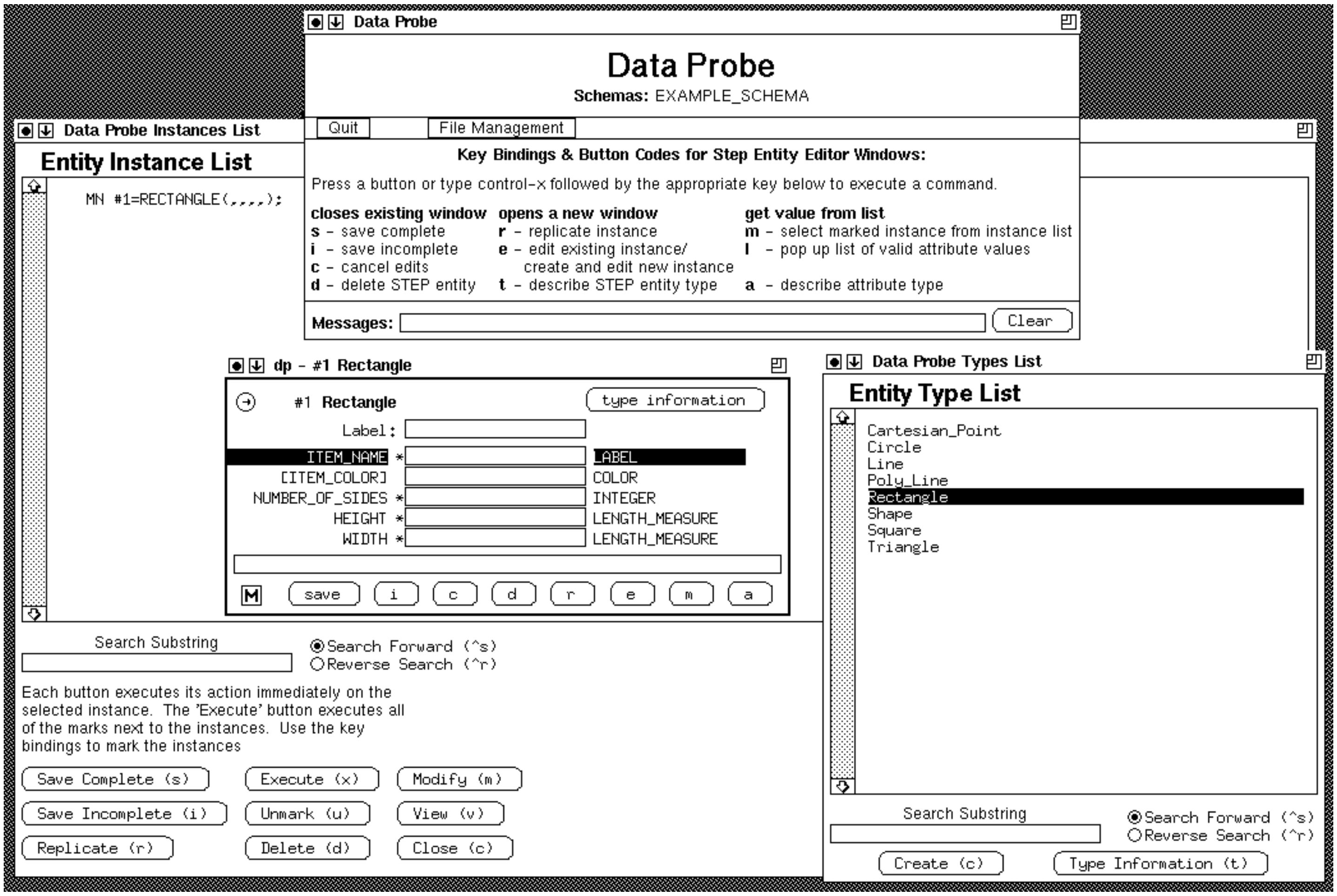
- Place the mouse cursor over the entry that says *Rectangle* in the list of entity types in the Entity Type List window.
- Click the left mouse button and release. The line should now be highlighted.

The next two steps cover pressing a button.

- Now move the mouse cursor over the button in the Entity Type List window that says *Create (c)*. Notice that the mouse cursor has changed to a hand. This indicates that it is over a button. Click the left mouse button and hold. The button has highlighted itself showing that it has been pressed. Notice that when you move the mouse cursor outside of the button area the button unhighlights itself. If you release the left mouse button while the mouse cursor is outside of the button area the button will not be pressed. Go ahead and release the left mouse button now inside of the button area to create an instance of a rectangle entity.
- A STEP Entity Editor (SEE) window will appear on the screen for the newly created instance of type *Rectangle* and an entry for the instance will appear in the entity instance list in the Entity Instance List window (see Figure 10). Notice that the entry in the entity instance list is preceded by *N* for *new*. This is the editing state of the instance. Since the instance is newly created there are no values associated with the entity instance in the SEE window or in the entity instance list. Later when the instance has been edited the attribute values for the instance will show up in both places. This instance has an instance identifier number associated with it. It is located to the left of the entity name in both the SEE window and its entry in the entity instance list. The instance has an instance identifier *1* since it is the first created instance. Notice that the instance's entry in the entity instance list is also preceded by *M*. The *M* tells you that the instance has a SEE window representing the instance that is in a modifiable state on the screen. If it had a view-only state the instance in the entity instance list would be preceded by *V*.

11.4 Create Two More Entity Instances Using Alternate Methods

- Select entity type *Line* from the entity type list in the Entity Type List window.
- Leave the mouse cursor inside the entity type list and type the letter *c* on your keyboard. This is the keystroke command method for creating a new entity instance from the entity type list.
- A SEE window will appear on the screen for the newly created instance of type *Line* and an entry for the instance will be added to the entity instance list in the Entity Instance List window. The instance will have an instance identifier *2* since it is the second instance created.
- Select entity type *Cartesian_Point* from the Entity Type List window.
- Click the left mouse button twice quickly over entity type *Cartesian_Point*. This is referred to as double clicking.
- If you double clicked on the list entry fast enough a SEE window will appear on the screen for a newly created instance of type *Cartesian_Point* and an entry for the instance will appear in the list of instances in the entity instance list. If the instance did not get created then you either need to allow less time between the two clicks of the mouse button or you moved the mouse cursor while you were clicking so that the



Data Probe

Schemas: EXAMPLE_SCHEMA

Key Bindings & Button Codes for Step Entity Editor Windows:

Press a button or type control-x followed by the appropriate key below to execute a command.

closes existing window	opens a new window	get value from list
s – save complete	**r** – replicate instance	**m** – select marked instance from instance list
i – save incomplete	**e** – edit existing instance/ create and edit new instance	**l** – pop up list of valid attribute values
c – cancel edits		
d – delete STEP entity	**t** – describe STEP entity type	**a** – describe attribute type

Messages:

```
MN #1=RECTANGLE(,,,,);
```

#1 **Rectangle**

Label:		
ITEM_NAME *		LABEL
[ITEM_COLOR]		COLOR
NUMBER_OF_SIDES *		INTEGER
HEIGHT *		LENGTH_MEASURE
WIDTH *		LENGTH_MEASURE

Cartesian_Point
Circle
Line
Poly_Line
Rectangle
Shape
Square
Triangle

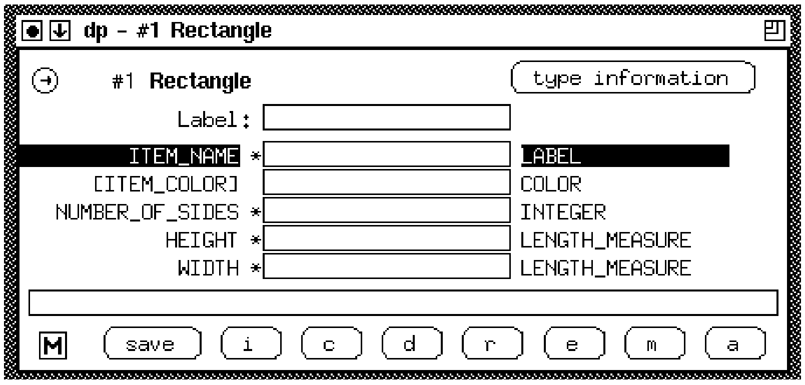
Each button executes its action immediately on the selected instance. The 'Execute' button executes all of the marks next to the instances. Use the key bindings to mark the instances

FIGURE 10 Creating an Entity Instance of Type Rectangle

selected entity type did not receive both mouse clicks. The new instance will have an instance identifier *3*.

11.5 Editing an Entity Instance in a SEE Window

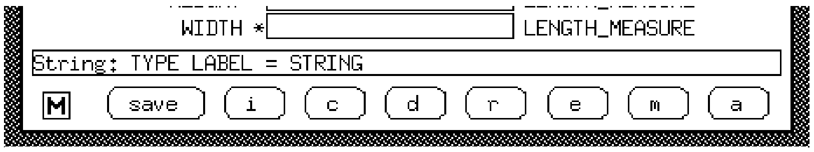
- Use the window manager on your computer to raise the SEE window for the rectangle instance that you created earlier so that it is in full view (see Figure 11).
- The top of the SEE window contains the instance identifier followed by the name of the type of the instance in bold font. The attributes are listed from top to bottom in the middle of the SEE window. The names of the attributes are on the left of the SEE window. The attribute name is followed by an editable box for the attribute value. The editable box is followed by the attribute's type. The attributes are preceded by an editable area called a label where you can make notes that will be remembered within the SEE window for the instance. The attributes are followed by a message area where results of commands are communicated from the Data Probe. Various buttons appear around the perimeter of the SEE window. The buttons on the bottom of the window are explained in the Data Probe window (see Figure 9).



```
ITEM_NAME *       LABEL
[ITEM_COLOR]      COLOR
NUMBER_OF_SIDES * INTEGER
HEIGHT *          LENGTH_MEASURE
WIDTH *           LENGTH_MEASURE
```

FIGURE 11 SEE Window for an Instance of Type Rectangle

- Notice there is an asterisk before the attribute editable boxes for all but the second attribute. An asterisk before an attribute value signifies that the attribute value is invalid. Right now the attributes that are invalid are invalid because they are required to have values but do not. The second attribute is valid without a value since it is an optional attribute as shown by the brackets around the attribute name.
- Select the first attribute in the SEE window to be edited. This is done by moving the mouse cursor over the editable box between the attribute name and the attribute type. In this case the attribute name is ITEM_NAME and the attribute type is LABEL. When you do this the attribute name and type should become highlighted. You may use this method to select any attribute in any SEE window for editing. Pointing and clicking the mouse cursor in the middle of an attribute value inside of an editable box for an attribute will place the cursor in the attribute editable box at the point in the attribute value where the mouse was clicked.
- Use the mouse to press the button at the bottom right of the SEE window. This will cause a message that gives greater detail about the attribute type to be displayed in the message area below the last attribute (see Figure 12). It will indicate that the attribute type LABEL has a base type of type string. String types will accept any characters that you type from the keyboard as input. Click the mouse back inside of the attribute editable box and input a value.



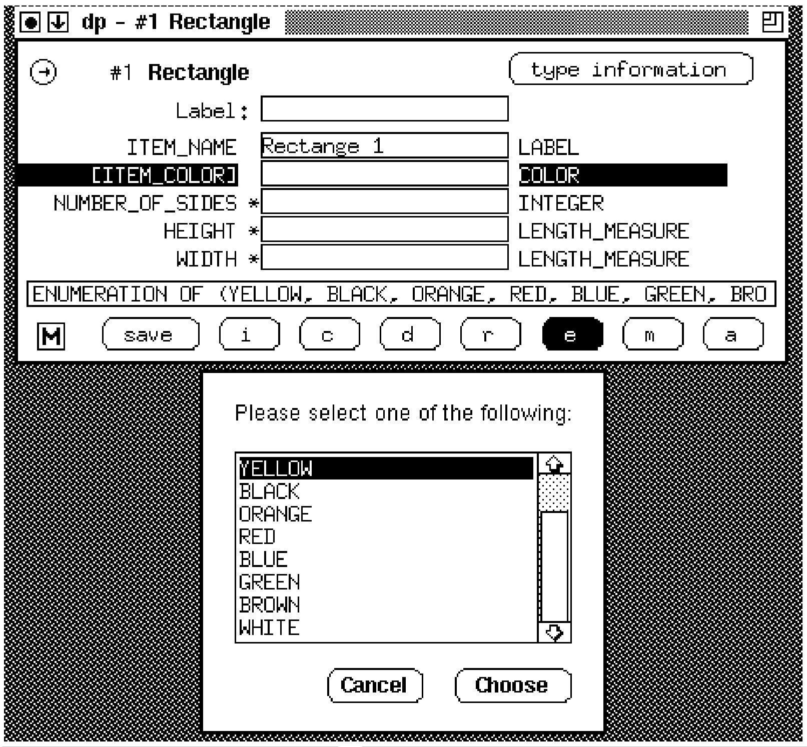
```
WIDTH *  LENGTH_MEASURE
String: TYPE LABEL = STRING
```

FIGURE 12 The SEE Window Message Area

- Press return after you have entered a value and the next attribute ITEM_COLOR will become highlighted meaning that it is now ready to be edited. Notice that the asterisk before the first attribute value has gone away. When you hit return the Data Probe performs validation on the attribute value.
- For this attribute you will use the keystroke command to determine the type of the attribute. Press and hold *control* and while still holding press *x*. Release both keys and type *a*. This key sequence is represented by the following characters ^*x a*. The SEE message area will inform you that color is an enumerated type and it will list the

enumeration values for you (see Figure 13). Point the mouse cursor inside the message area and press the middle mouse button to practice grab scrolling. Point the mouse cursor inside the message area and press the right mouse button to practice rate scrolling. These types of scrolling are implemented in the same way inside of lists except that the scrolling is vertical instead of horizontal.

■ The Data Probe will allow you to type in one of the enumerated values listed in the SEE window message area however it also allows you a more convenient way of filling in the attribute value. Use the mouse to press the button at the bottom of the SEE window labelled *e* for edit. A SEE enumeration chooser will appear on the screen listing all of the valid choices for the enumeration (see Figure 13). The keystroke command for this is ^*x e*.



Label:		
ITEM_NAME	Rectange 1	LABEL
[ITEM_COLOR]		COLOR
NUMBER_OF_SIDES *		INTEGER
HEIGHT *		LENGTH_MEASURE
WIDTH *		LENGTH_MEASURE

ENUMERATION OF (YELLOW, BLACK, ORANGE, RED, BLUE, GREEN, BRO

Please select one of the following:

YELLOW
BLACK
ORANGE
RED
BLUE
GREEN
BROWN
WHITE

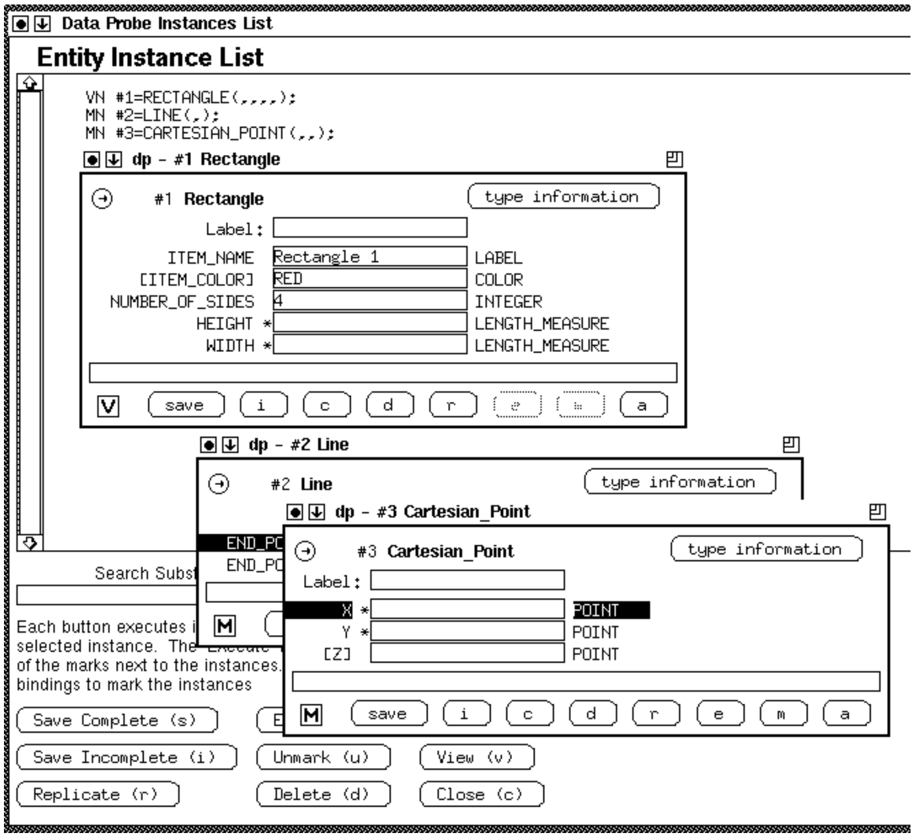
FIGURE 13 SEE and Associated Enumeration Chooser

■ Using the mouse select one of the enumerated values from the list in the SEE Enumeration Chooser. This is done just as you selected an entity type from the entity type list in the Entity Type List window.

■ Press the button at the bottom of the SEE Enumeration Chooser labelled *Choose* and the value will be filled in for you in the attribute editable box for ITEM_COLOR. You may also fill in the attribute value from a selection in the SEE Enumeration Chooser by double clicking one of the entries in the SEE Enumeration Chooser.

■ Hit ^*x n* to go to the next attribute. Type a few alphabetic characters. Your computer will beep whenever you type anything other than an integer. The Data Probe knows that the number_of_sides attribute requires an integer value so it will not let you enter non-integer characters. Enter an integer value and hit return to go to the last attribute.

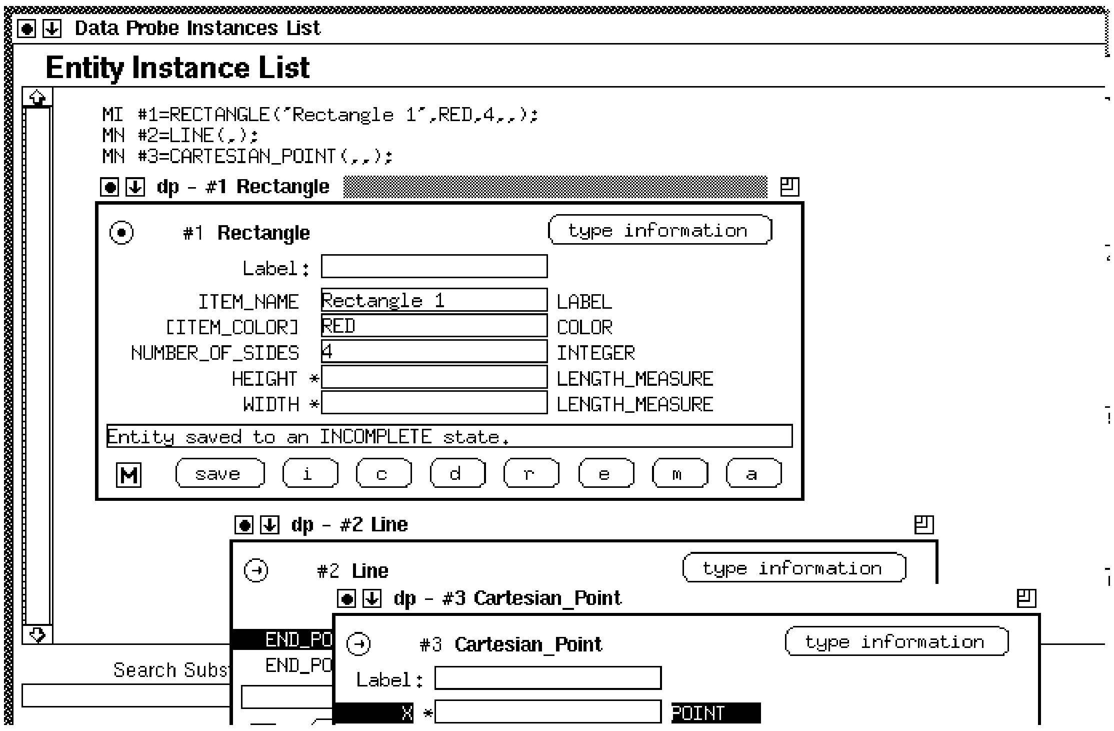
- Hit return again and the first attribute will become highlighted. Hit *^x n* or *return* a couple of times. These are keystroke commands for selecting the next attribute value. Hit *^x p* a couple of times. This is the keystroke command for selecting the previous attribute value.
- Press the button on the bottom left of the SEE window marked *M*. Notice that the button now says *V*, the attribute values may no longer be edited, and you may not press the edit, or mark buttons. This button toggles the window between modify and view-only mode. Also the entry for the instance in the entity instance list is now preceded by *V* instead of *M* (see Figure 14). Press the button again to toggle back to modify mode.



```
Data Probe Instances List
Entity Instance List
    VN #1=RECTANGLE(,,,,);
    MN #2=LINE(,);
    MN #3=CARTESIAN_POINT(,,);

dp - #1 Rectangle
    #1 Rectangle                          type information
              Label:
          ITEM_NAME   Rectangle 1         LABEL
       [ITEM_COLOR]   RED                 COLOR
    NUMBER_OF_SIDES   4                   INTEGER
           HEIGHT *                       LENGTH_MEASURE
            WIDTH *                       LENGTH_MEASURE
    V  save  i  c  d  r  e  m  a

dp - #2 Line
    #2 Line                               type information
    END_PC
    END_PC
    M

dp - #3 Cartesian_Point
    #3 Cartesian_Point                    type information
    Label:
         X *                    POINT
         Y *                    POINT
       [Z]                      POINT
    M  save  i  c  d  r  e  m  a

Search Subst
Each button executes i
selected instance.  The
of the marks next to the instances.
bindings to mark the instances

Save Complete (s)      E
Save Incomplete (i)    Unmark (u)    View (v)
Replicate (r)          Delete (d)    Close (c)
```

FIGURE 14 View-Only SEE Window (Rectangle)

- Press the button that looks like a pin pulled out in the top left corner of the window opposite the type information button. Notice that the pin in the button now looks like it is stuck in (see Figure 15). This button has just ‘pinned’ the window to the screen so that when you save the entity instance in the next step the window will not remove itself from the screen.
- Notice in Figure 14 that the values for the instance that are in the SEE window for the rectangle instance do not match the values that are in the entity instance list. This is because you have not saved the attribute values that you have edited for the instance in the SEE window. Press the button marked *save* at the bottom left of the SEE window to save the values you just entered for the rectangle instance. Saving the instance should have caused a beep and there should be a message in the SEE



```
MI #1=RECTANGLE('Rectangle 1',RED,4,,);
MN #2=LINE(,);
MN #3=CARTESIAN_POINT(,,);
```

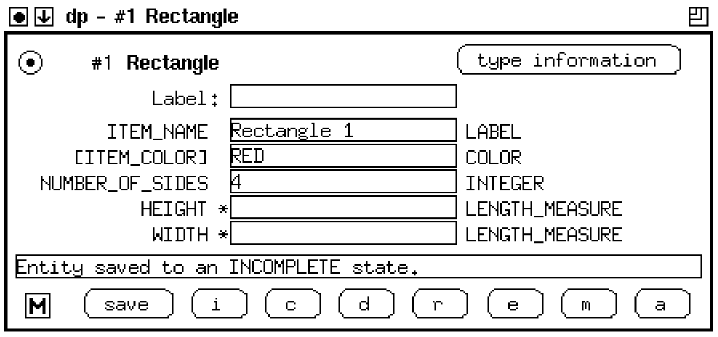
#1 Rectangle

Label:		
ITEM_NAME	Rectangle 1	LABEL
[ITEM_COLOR]	RED	COLOR
NUMBER_OF_SIDES	4	INTEGER
HEIGHT *		LENGTH_MEASURE
WIDTH *		LENGTH_MEASURE

Entity saved to an INCOMPLETE state.

FIGURE 15 Rectangle Instance Attribute Values Saved from SEE Window

message area saying that the instance was saved to an incomplete state (see Figure 15). Even though you told the Data Probe to save the instance to a complete state it was only saved to an incomplete state. This is because you didn't fill in a value for the last two attributes which are required. Thus the instance as a whole was not valid and could only be save to an incomplete state. Look at the entity instance list entry for this instance. Notice that the values you just filled in are now shown in the instance in the entity instance list corresponding to the one you were working on in the SEE window (see Figure 16). Notice also that the instance is preceded by *I* instead of *N*. This shows that the instance is no longer new but has been saved to an incomplete state.



#1 Rectangle

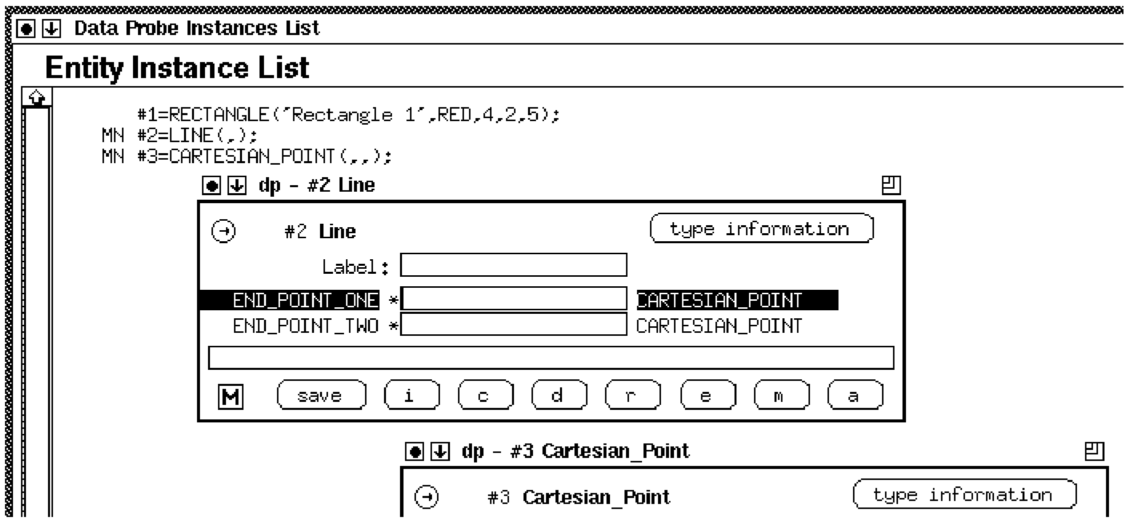
Label:		
ITEM_NAME	Rectangle 1	LABEL
[ITEM_COLOR]	RED	COLOR
NUMBER_OF_SIDES	4	INTEGER
HEIGHT *		LENGTH_MEASURE
WIDTH *		LENGTH_MEASURE

Entity saved to an INCOMPLETE state.

FIGURE 16 Pinned SEE Window

- Enter a value now for the last two attributes. Press the pin button again to unpin the window from the screen.

- Press the save button again. Since this time the window was unpinned the window should have gone away. This time there should not have been a beep since the instance should have been able to be saved to a complete state without any problems. The *I* preceding the entry for the instance in the entity instance list should have gone away signifying that the instance is now in a complete state. The change you just made to the last attribute value should be reflected in the instance entry in the entity instance list. The *M* preceding the entry for the instance in the entity instance list should also have gone away since the SEE window is no longer on the screen (see Figure 17).



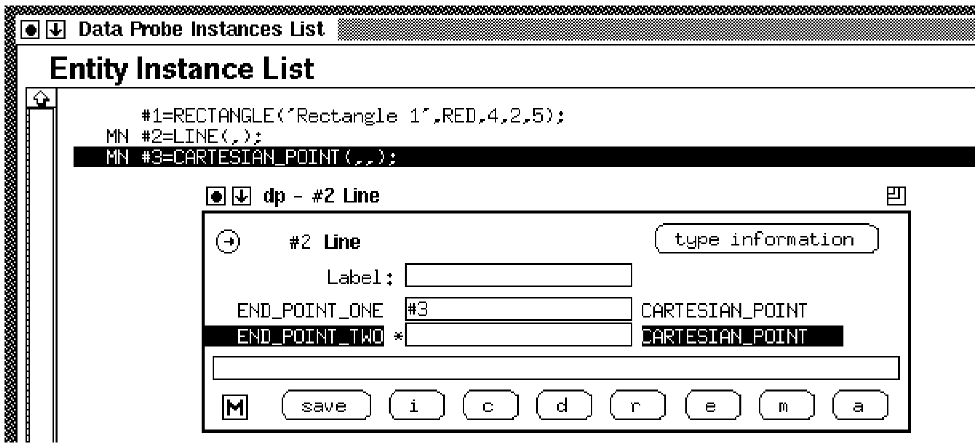
```
    #1=RECTANGLE('Rectangle 1',RED,4,2,5);
MN  #2=LINE(,);
MN  #3=CARTESIAN_POINT(,,);
```

```
Label:
END_POINT_ONE *              CARTESIAN_POINT
END_POINT_TWO *              CARTESIAN_POINT
```

FIGURE 17 Rectangle Entity Saved Complete and SEE Window Closed

11.6 More SEE Editing Functionality

- Use the window manager on your computer to raise the SEE windows for the other two instances that you created earlier so that they are in full view.
- Click with the mouse over the entry for the cartesian_point instance in the entity instance list. The entry for cartesian_point in the entity instance list should now be highlighted. Now click with the mouse inside the first attribute editable box inside the SEE window for instance with instance identifier *2* of type *line*. The first attribute should now be highlighted. Now press the button at the bottom of the SEE window labelled *m*. This will cause the instance identifier number of the selected instance in the entity instance list to be filled in as the attribute value of the selected attribute in the SEE window (see Figure 18). Hit return to go to the next attribute.
- Now press the button labelled *e* at the bottom of the SEE window. You will notice that a new instance of the appropriate type has been created and placed on the screen and that the instance identifier of this new instance has been filled in as the attribute value (see Figure 19).
- Press the pin button to pin this entity instance to the screen. Now press the save button. Enter a new value in one of the attributes. Now type *^x u*. The previous attribute value saved will replace the value you just typed in. This is the keystroke command for attribute undo. Now change the attribute values for both of the attributes. Press the button labelled *c*. This will cancel the changes you just made. Notice the previous attribute values have returned.



Data Probe Instances List

Entity Instance List

```
    #1=RECTANGLE('Rectangle 1',RED,4,2,5);
 MN #2=LINE(,);
 MN #3=CARTESIAN_POINT(,,);
```

dp - #2 Line

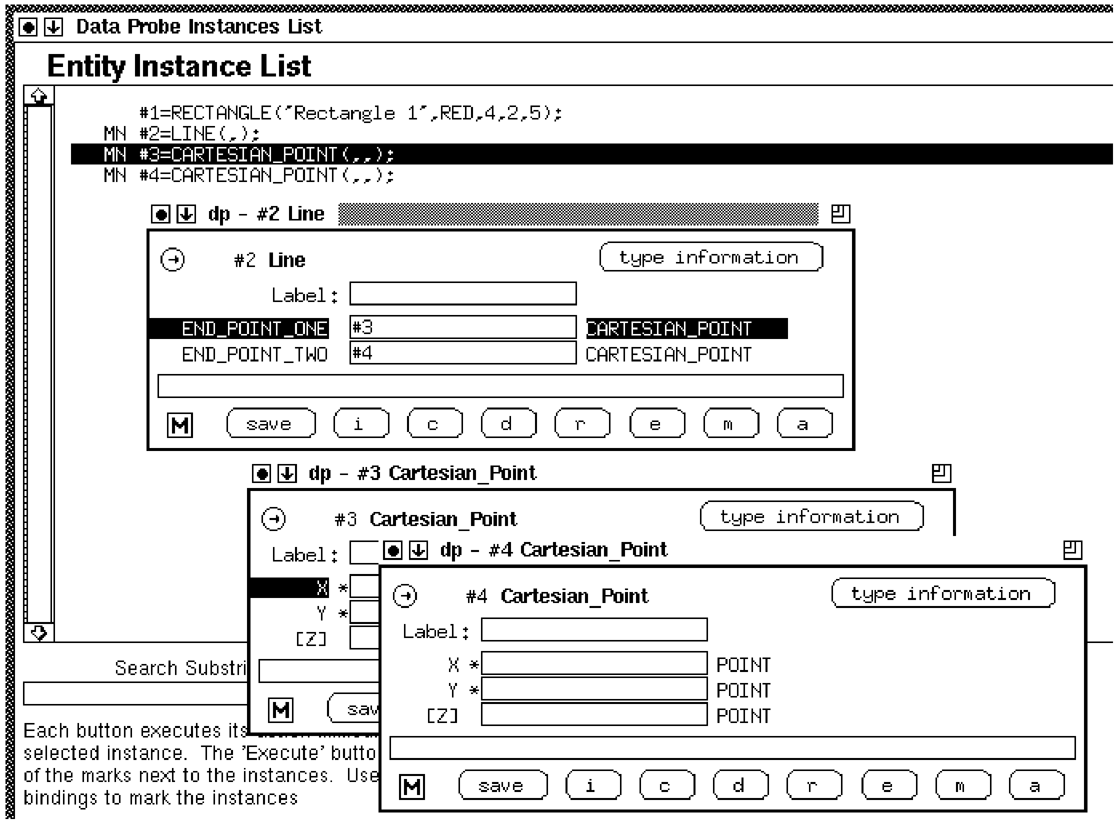
#2 Line

Label:

END_POINT_ONE #3 CARTESIAN_POINT

END_POINT_TWO * CARTESIAN_POINT

FIGURE 18 Filling in an Attribute Value Using the Mark Button in a SEE Window



Data Probe Instances List

Entity Instance List

```
    #1=RECTANGLE('Rectangle 1',RED,4,2,5);
 MN #2=LINE(,);
 MN #3=CARTESIAN_POINT(,,);
 MN #4=CARTESIAN_POINT(,,);
```

dp - #2 Line

#2 Line

Label:

END_POINT_ONE #3 CARTESIAN_POINT

END_POINT_TWO #4 CARTESIAN_POINT

dp - #3 Cartesian_Point

#3 Cartesian_Point

Label:

X *

Y *

[Z]

dp - #4 Cartesian_Point

#4 Cartesian_Point

Label:

X * POINT

Y * POINT

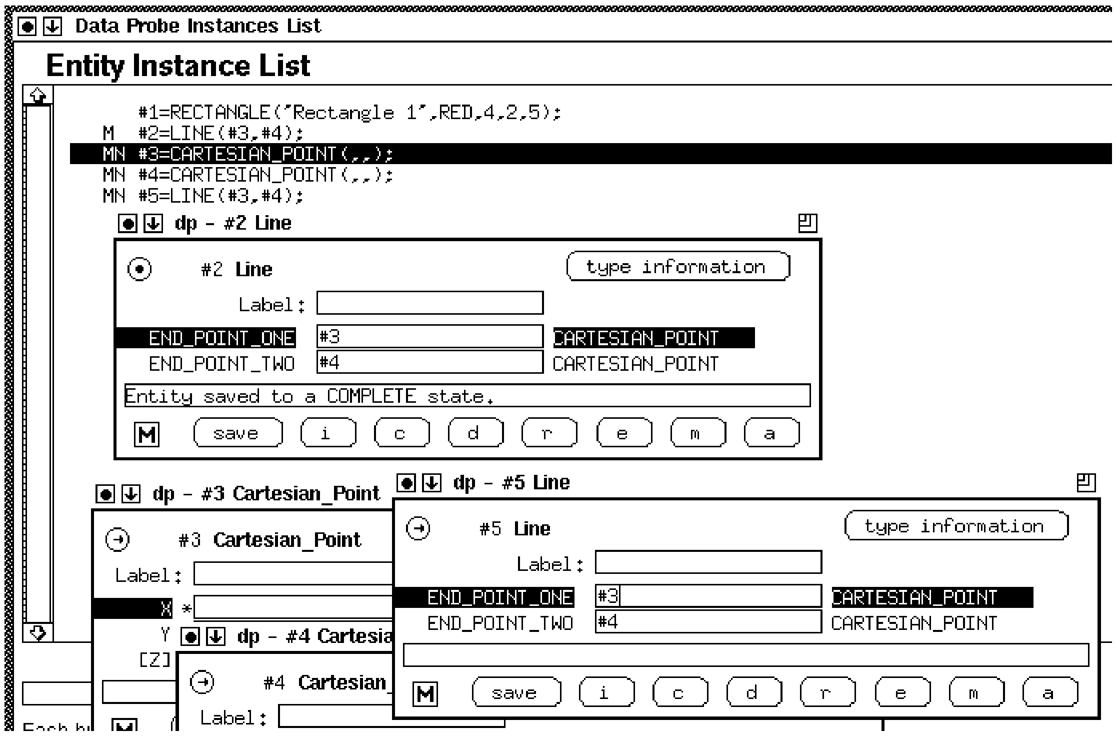
[Z] POINT

Search Substri

Each button executes its
selected instance. The 'Execute' butto
of the marks next to the instances. Use
bindings to mark the instances

FIGURE 19 Filling in an Attribute Value Using the Edit Button in a SEE Window

- Press the button labelled *r* at the bottom of the SEE window. A new entity instance has been created with attribute values identical to the current instance (see Figure 20). Whenever a new instance is created a SEE window is placed on the screen for that instance.



```
    #1=RECTANGLE("Rectangle 1",RED,4,2,5);
M   #2=LINE(#3,#4);
MN  #3=CARTESIAN_POINT(,,);
MN  #4=CARTESIAN_POINT(,,);
MN  #5=LINE(#3,#4);
```

```
#2 Line
Label:
END_POINT_ONE  #3  CARTESIAN_POINT
END_POINT_TWO  #4  CARTESIAN_POINT
Entity saved to a COMPLETE state.
```

```
#3 Cartesian_Point
Label:
X *
Y
[Z]
```

```
#4 Cartesian_
Label:
```

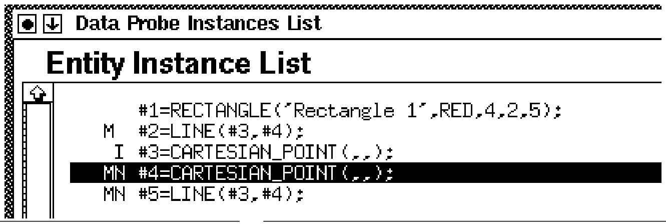
```
#5 Line
Label:
END_POINT_ONE  #3  CARTESIAN_POINT
END_POINT_TWO  #4  CARTESIAN_POINT
```

FIGURE 20 #5 Line Instance Replicated From #2 Line Instance

11.7 Working from the Entity Instance List Window

- You may use the Entity Instance List window to manipulate instances in the list of entity instances and you may open or close SEE windows.
- The Entity Instance List window will allow you to change the editing state of instances in the entity instance list.
- Select from the entity instance list the cartesian_point instance with instance identifier *3*. Notice that it has letters *M* and *N* preceding the instance in the list. The *M* means that there is a SEE window on the screen for the instance. The *N* means that the instance is newly created.
- Press the button at the bottom of the window labelled *Save Incomplete (i)*. This will cause the SEE window to be removed from the screen and the instance to be saved to an incomplete editing state. The *N* will have turned into *I* to indicate that the editing state has changed from new to incomplete. The *Save Complete (s)* and *Delete (d)* buttons will similarly cause the editing state of the selected instance to change from its current state to a complete or deletion state. The state of the instance will only be changed to a complete state if all of the attribute values are valid. Another result of

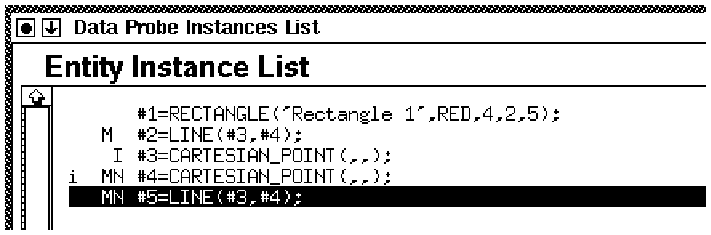
changing the editing state of an instance from the instance list is that the window will be removed from the screen. As a result of the SEE window being removed the *M* is also removed from preceding the instance in the list (see Figure 21).



```
     #1=RECTANGLE('Rectangle 1',RED,4,2,5);
  M  #2=LINE(#3,#4);
   I #3=CARTESIAN_POINT(,,);
  MN #4=CARTESIAN_POINT(,,);
  MN #5=LINE(#3,#4);
```

FIGURE 21 Result of Pressing the Save Incomplete Button From the Entity Instance List Window

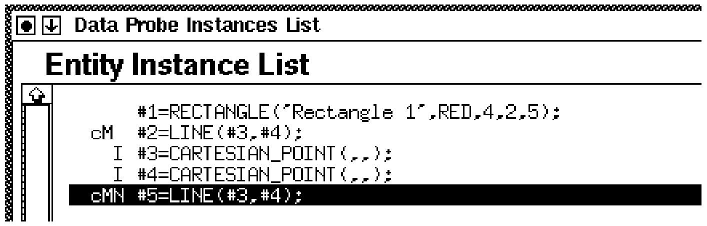
- Another way to change the editing state of an instance from the entity instance list is to use keystroke commands.
- Entity instance cartesian_point with instance identifier *4* should now be selected in the entity instance list. If it is not select it. Type *i* from the keyboard. This will cause *i* to appear preceding the entity instance and information marks *M* and *N* (see Figure 22). Now you may either press the button labelled *Execute (x)* or type the corresponding keystroke command *x* to execute the command to save instance with instance identifier *4* to an incomplete state. The results will be the same as using the *Save Incomplete (i)* button to do the command as you did with the previous entity instance.



```
       #1=RECTANGLE('Rectangle 1',RED,4,2,5);
    M  #2=LINE(#3,#4);
     I #3=CARTESIAN_POINT(,,);
i   MN #4=CARTESIAN_POINT(,,);
    MN #5=LINE(#3,#4);
```

FIGURE 22 Keystroke Command mark to Save an Instance to an Incomplete Editing State from the Entity Instance List Window

- Commands that are executed in this window using buttons operate on the selected instance immediately. Commands issued using keystrokes in this window are executed in batch mode. Batch mode refers to a batch of commands being built up and all of them being executed at one time when told to do so some point later. The keystrokes for the commands are indicated in parentheses on the buttons that implement the same command. The batch of commands that will be executed when the execute command is issued are noted by lower case letters preceding the entity instances in the entity instance list.
- Select the entity instance that has instance identifier *2*. It should have an *M* preceding it. Type *c* and a *c* should appear preceding the entity instance entry. Do this for each instance that has *M* preceding it. Now each instance entry in the entity instance list that has an *M* preceding it should also have a *c* preceding the *M* (see Figure 23). Type the execute keystroke command *x*. All of the SEE windows that are on the screen should remove themselves from the screen and the *M*'s preceding the instances in the list should go away as well.



Data Probe Instances List

Entity Instance List

```
    #1=RECTANGLE('Rectangle 1',RED,4,2,5);
cM  #2=LINE(#3,#4);
  I #3=CARTESIAN_POINT(,,);
  I #4=CARTESIAN_POINT(,,);
cMN #5=LINE(#3,#4);
```

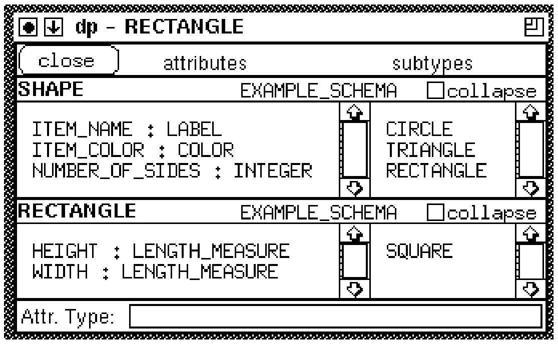
FIGURE 23 Close Keystroke Commands in the Entity Instance List Window

- To place a SEE window on the screen for editing an entity instance select the instance from the entity instance list and press the *Modify (m)* button. To place a SEE window on the screen for an entity instance in view-only mode select an instance from the list and press the *View (v)* button. You may also use the keystroke command as indicated on the commands buttons for placing SEE windows on the screen.

11.8 Displaying Entity Type Information in SED Windows

STEP Entity Descriptor (SED) windows may be placed on the screen from several places. This section has you place SED windows on the screen from each of these places.

- Select the entry for entity type *Rectangle* in the Entity Type List window. Press the *Type Information (t)* button at the bottom of the Entity Type List window. A SED window will be placed on the screen displaying information from the information model about entity type *Rectangle* (see Figure 24).



dp - RECTANGLE

close attributes subtypes

SHAPE EXAMPLE_SCHEMA ☐collapse

```
ITEM_NAME : LABEL                CIRCLE
ITEM_COLOR : COLOR               TRIANGLE
NUMBER_OF_SIDES : INTEGER        RECTANGLE
```

RECTANGLE EXAMPLE_SCHEMA ☐collapse

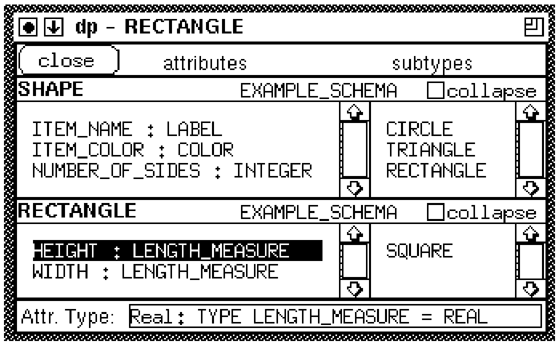
```
HEIGHT : LENGTH_MEASURE          SQUARE
WIDTH : LENGTH_MEASURE
```

Attr. Type:

FIGURE 24 SED Window for Entity Type Rectangle

- The SED window contains entity description blocks from top to bottom. The order from top to bottom indicates the inheritance hierarchy of the entity type that you selected from the most distant ancestor entity type down to the selected entity type. The left side of the entity description block for each entity type contains a list of the attributes that were defined by that entity type. The right side of the entity description block for each entity type contains a list of the subtypes defined from the entity type. Located to the right of the entity type name is the schema name in which the entity type was defined. To the right of the schema name is a button that can be used to close the information contained within the body of the entity description block part of the SED window.

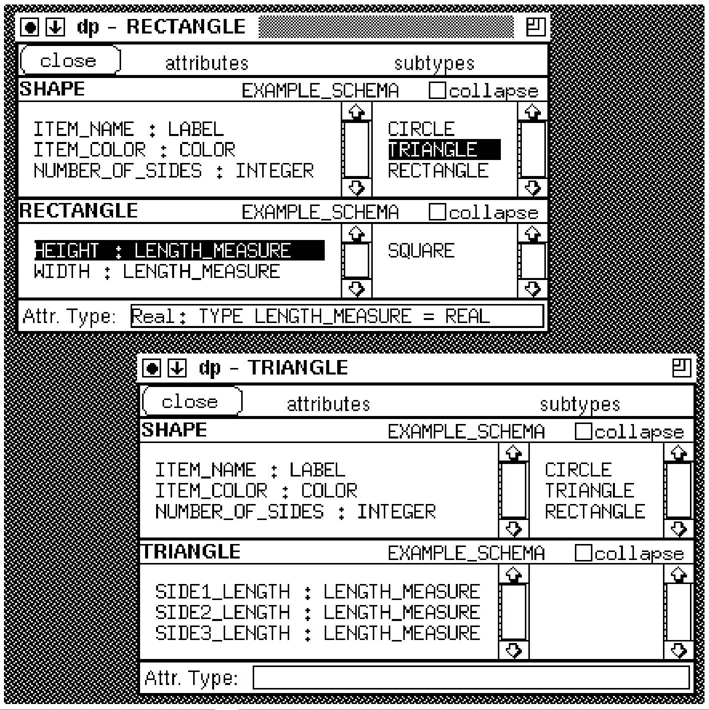
■ Double click on one of the attributes listed on the left side of the SED window. If your double click was successful the message area at the bottom of the SED window should now display more detailed information about the attribute type (see Figure 25).



```
SHAPE
ITEM_NAME : LABEL              CIRCLE
ITEM_COLOR : COLOR             TRIANGLE
NUMBER_OF_SIDES : INTEGER      RECTANGLE

RECTANGLE
HEIGHT : LENGTH_MEASURE        SQUARE
WIDTH : LENGTH_MEASURE

Attr. Type: Real: TYPE LENGTH_MEASURE = REAL
```

FIGURE 25 Detailed Attribute Information For Attribute Height

■ Double click on one of the subtypes listed on the right side of the SED window. If your double click was successful a SED window should appear on the screen displaying information from the information model for the entity type that you just double clicked on (see Figure 26).

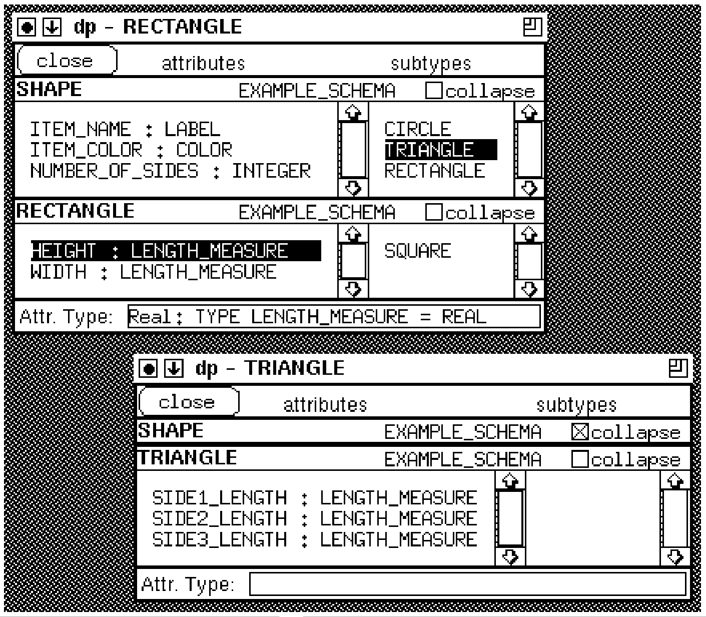


```
SHAPE
ITEM_NAME : LABEL              CIRCLE
ITEM_COLOR : COLOR             TRIANGLE
NUMBER_OF_SIDES : INTEGER      RECTANGLE

RECTANGLE
HEIGHT : LENGTH_MEASURE        SQUARE
WIDTH : LENGTH_MEASURE

Attr. Type: Real: TYPE LENGTH_MEASURE = REAL
```

```
SHAPE
ITEM_NAME : LABEL              CIRCLE
ITEM_COLOR : COLOR             TRIANGLE
NUMBER_OF_SIDES : INTEGER      RECTANGLE

TRIANGLE
SIDE1_LENGTH : LENGTH_MEASURE
SIDE2_LENGTH : LENGTH_MEASURE
SIDE3_LENGTH : LENGTH_MEASURE

Attr. Type:
```

FIGURE 26 Triangle SED Window Opened From Rectangle SED Window

- Try opening and closing part of a SED window by clicking on the button labelled *collapse*. This is used to hide duplicate information between SED windows when two windows are displaying entities that are inherited from the same ancestors (see Figure 27).



```
dp - RECTANGLE
close        attributes                subtypes
SHAPE                  EXAMPLE_SCHEMA  ☐collapse
 ITEM_NAME : LABEL                   CIRCLE
 ITEM_COLOR : COLOR                  TRIANGLE
 NUMBER_OF_SIDES : INTEGER           RECTANGLE
RECTANGLE              EXAMPLE_SCHEMA  ☐collapse
 HEIGHT : LENGTH_MEASURE             SQUARE
 WIDTH : LENGTH_MEASURE
Attr. Type: Real: TYPE LENGTH_MEASURE = REAL

dp - TRIANGLE
close        attributes                subtypes
SHAPE                  EXAMPLE_SCHEMA  ☒collapse
TRIANGLE               EXAMPLE_SCHEMA  ☐collapse
 SIDE1_LENGTH : LENGTH_MEASURE
 SIDE2_LENGTH : LENGTH_MEASURE
 SIDE3_LENGTH : LENGTH_MEASURE
Attr. Type:
```

FIGURE 27 Triangle SED Window Partially Collapsed to Hide Duplicate Information

- The last way to open a SED window is from a SEE window. Use one of the methods shown earlier to open a SEE window. The SEE window offers two methods of opening a SED window for the entity type of the instance. The first is to press the *type information* button. The second is to use the type information keystroke command *^x t*. Use one of these methods to open a SED window displaying entity type information for the instance in the SEE window (see Figure 28).
- You may close the SED windows you have opened by pushing the *close* button on the top left of the SED windows.

11.9 Save Your Work to a File

- The Data Probe provides functionality for saving your work to either a STEP Exchange file or a Working Session file. The Data Probe will write its entity instance list to a STEP Exchange file only if all of the instances are able to be validated. The Data Probe allows you to save unfinished work to a Working Session file. A Working Session file contains editing state marks that are associated with the instances in the entity instance list along with the instances.
- Since there are incomplete instances in the entity instance list the Data Probe will not allow you to save your work to a STEP Exchange file. Follow these steps to save your work to a Working Session file. Point the mouse cursor over the menu bar on the Data Probe window that is labelled *File Management*. Press and hold the left mouse button down and the File Management menu will pull down and draw itself

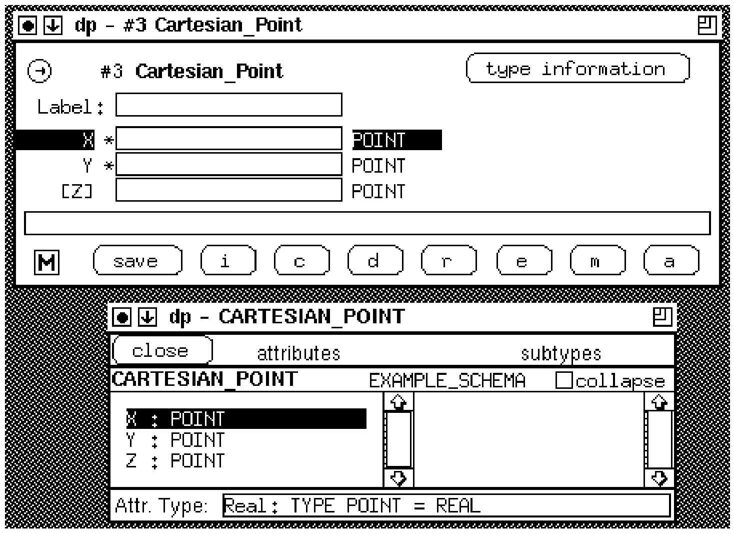


FIGURE 28 SED Window For Entity Type Cartesian_Point Opened From SEE Window For an Instance Of Cartesian_Point

(see Figure 29). Slide the mouse cursor down over the menu entry that says *Write Working File* and release the mouse button. A file chooser dialog box will appear on the screen. Click the mouse inside of the editable box and type in the name of a file in which to save your work (see Figure 30). Press the button labelled *Execute* and the file will be written.

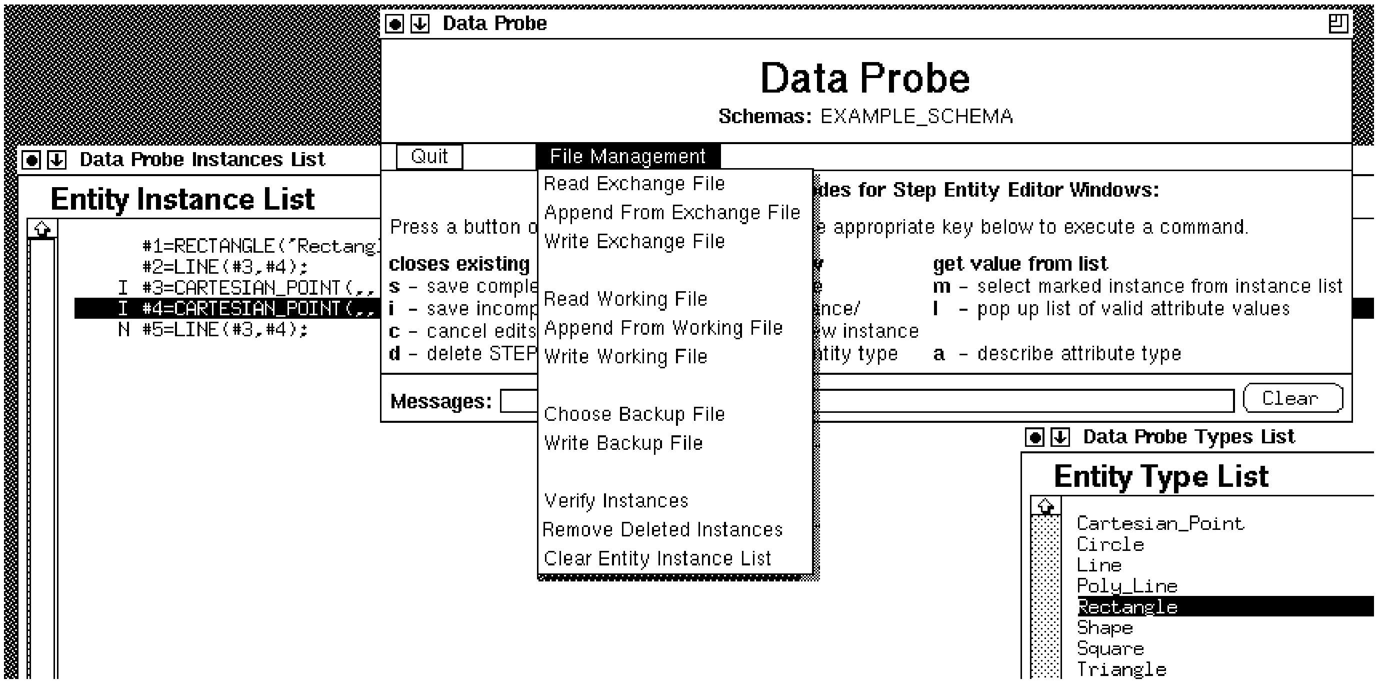
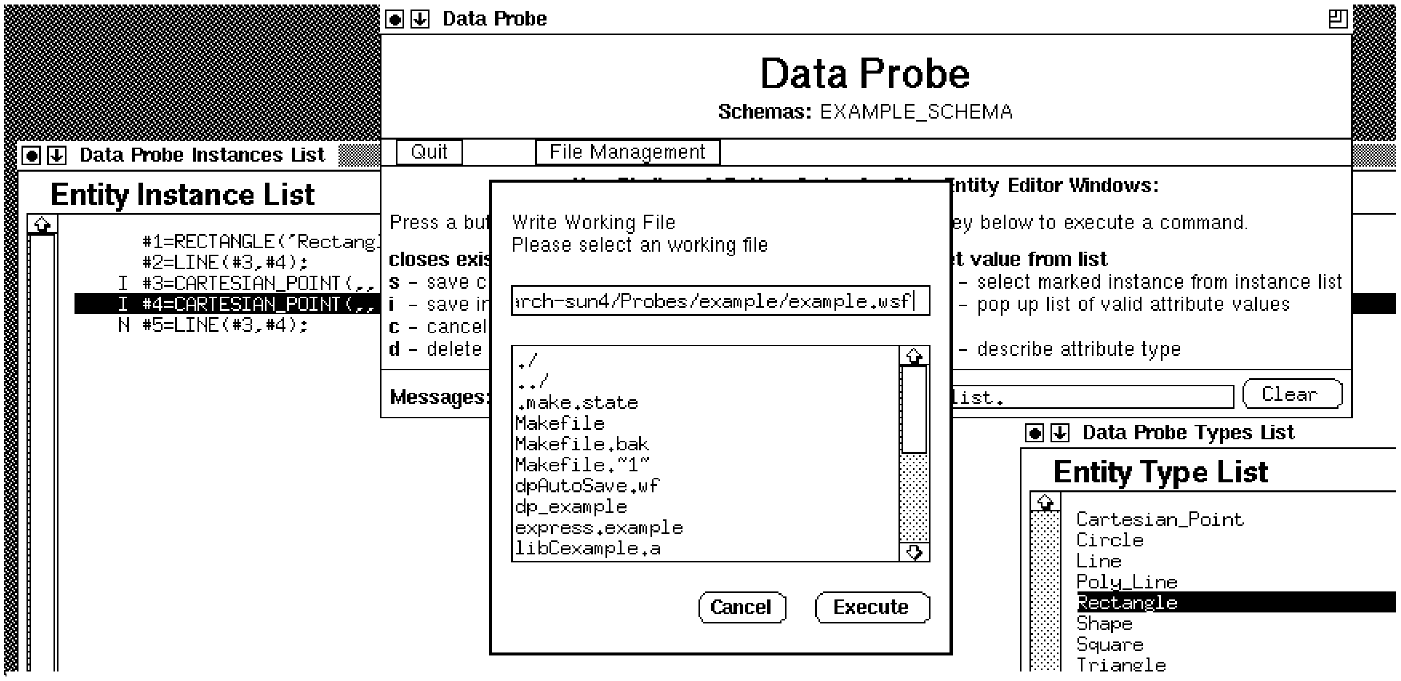


FIGURE 29 Data Probe File Management Menu



```
#1=RECTANGLE('Rectang
#2=LINE(#3,#4);
I #3=CARTESIAN_POINT(,,
I #4=CARTESIAN_POINT(,,
N #5=LINE(#3,#4);
```

```
Press a bu
closes exis
s - save c
i - save in
c - cancel
d - delete
Messages:
```

```
ntity Editor Windows:
ey below to execute a command.
t value from list
- select marked instance from instance list
- pop up list of valid attribute values
- describe attribute type
list.
```

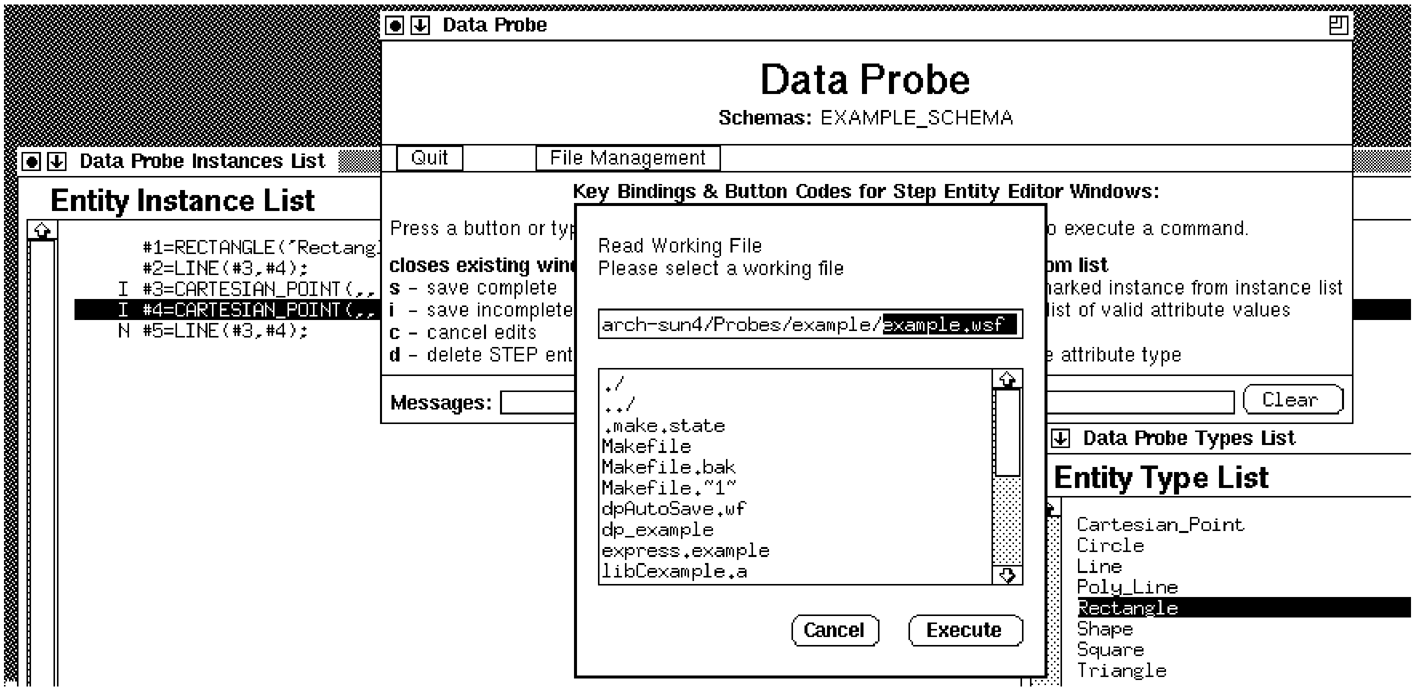
```
arch-sun4/Probes/example/example.wsf
./
../
.make.state
Makefile
Makefile.bak
Makefile.~1~
dpAutoSave.wf
dp_example
express.example
libCexample.a
```

```
Cartesian_Point
Circle
Line
Poly_Line
Rectangle
Shape
Square
Triangle
```

FIGURE 30 File Chooser Dialog Box for Writing a Working Session File

11.10 Read a File

- Next choose the *Clear Entity Instance List* menu option from the *File Management* menu just as you chose the command to write a Working Session File above. This command will clear all of the instances from the entity instance list.
- Now choose the *Read Working File* menu option from the *File Management* menu. The file chooser dialog box you saw a minute ago will reappear (see Figure 31). It will still have the name of the file you just wrote in the editable box. Press the *Execute* button to read in the Working Session file you just wrote. The Working Session file contains the editing state of each instance that was written. It doesn't however save information about which instances were being edited on the screen. It was not necessary to clear the entity instance list in the last step. Reading a file by default replaces any instances that exist when the file is read. If you want to append instances to the ones in the entity instance list choose the option to append a file.



Data Probe

Data Probe

Schemas: EXAMPLE_SCHEMA

Key Bindings & Button Codes for Step Entity Editor Windows:

Press a button or typ… …o execute a command.

closes existing win… …om list

s - save complete …arked instance from instance list

i - save incomplete …list of valid attribute values

c - cancel edits

d - delete STEP ent… …e attribute type

Messages:

Data Probe Instances List

Entity Instance List

```
     #1=RECTANGLE('Rectang
     #2=LINE(#3,#4);
  I  #3=CARTESIAN_POINT(,,
  I  #4=CARTESIAN_POINT(,,
  N  #5=LINE(#3,#4);
```

Read Working File
Please select a working file

arch-sun4/Probes/example/example.wsf

```
./
../
.make.state
Makefile
Makefile.bak
Makefile.~1~
dpAutoSave.wf
dp_example
express.example
libCexample.a
```

Data Probe Types List

Entity Type List

```
Cartesian_Point
Circle
Line
Poly_Line
Rectangle
Shape
Square
Triangle
```

FIGURE 31 Reading a Working Session File

12 References

[Clark90] Clark, S. N., An Introduction to The NIST PDES Toolkit, NISTIR 4336, National Institute of Standards and Technology, Gaithersburg, MD, May 1990

[ISO1] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration— Product Data Representation and Exchange — Overview and Fundamental Principles*, Draft International Standard, ISO TC184/SC4, 1992.

[ISO11] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Description Methods: The EXPRESS Language Reference Manual*, Draft International Standard, ISO TC184/SC4, 1992.

[ISO21] International Organization for Standardization, *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Clear Text Encoding of the Exchange Structure*, Draft International Standard, ISO TC184/SC4, 1992.

[Linton91] Linton, M., InterViews Reference Manual Version 3.0-alpha, Computer Systems Laboratory, Departments of Electrical Engineering and Computer Science, Stanford University, Silicon Graphics, January 1991.

[Mitch91] Mitchell, M., A Proposed Testing Methodology for STEP Application Protocol Validation, NISTIR 4684, National Institute of Standards and Technology, Gaithersburg, MD, September 1991.

[Mitch92] Mitchell, M. J., Morris, K. C. The Use of Application Model Validation in Testing a Proposed Standard, *Proceedings of the Sixth Annual ASME Database Symposium - Engineering Data Management: Key to Integrated Product Development*, American Society of Mechanical Engineers, New York, August 1992.

[Morris91a] Morris, K.C., McLay, M. and Carr, P. J., Validation Testing System Requirements, NISTIR 4676, National Institute of Standards and Technology, Gaithersburg, MD, September 1991.

[Morris91b] Morris, K.C., Mitchell, M.J. and Sauder, D. Validating STEP Application Protocols at the National PDES Testbed, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, November 1991.

[Morris92a] Morris, K.C., Architecture for the Validation Testing System Software , NISTIR 4742, National Institute of Standards and Technology, Gaithersburg, MD, January 1992.

[Morris92b]	Morris, K. C., Sauderd, D., Ressler, S., Validation Testing System: Reusable Software Component Design, NISTIR 4937, National Institute of Standards and Technology, Gaithersburg, MD, October 1992.
[Scheifler89]	Scheifler, R.W., X Protocol Reference Manual for Version 11, O’Reilly and Associates, Inc., Sebastopol, CA, 1989.

Appendix A: Background Terms

The following terms are used throughout the paper. All but the last two definitions are from the Express Language Reference Manual [ISO11].

- **schema** - A collection of items forming part or all of a model.
- **type** - A representation of a domain of valid values.
- **entity type** - A type which represents a collection of conceptual or real-world physical objects which have common properties.
- **attribute** - A trait, quality or property that is a characteristic of an entity.
- **entity instance** - Data values that have meaning because they relate to a defined entity type.
- **inheritance** - An entity type may have one or more entity types inherited from it. An entity type is said to be a parent or supertype of the entity types that are inherited from it. The entity types inherited from an entity type are said to be children or subtypes of the supertype entity type. Each of the subtype entity types may serve as supertypes of more entity types inherited from them. This may continue indefinitely forming a hierarchical arrangement of entity types. Supertypes and supertypes of supertypes, etc. are called ancestors. A subtype entity type inherits (aquires) all attributes defined in any ancestor entity type and may define additional attributes that will be inherited by its subtypes.

The Express in Figure 1 shows definitions for two entity types: *shape* and *circle*.

```
TYPE color = ENUMERATION OF (red, green, blue);
END_TYPE;

ENTITY shape
SUPERTYPE OF (ONEOF circle);
    color : OPTIONAL color;
    number_of_sides : INTEGER;
END_ENTITY;

ENTITY circle
SUBTYPE OF (shape);
    radius : real;
END_ENTITY;
```

FIGURE 1 Sample Express Entity Definitions

Shape has two attributes -- *color* which is of type enumeration and *number_of_sides* which is of type integer. Entity type *circle* is inherited from entity type *shape*. *Circle* adds a new attribute, *radius*, to its set of inherited attributes (*color* and *number_of_-sides*). Attribute *radius* has type real. Figure 2 shows examples of instances of entity types *shape* and *circle*. The last three columns of each row contain the values that make up the instance.

Entity Type	*color* (attribute of *shape*)	*number_of_ sides* (attribute of *shape*)	*radius* (attribute of *circle*)
shape	red	1	
shape	blue	4	
circle	red	1	29.31
circle	blue	1	4.7

FIGURE 2 Entity Instance Examples

Appendix B: Example Schema

(* This information model written in Express is created solely for illustration of Data Probe functionality. This information model was used to create the Data Probe tool used in the figures and in the tutorial. *)

```
SCHEMA example_schema;

TYPE label = STRING;
END_TYPE;

TYPE color = ENUMERATION OF (red, green, blue, yellow, orange, white, black,
brown);
END_TYPE;

TYPE length_measure = REAL;
END_TYPE;

TYPE point = REAL;
END_TYPE;

ENTITY shape
SUPERTYPE OF (ONEOF (circle, triangle, rectangle));
    item_name : label;
    item_color : OPTIONAL color;
    number_of_sides : INTEGER;
END_ENTITY;

ENTITY circle
SUBTYPE OF (shape);
    radius : real;
END_ENTITY;

ENTITY triangle
SUBTYPE OF (shape);
    side1_length, side2_length, side3_length : length_measure;
END_ENTITY;

ENTITY rectangle
SUPERTYPE OF (square)
SUBTYPE OF (shape);
    height : length_measure;
    width : length_measure;
END_ENTITY;
```

```
ENTITY square
SUBTYPE OF (rectangle);
END_ENTITY;

ENTITY cartesian_point;
    x : point;
    y : point;
    z : OPTIONAL point;
END_ENTITY;

ENTITY line;
    end_point_one : cartesian_point;
    end_point_two : cartesian_point;
END_ENTITY;

ENTITY poly_line;
    points : LIST OF line;
END_ENTITY;

END_SCHEMA;
```