XDE Project Management Tools User Guide

XEROX

610E00210
September 1985

Xerox Corporation
Office Systems Division
2100 Geng Road
MS 5827
Palo Alto, California 94303

Printed in U.S. A.

TABLE OF CONTENTS

Overview

Distributed network programming environments such as XDE present record-keeping and project management problems not found in smaller efforts. Large-scale programming projects involve many programmers and many different versions of the software being developed. XDE's Project Management Services and Tools are designed to provide solutions for the project management problems inherent in large distributed network programming projects.

The usefulness of Project Management Services and Tools is not limited to program development projects, however. They can be productively applied to any project that deals with information distributed across a network. For example, Xerox uses the tools extensively for a system to enter, maintain, and report on software and hardware problems.

The Project Management Services and Tools consist of

Adobe, a network-oriented database management system;

Librarian, for controlling file access.

Services reside on network servers. They manage the files and regulate access to them. Tools reside on your workstation and allow you to access files and manipulate information.

This manual tells you how to use Adobe and Librarian Tools. Adobe and Librarian Services are covered in the *Project Management Services System Administrator's Guide*.

What you need to know before using this manual

In writing this manual we have made certain assumptions. The first is that Adobe and Librarian will be useful not only to programmers but also to nonprogrammers working in a programming environment.

If you are not a programmer, we have assumed that since you are working in a programming environment you are familiar with general computer functions and have some knowledge of database structure and common database operations such as sorting, report generation, and entry of new data. We have taken it for granted that you are already familiar and comfortable with XDE and specifically, that you know how to

use the mouse, manipulate windows, and retrieve files from a server.

If you are a novice user, you should not attempt to use this manual until you have completed the XDE Tutorial. Consult your supervisor to find out what other training materials are available to you.

PART I ADOBE TOOLS

1. INTRODUCTION TO ADOBE

1.1 Adobe defined

Adobe is a network-oriented DBMS (database management system) designed specifically to be used in a program development environment for tracking the status of project components and for requesting actions to be taken. Adobe also provides an efficient and organized way to manage problem reports. Although designed for the kinds of applications common in a programming environment, the Adobe DBMS is easily adaptable to more general business database applications, such as personnel records, inventory tracking, and customer directories.

The Adobe DBMS has two main components: Adobe services and Adobe tools.

Adobe services reside on network servers and manage the Adobe databases. They regulate access to the databases and maintain description files, accelerator files, and default user files. Adobe services are maintained by the Adobe system administrator, who is also responsible for creating new databases according to criteria furnished by the users.

Adobe tools reside on individual workstations and manage data in already existing databases. They are the means by which the user communicates with Adobe services, which do the actual database manipulations. The tools are used to enter data, query and sort it, and generate reports.

XDE's flexible windowed access from individual workstations to Adobe services makes it possible for several databases to be simultaneously active. A workstation may also have multiple Adobe tools active at one time, performing different tasks with selected network databases. Adobe services limit access to a database as required; for example, while records are being edited. Figure 1.1 illustrates typical Adobe usage on a network with multiple Adobe services and multiple workstations.

This manual covers only the Adobe tools The creation and management of Adobe databases are covered in the *Project Management Services System Administrator's Guide*. The workstation user who wishes to create a new database should see his or her Adobe system administrator.

1.2 Adobe terms

Adobe uses some special terms that you will need to know:

A database is referred to in Adobe as a *system*.

A database record is referred to as an *action request,* or AR.

To make your adjustment to these new terms easier, we will remind you of them a few times in the early parts of the manual, and occasionally we will use *database system* or *AR record* to help you associate the Adobe terms with the conventional terms.

1.3 The Adobe tools

The Adobe tool set consists of the following:

Submit Allows the user to enter new records into a database, according to a template specified in the database definition. Each record is assigned an ID number when it is entered, sometimes called an ARId.

Edit Allows examination and editing of specific database records. You may gain exclusive access to an entry by checking out the record. You may then edit it and check it back in.

Query Searches database records and produces a list of ID numbers for those records that satisfy the criteria specified by the user in the Query tool fields. This list, called a query list, is used by the Edit, QueryList, and Report tools.

QueryList Combines several query lists to produce a new list according to a user-specified operation. Operations such as union, intersection, and complement can be applied to existing query lists to produce a new query list without going through the entire query process again.

Report Generates a report based on records in a query list. The contents and format of the report are designed by the user with Report tool field selection options.

Sort Sorts reports generated by the Report tool, based on a set of sort criteria (keys) selected by the user The Sort tool can also break up a larger report into smaller ones.

1.4 The Adobe tool window

Each Adobe tool has its own customized window in which allowable commands and the fields for the particular database in use are displayed.

The Adobe window is divided into either three or four subwindows, depending on which Adobe tool is loaded. Figure 1.1 illustrates the Adobe tool window as it appears when the Submit tool is in use and no database has yet been specified.

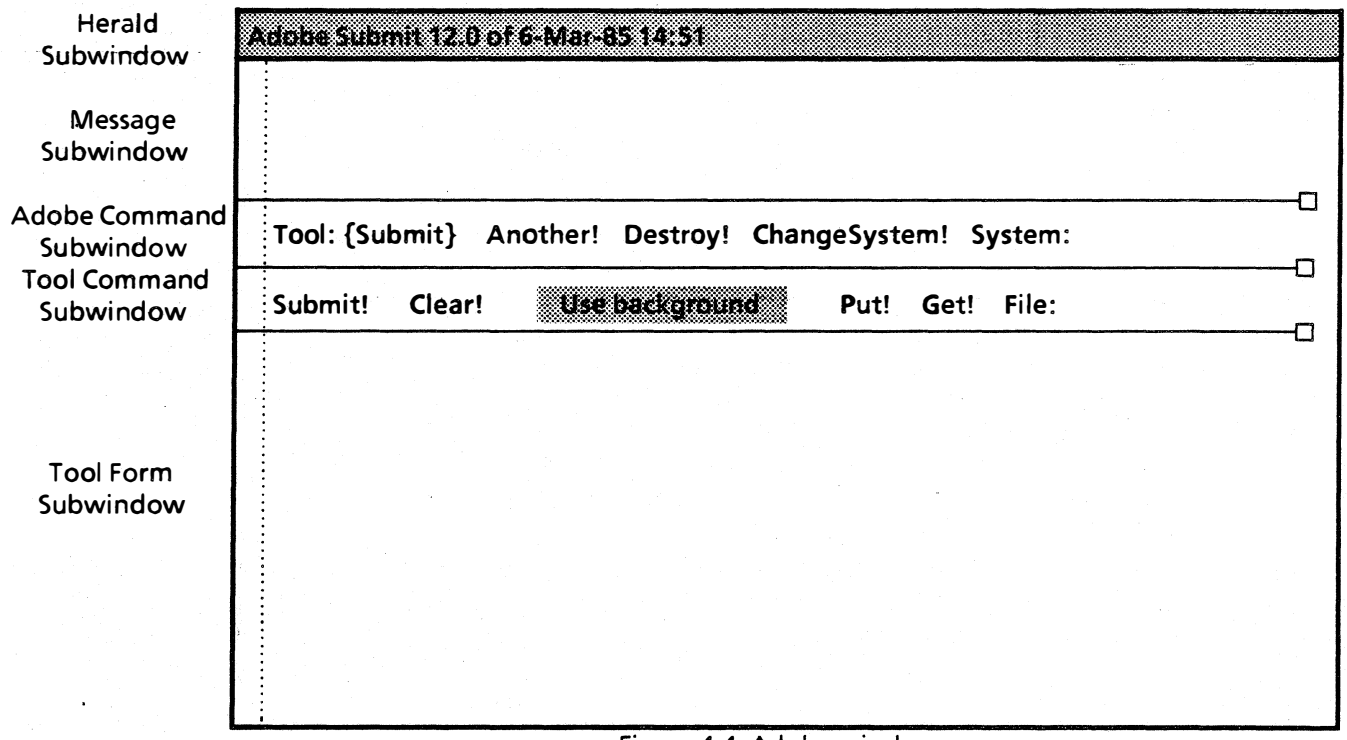

Figure 1.1 Adobe window

1.4.1 Herald subwindow

Displays the name of the tool, the version date and time of the tool, and the name of the database system currently in use. This window is the same for all the tools.

1.4.2 Message subwindow

Displays error and status messages. This window is the same for all the tools. See Chapter 5 for possible error messages and what to do about them.

1.4.3 Adobe command subwindow

Through this window you select the tool you want to use and the database system you need to access. This window is the same for all the tools. Note that a colon indicates a field to be filled in; an exclamation point indicates a command that is activated by selecting it with the mouse.

Tool: Identifies the Adobe tool currently active in this window. To change tools, use the mouse to place the cursor over the word **Tool** and chord (press both mouse buttons at once). A menu will be displayed, showing the tool options **Submit**, **Query**, **QueryList**, **Report**, **Sort**, and **Edit**. Select the option you want and release the buttons. The window you are using will be

changed to the window for the tool you selected. If you chose **Edit**, the **Tool** field will now read **{Edit}** instead of **{Submit}** and the Herald subwindow will now display **Adobe Edit** as the name of the window.

Another! Creates another Adobe tool window. Once you have selected **Another!**, the cursor will change form and display a graphic mouse. Press the POINT (left) button on the mouse at the place on the screen where you want the top-left corner of the new window. Pressing the ADJUST (right) button instead of point will abort the creation of another Adobe window. A window created by **Another!** will be for the same tool and the same database as the window from which you invoke the command. You may then change the tool as described under **Tool** and select a new database using **ChangeSystem!**. (Remember that in Adobe, *system* means *database*.)

Destroy! Destroys the Adobe tool window from which the command is invoked. You may destroy a window when you have finished working with it.

ChangeSystem! Changes from the current database system to the database system specified in the **System** field. You may also change systems by chording anywhere in the Adobe tool window and selecting a system from the pop-up menu. Items in this menu are specified in the **User.cm** file.

System: The field that specifies the Adobe system to be accessed. Type in a system name and invoke **ChangeSystem!**

1.4.4 Tool command subwindow

Each Adobe tool has its own set of commands. The commands shown in Figure 1.1 are for the Submit tool. Commands for the other tools will be shown and explained in the respective sections for each tool.

1.4.5 Tool form subwindow

All Adobe tools except **Sort** have form subwindows . The form subwindow displays the selected fields for the system being used, on which the tool may act. Sample forms for several different types of databases are illustrated in the sections explaining each tool.

Since the form that appears in the subwindow is based on the field structure of the database you are using, the form will be different for each database system (unless you have two or more databases with identical field structures).

When the form appears, some of the fields may already be filled in. Some fields may be subject to **SystemMust** defaults, which are set up at the time the database system is defined. For example, **Date of Submission** may be set up to automatically show the date and time of submission, or the name of the

workstation user may be automatically entered in a **Submitted by** field. **SystemMust** defaults cannot be changed by the user.

Fields may also be filled in automatically if a default value for that field is specified in the **.user** file associated with the database system you are using.

To use the form subwindow, you need to be familiar with the kinds of fields used by Adobe systems.

1.4.5.1 Field types

ARId: Each AR record is automatically assigned a unique ID number by the system. This is a read-only field. An ARId cannot be changed by the user, even if it appears in the form subwindow.

Numeric: Numeric fields contain numerical information only, and the system will not allow non-numeric data to be entered. The screen will blink to alert you that you have tried to enter illegal characters in a numeric field.

DateTime: This type of field is used exclusively to record a date and the time of day. The date is entered in the form dd-mm-yy (day, month, and year); for example, 8-Mar-85 (the day does not require a leading zero). The time is entered in the form hh:mm:ss (hour, minutes, seconds). Dates entered with no time following them will have 00:00:00 added to them.

DateTime fields may be left empty, but if an invalid date and time are entered, they will be replaced by the current date and time.

BoundedString Bounded string fields have a specified length limit. Most character string fields filled in by the user are of this type.

Adding a character that would cause a bounded string field to exceed its length limit will cause the screen to blink, and the character will be discarded.

By limiting string length, Adobe can maintain the data in these fields on the Adobe Service, which allows bounded string fields to be used as query keys (see Chapter 3, section 3).

UnboundedString Unbounded string fields have no length limit. They are suitable for entering information of upredictable length, such as commentaries and explanations. However, since the length is unlimited, they cannot have their data maintained on the Adobe Service and therefore they cannot be queried on.

A moderate amount of text may be put in an unbounded string field, but if you need to write the equivalent of several pages of text, it is best to create a file containing the text and point to that file's location from the AR record. This allows the tools to display the values of a record more quickly.

Enumerated Enumerated fields have fixed sets of preset values, and no values outside of the fixed set will be accepted in the field.

Enumerated fields are identified in the form subwindow by a set of curly brackets following the colon; for example,

Location: {}. When you chord over the word **Location**, a pop-up menu displays the acceptable values for that field, such as New York, Chicago, Los Angeles, and Coyote Gulch. Selecting your choice causes that value to be entered in the field.

Enumerated fields are of value when the possible entries in a field are a limited number of preset choices. The menu not only allows information to be entered faster but also protects against invalid entries.

Enumerated fields are of two types: independent and dependent. An independent enumerated field has a fixed set of values that never changes.

A dependent enumerated field has several possible sets of values. As its name suggests, the set of values that prevails depends on what value is first selected from the parent independent field. For example, a dependent enumerated field for **Location** might be **Salesperson**. If the selection for **Location** is New York, the **Salesperson** menu might read McConnel, Martinez, and Matsumoto. If the selection for **Location** is Los Angeles, the **Salesperson** menu might then display Garfinkel, Marinkovich, and Carter. And so on, for each location.

NIL, indicating that there is no specified value, is automatically provided as a possible value in all sets of values for enumerated fields. When an independent enumerated item is changed, any dependent items are reset to **NIL**. That is, if **Location:** is changed from New York to Coyote Gulch, **Salesperson:** will be reset to **NIL**. A new selection must be made from the **Salesperson:** menu, which will now be set to the dependent enumerated field menu for Coyote Gulch.

ALL may be optionally provided as a value to indicate all values are selected, where appropriate.

1.4.5.2 Sample form subwindow

Below is a sample form as it might appear in the form subwindow when you are using the Edit tool to change an AR record in a system called Personnel Records.

Last Name:

First Name: **Middle Initial:**

SSN = 0

Hire Date:

Current Status: {} **Grade = 0**

Section: {} **Area:** {}

Job Title:

Other Notes:

Edit-by: **Edit-Date:**

The following explains what kind of field each entry in the form requires. The lengths given for the fields and the sample values for enumerated fields are for illustration only.

Last Name: is a bounded string field of 20 characters.

First Name: is a bounded string field of 15 characters.

Middle Initial: is a bounded string field of one character.

SSN: is a numeric field for Social Security number. Since this is a numeric field, you cannot enter dashes in the number.

Hire Date: is a **DateTime** field.

Section: is an independent enumerated field that might have the following values: **{Advanced Development, Services Software, System Software, Workstation Software, Other}**. A pop-up menu showing these values will appear when you chord over **Section**.

Area: is a dependent enumerated field, dependent on **Section**. Its sets of values might be as follows:

If **Section = Advanced Development, Area** may = **{Graphics & Prints, International, Networks, User Interfaces, Workstations, Other}**.

If **Section = Services Software, Area** may = **{Communication Services, Basic Services, Distributed Services, Other}**.

If **Section = System Software, Area** may = **{IO Architecture, Mesa, Pilot, Other}**.

If **Section = Workstation Software, Area** may = **{Basic Workstation, Workstation Documents, Workstation Functions, Workstation Services, Other}**.

A pop-up menu showing the appropriate values (depending on **Section**) will appear when you chord over **Area**.

Current Status: is an independent enumerated field with the following possible values: **{active, retired, terminated, resigned, other}**. These options will be displayed in a pop-up menu when you chord over **Current Status**.

Job Title: is a bounded string field of 40 characters.

Grade: is a numeric field indicating the grade of the employee.

Other Notes: is an unbounded string. It may be used to record general notes about the employee, evaluations, records of transfers between sections, and any other information the personnel manager might want to keep.

Edited-by: is a **SystemMust** default field. It will always be set to the LoginName of the person who is editing the database.

Edit-Date: is another **SystemMust** default field, set to the date and time when the AR record is checked in after editing.

Examples of other types of forms will be given in the sections describing individual Adobe tools. You may wish to refer back to this section to refresh your memory on the types of fields used and what kinds of information each is appropriate for.

If Adobe is not already installed on your workstation, you will need to use your File tool to retrieve **Adobe.bcd** from the release directory. A coworker or your Adobe Services administrator will be able to tell you where the release directory is.

Two types of files allow you to customize Adobe tools and their windows. These files are **User.cm** and **.user** files.

The **User.cm** file resides on your workstation, and is the file you use to set up tool windows and to specify startup defaults to suit your individual requirements.

The **.user** files specify form window layouts and column reports for database systems. A default **.user** file for each system resides on a network Adobe service, and Adobe will retrieve this file if it does not find a **.user** file locally. These files are normally set up by the Adobe system administrator when a new database system is created. You will rarely need to create your own.

The similarity in the names of the two types of files is somewhat confusing, especially since they perform similar functions but operate in different areas. The distinction between the two is that the Adobe section of the **User.cm** file operates on the Adobe tools as a whole and is used to customize the user's entire workstatioin environment, while **.user** files are associated with specific database systems only. The differences will become clearer as you read the sections below.

2.1 Setting up the User.cm file

The **User.cm** file in your development environment can be used to specify certain conditions when Adobe is loaded, such as whether Adobe is active or inactive, the size and location of the Adobe tool window, the initial system to be addressed, and the initial tool to be used. Adobe looks at the data in the **User.cm** file at startup time and whenever a tool is activated from the Inactive menu.

Find the Adobe section in your **User.cm** file. If there is no Adobe section, retrieve **SampleUser.cm** from the release directory and insert the Adobe section into your existing **User.cm**. You may use the Adobe section provided for you in **SampleUser.cm** as is, or you may edit it to your own preferred values.

The Adobe section of **User.cm** allows you to specify a tool of your choice as the default tool when Adobe starts up, designated as follows:

InitialTool: AdobeTool

where AdobeTool is one of the following: Submit, Edit, Query, Report, Sort, QueryList. If a tool is not specified, Adobe will start in Submit.

The Adobe section also allows you to specify the database system(s) Adobe will recognize initially, as follows:

KnownSystems: ["Personnel Records" Tasks "Product Software:OSBU South"]

System names that contain spaces must be enclosed in quotes, and that names of systems not in your domain must include the domain where they reside. You may specify as many known systems as you wish. The Adobe Systems menu will contain all the valid systems you specify in the **KnownSystems** field. Databases not named in the **KnownSystems** field may still be accessed via the **ChangeSystem!** command.

A particular system may be specified as the default startup system, as follows:

InitialSystem: "System Name"

Adobe's initial state may be specified as active or inactive, as follows:

InitialState: active

Window size and location can be specified as follows:

Windowbox: [x:16, y:48, w:480, h:340]

where x and y are pixel coordinates for the screen location of the upper-left corner of the window box; w = width and h: = height, in pixels.

Location of the Adobe tiny window can be specified as follows, using the x and y pixel coordinates:

TinyPlace: [x:452,y:778]

Suppose the job you do most often with Adobe is submitting records to Personnel Records However, you also use a database called Tasks and another called Product Software that does not have an entry in your domain. Your entire Adobe **User.cm** section will look something like this:

[Adobe]
InitialState: active
InitialTool: Submit
InitialSystem: "Personnel Records"
KnownSystems: ["Personnel Records" Tasks "Product **Software:**OSBU South"]

Windowbox: [x:16,y:48,w:480,h:340]
TinyPlace: [x:452,y:778]

2.2 System .user files

Adobe database systems are preset with default values for form layouts, report column widths, and report ordering. These defaults may be overridden with **.user** files.

If the default **.user** file does not suit your needs, you may design your own layout for a tool's form subwindow, selecting only those fields you need for display. You may also set certain fields to display default values. For complete instructions, see Chapter 4, Creating a New Database.

After modification, the **.user** file will reside on your workstation. Adobe looks for a **.user** file locally before it goes to the system default file, so you do not have to do anything other than make the changes you want and store it locally.

2.3 Loading Adobe

Once you have set up your **User.cm** file and retrieved **Adobe.bcd**, you can start Adobe by typing *Adobe*, followed by a carriage return, at the command prompt in your Executive.

If initial state is active:

If the initial state specified in the **User.cm** file is active, an Adobe window will appear on your screen. Which tool window appears is determined by the **InitialTool** field in the **User.cm** file. If no tool is specified, **Submit** will appear.

To change tools and/or systems, or to create additional windows, see the instructions for the Adobe command subwindow in Chapter 1, section 1.4.3.

Remember that as long as you have selected **Use background** in the Tool command subwindow (**Use background** will be highlighted when it is on), you may open as many windows as you like, use several tools simultaneously, and access a different database in each window.

If initial state is inactive:

To call an Adobe window when the initial state is inactive, you must chord in the grey area of your screen to get the Inactive menu.

When you call Adobe this way, the menu will list **Adobe**, with no tool specified. Selecting **Adobe** from the menu will create a window for the tool that is your **InitialTool** in the **User.cm**. If no initial tool is specified, a Submit window will be created. From this window you may create other windows.

2.4 Storing Adobe windows

When you are through working with Adobe, you may store your tools in one of two ways.

If you want to save the window as is, with the current database system and form subwindow intact so that you can resume work later where you left off, save it as a tiny window. You can save as many Adobe windows as you like in this way.

If you have no reason to save the window as is, you may deactivate it. Chord anywhere in the window, select the Window Manager menu, and select Deactivate. The window will then be stored and listed on the Inactive menu. Remember, however, that any data you have in a window when you deactivate it will be lost.

You are now ready to use the Adobe tools. The procedures for using each tool will be described in detail in the following chapters.

3.1 Submit

The Adobe Submit tool is used to enter new records into databases. Figure 3.1 illustrates Submit using a system for recording software action requests called System Software Action Requests.

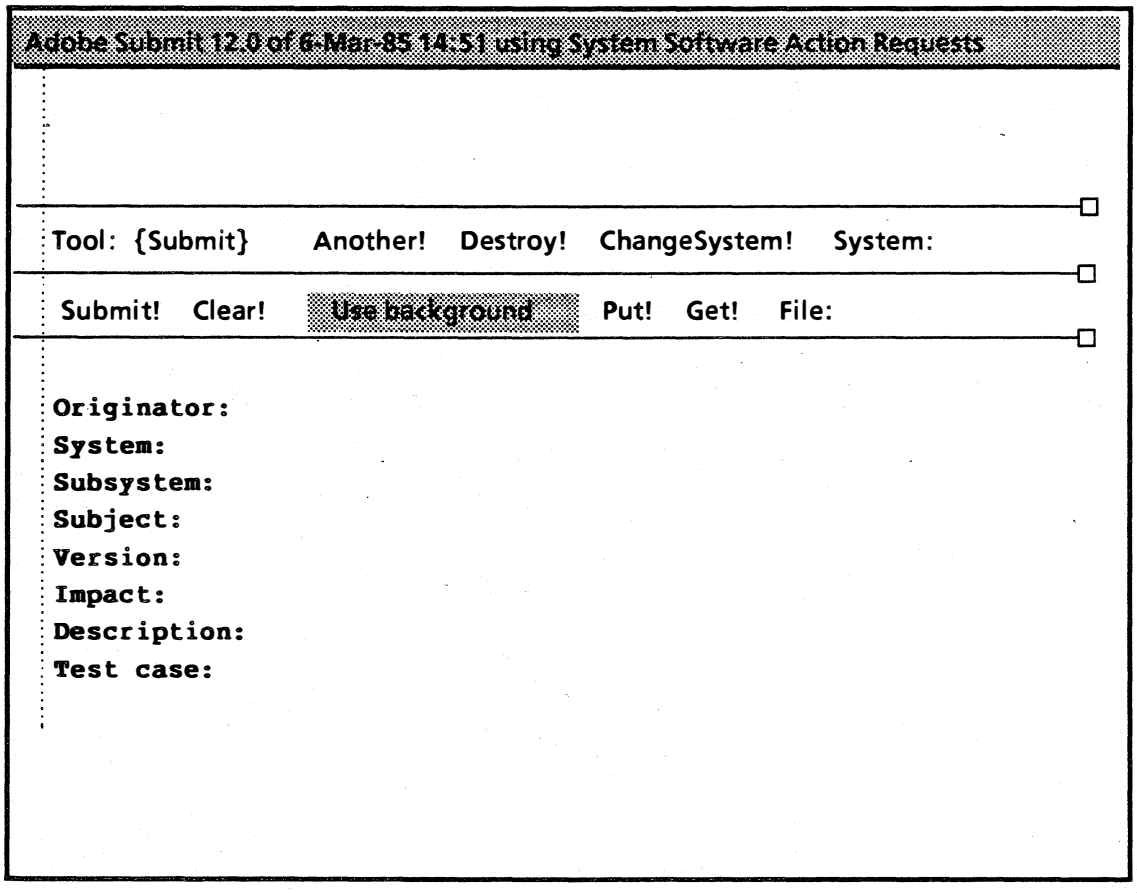

Figure 3.1 Adobe Submit window

3.1.1 Submitting an AR record

An AR record can be submitted using the following procedure:

- Bring up an Adobe tool window. (If you have not yet read **Loading Adobe** in Chapter 2, do so now.)

- Set the **Tool** field in the Adobe command subwindow to Submit, if necessary.
- In the Adobe command subwindow, chord anywhere in the Adobe tool window and select the database you want to use from the Adobe System pop-up menu. Alternatively, you may enter the name of the system you want in the **System** field and select **ChangeSystem!**
- Fill in the fields in the form subwindow.
- Select the **Submit!** command when you have finished entering data.
- If the submit is unsuccessful, save the form by typing a file name in the **File** field and invoking the **Put!** command. The file can be retrieved and resubmitted later by filling in the file name in the **File** field and invoking **Get!**, then **Submit!**

A detailed explanation of the Submit commands follows.

3.1.2 Submit command subwindow

The commands in the Submit command subwindow are as follows:

Submit! Causes the form you have completed in the form subwindow to be submitted. When the form has been submitted successfully, a message appears in the message subwindow identifying it by its newly assigned ID number. After invoking **Submit!**, all fields in the form subwindow become read-only. To enter a new form, you must either wait until the Submit operation is completed or obtain a new window.

Clear! Empties all the fields in the form and resets the defaults. This command will ask for confirmation before clearing if you have edited the form but have not invoked **Submit!** This is to protect the form from disappearing in case you forget to invoke **Submit!** before going on to another form.

Use background Allows you to process commands in more than one Adobe window simultaneously, using a background process. When **Use background** is highlighted, as in the illustration above, the background process is on. You can turn the background process off by clicking on **Use background**. Any time you want to do more than one operation at a time **Use background** should be on (highlighted).

Put! Saves the information you have entered in the form fields to a file specified in the **File** field of the Submit command subwindow. You may wish to save the form if the record could not be submitted.

Get! Retrieves the information stored in the file specified in the **File** field. (Note: if you use another editor to modify a form that you have **Put!** in a file, a subsequent **Get!** may fail, if the special format of the file has not been preserved.

File: The field where you specify the name of the file to be stored or retrieved. The **Get!** and **Put!** commands act upon the name specified in this field, as explained above.

3.1.3 Submit form subwindow

Information for new AR records is filled in via the Submit form subwindow. The form that appears in the subwindow will show all the fields that need to be filled in for the database system you are using. If you don't have all the information when you first submit an AR record, you may leave fields blank and fill them in later using the Edit tool.

You may proceed from one field to the next by pressing **Next** on your keyboard, or you may select the field you want with the mouse. Enumerated fields (identified with curly brackets) will be skipped if you use the **Next** key; to fill them in, chord over them and select from the pop-up menu. When you are through filling in information, store the record by invoking the **Submit!** command.

To use the form subwindow, you need to be familiar with the kinds of fields used by Adobe systems. If you are not already familiar with them and have not read the **Fields** section in Chapter 1, go back and do so now.

The Submit form subwindow is illustrated in Figure 3.1.

3.2 Edit

The Adobe Edit tool is used to examine and modify individual AR records. Figure 3.2 shows Edit using a system called Hardware Action Requests to record hardware problems .

3.2.1 Editing an AR record

An AR record whose number is known can be examined or edited using the following procedure:

- Bring up an Adobe Tool window. (If you have not yet read **Loading Adobe** in Chapter 2, do so now.)
- Set the **Tool** field in the Adobe command subwindow to Edit, if necessary.
- In the Adobe command subwindow, chord anywhere in the Adobe tool window and select the database you want to use from the Adobe System pop-up menu. Alternatively, you may enter the name of the system you want in the **System** field and select **ChangeSystem!**
- Enter the AR record number (ARId) in the **Number** field.
- If you wish to modify the AR record, select **Checkout!** and modify the form items as desired. Alternatively, if you do

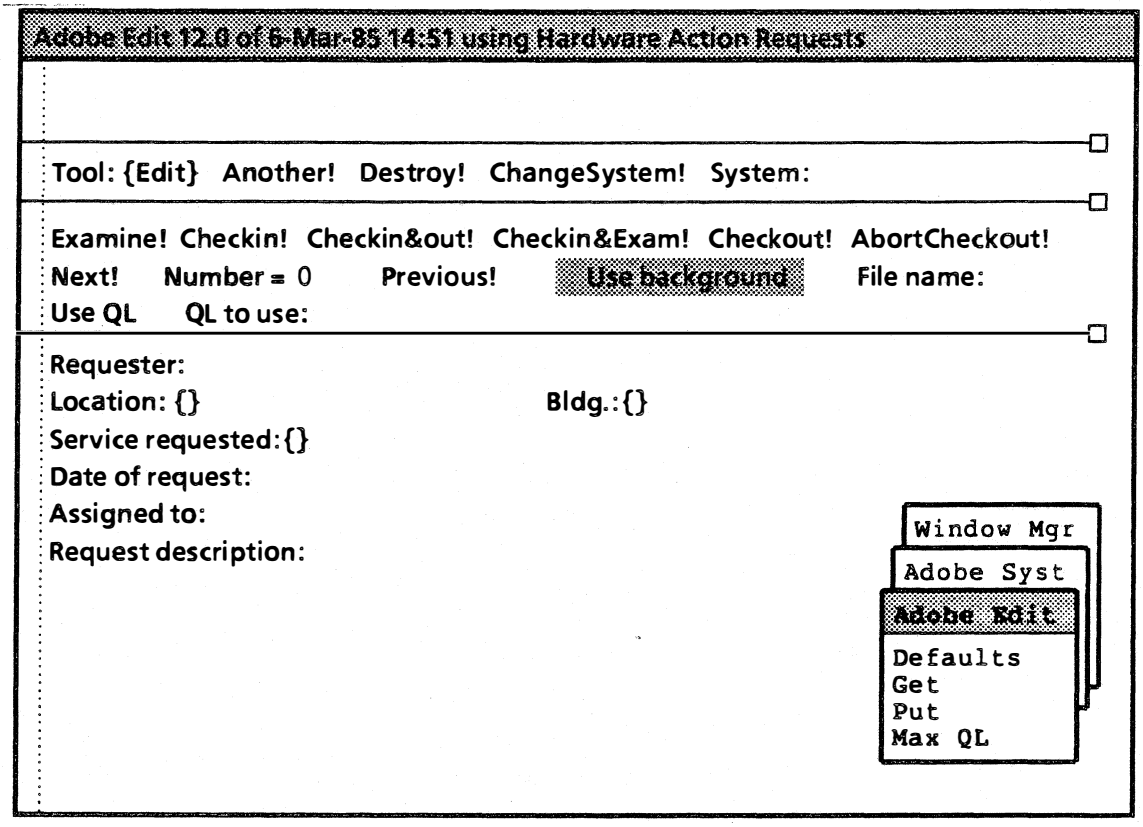

Figure 3.2 Adobe Edit window

not want to check out the record until you have examined it, select **Examine!**

- Check in the modified AR by selecting **Checkin!**

AR records in a query list can be examined or edited using the following procedure:

- Generate a query list (see instructions for **Query** or **QueryList**).
- Chord over **QL to use** and select the name of the query list you wish to use from the pop-up menu. Alternatively, you may type in the name of the query list in the **QL to use** field. Select **UseQL** in the Edit Command subwindow. (**UseQL** will be highlighted if it is on.)
- Select **Next!** to advance to the next AR record in the query list. The number of the AR is displayed in the **Number** field.
- Follow the steps described above for examining or modifying an AR record.

A detailed explanation of the Edit tool commands follows.

3.2.2 Edit command subwindow

The commands in the Edit command subwindow are as follows:

Examine! Retrieves the AR record specified by the ARId number in the **Number** form field and displays it in the form subwindow. (See **Number:** below.) This command does not allow you to modify the AR record. It is used only when you want to read an AR record without changing it.

Checkout! Retrieves an AR record for examination and modification. When an AR is checked out, no one else may alter it until it is checked back in. A message will appear in the window name frame indicating that the record is checked out.

Checkin! Stores the AR record complete with any modifications that you made to it.

Checkin&out! Checks in the AR record currently in your form subwindow and checks out the AR record whose number is indicated in the **Number** form field. This allows you to go directly from one record to another in one operation, just by entering a new number and invoking **Checkin&out!** Checking out a record allows you to modify it.

Checkin&Exam! Checks in the AR record that is currently checked out and displays the AR indicated in the **Number** form field. Remember that the **Examine!** command allows you only to read a record, not to modify it.

AbortCheckout! Checks in the AR record you have checked out without storing any changes you have made to it.

UseQL Instructs Adobe to use the AR records specified in the currently selected query list in the **QL to use** field when **Next!** or **Previous!** is invoked. **UseQL** is turned on by selecting it with the mouse; it will then be shaded on your screen. To turn it off, select it again.

QL to use: Is the name of the query list to use if **UseQL** is on. A pop-up menu that contains the names of the existing query lists for the Adobe system in use is associated with this field. To get the menu, chord over **QL to use** and select the query list of your choice. If the **QL to use** field is empty and **UseQL** is on, the default query list, **SysQL**, is used.

Number: Identifies the ARId of the AR record to be examined or checked out next.

Next! Modifies the **Number** field to the next ARId number in sequence.

If **UseQL** is on, selecting **Next!** will modify the **Number** field to the next ARId number in the query list. If there are no more elements in the query list, the message "Query List Exhausted" will appear in the Adobe message subwindow.

Previous! Modifies the **Number** field to the prior ARId number.

If **UseQL** is on, selecting **Previous!** will modify the **Number** field to the number one step back on the query list. If **Number** is currently the first element in the query list, invoking **Previous!** will cause the message "Query List Exhausted" to appear in the Adobe message subwindow.

Use background Allows you to process commands in more than one Adobe window simultaneously, using a background' process. When **Use background** is highlighted, as in Figure 3.2, the background process is on. You can turn the background process off by clicking on **Use background**. When you want to do more than one operation at a time Use background should be on (highlighted).

File Name: Is the field where you may specify a file for an AR record to be stored or retrieved. The **Get!** and **Put!** commands in the Adobe Edit pop-up menu use the name specified in this field (see section 3.2.4).

3.2.3 Edit form subwimdow

Changes to existing AR records are made via the Edit form subwindow. The form that appears in the subwindow will show all the fields in the database system you are using.

SystemMust default fields, such as the **Date of Request** field in the sample form subwindow in Figure 3.2, cannot be edited.

Enumerated fields (identified by curly brackets) may be edited only by selecting another option from the pop-up menu.

You can proceed from one field to the next by pressing SHIFT and NEXT simultaneously on your keyboard. Using NEXT without SHIFT to proceed through the form will delete the contents of each field. You can also use the mouse to select the field you want. When you are through editing, you store the changes to the record by invoking the **Checkin!**, **Checkin&Out!**, or **Checkin&Exam!** command.

The Edit form subwindow is illustrated in Figure 3.2.

3.2.4 Edit menu

Chording anywhere in the Adobe Edit tool window will bring up the Adobe Systems, Window Manager, and Adobe Edit pop-up menus. Selecting the Adobe Edit menu will give you the following choices: **Defaults**, **Get!**, **Put!**, and **Max QL**.

Defaults Sets fields that have been given default values in the .user file to their default values; other fields are not affected.

Put! Stores the AR record currently in the form subwindow under the name specified in the File field.

Get! Retrieves the AR record filed under the name specified in the File: field.

Max QL Sets the maximum length of a query list for use with the Tool Driver (see section 3.7). It sets the size of the query list to the value that is currently selected. This command should only be used by persons who are familiar with Tool Driver. If a query list that is too large is used with Tool Driver to edit AR records, the database can be damaged.

3.3 Query

The Adobe Query tool is used to generate lists of AR records that have some common characteristic(s). Such a list is called a *query list*. Producing a query list involves using Adobe Query to search the database for AR records that match specified search criteria.

Query lists are database-specific; that is, you may not create a query list that includes more than one database. However, there may be many query lists for one database, each with a different combination of search criteria.

A query list can only be accessed by tools set for the database for which it was made. For example, a query list created for the database system Personnel Records can only be accessed by other Adobe tools set to Personnel Records.

Query lists remain in existence as long as at least one Adobe tool stays active or inactive. If all Adobe Tools are destroyed, the query lists will then be destroyed. However, query lists can be saved in files and later restored to a system's list of query lists. See section 3.4 for details.

Querying uses data maintained by the Adobe services. This data is updated automatically whenever a **Submit!** or **Checkin!** occurs. Therefore, the Adobe services' data is always up to date. However, query lists are only current as of the date they are created. Therefore, if a database has been updated or edited since the last query list was created, a new query list should be made.

3.3.1 Query window

Figure 3.3 illustrates the Query window using a system for recording software problems called System Software Action Requests.

3.3.2 Querying a database system

A database can be queried using the following procedure:

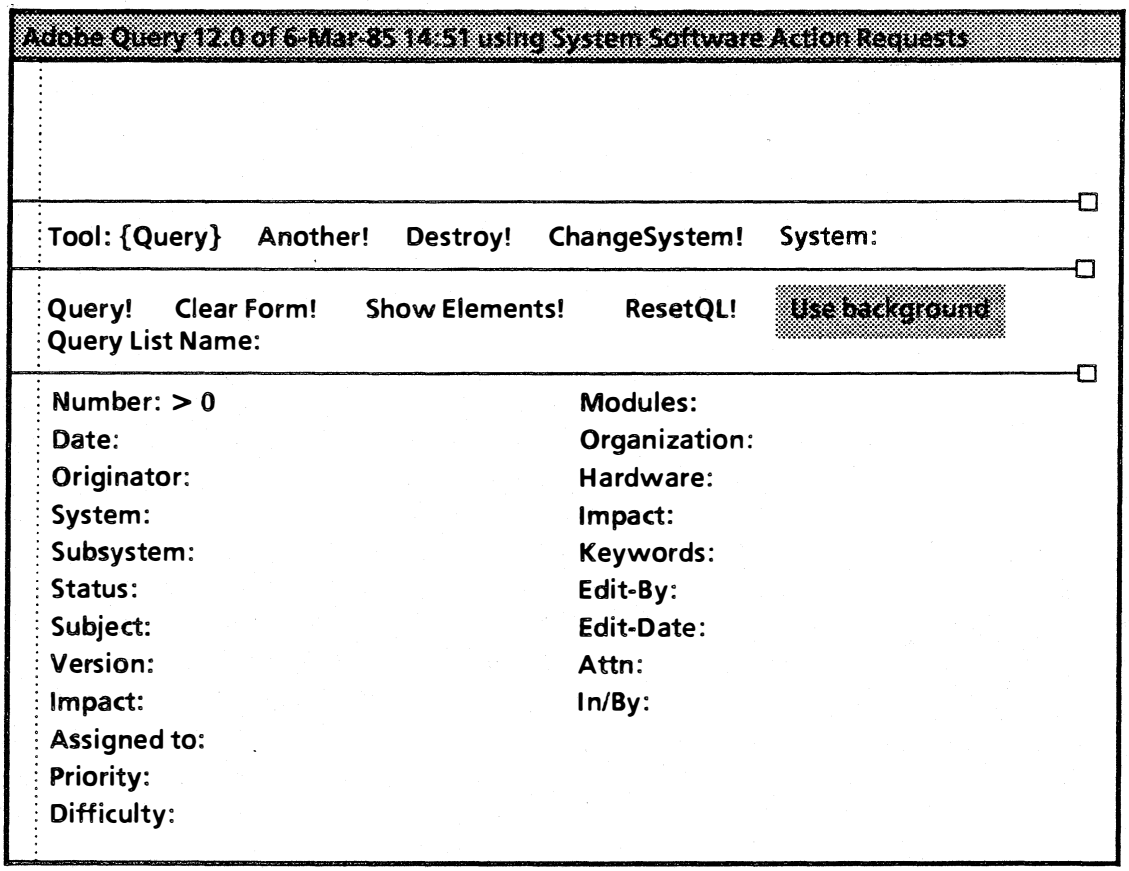

Figure 3.3 Adobe Query window

- Bring up an Adobe tool window.
- Set the **Tool** field in the Adobe command subwindow to **Query**.
- In the Adobe command subwindow, chord anywhere in the Adobe tool window and select the database you want to use from the Adobe System pop-up menu. Alternatively, you may enter the name of the system you want in the **System** field and select **ChangeSystem!**
- Enter a name for the query list in the **Query List Name** field. If no name is specified, the default name **SysQL** will be used.
- In the form subwindow, use the pop-up menus for each field you want to search to get a list of legal operators for that field. After selecting an operator, add your search expression. A complete explanation of operators and search expressions will be given in section 3.3.4.
- Select the **Query!** command when you have finished entering your search criteria.
- Select **Clear Form!** to clear all fields in the form subwindow and prepare for another query list.

A detailed explanation of Query commands follows.

3.3.3 Query command subwindow

The commands in the Query command subwindow are as follows:

Query! Starts the query based on the search expressions in the form subwindow. If a search expression is invalid, an error message will appear in the message subwindow. You will have to correct the form before the query can proceed.

Clear Form! Empties all the fields in the Query form subwindow and resets any defaults.

Show Elements! Displays the query list in the system default window, usually the herald window.

ResetQL! Resets the query list named in the **Query List Name** field to an empty list.

Use background Allows you to process commands in more than one Adobe window simultaneously, using a background process. When **Use background** is highlighted, as in Figure 3.3, the background process is on. You can turn the background process off by clicking on **Use background**. When you want to do more than one operation at a time, **Use background** should be on (highlighted).

Query List Name: Specifies the name of the query list. If this field is left blank, the default query list **SysQL** is used. If the name of an already existing query list is given, the old list will be replaced by the new one the current query generates.

3.3.4 Query form subwindow

The characteristics of the AR records you want included in your query list are specified by filling in the applicable fields in the Query form subwindow. You may have only one search field, or you may have many.

An example of a query list with only one search field is a list of all employees (in Personnel Records) who have been hired since January 1, 1984.

An example of a query list with more than one search field is a list of all employees who have been hired since January 1, 1984, whose job status is still Active, and whose Grade is more than 3. When you have more than one search field, the query list will include *only* those AR records that satisfy *every* search expression you have chosen.

The kinds of queries you can make on a database vary with field type. If you are not familiar with the types of fields in Adobe database systems, and you have not read **Field Types** in Chapter 1, go back and do so now.

Each field in the Query form subwindow has a menu associated with it. The menu is to assist you in creating search expressions.

It contains the legal operators for that field. You do not have to memorize the operators for each type of field, but you do need to know what they mean when you see them on the menu so that you can create your search expressions. Below is a list of the operators and their meanings.

= equal to

not equal to

= and # are case sensitive; that is, **= microcode** will not result in selection of items written as *Microcode*. If you are searching for alphabetic strings and you suspect that capitalization may be inconsistent in the database system you are searching, it may be better to use **HAS** and **~HAS** (see below).

> greater than

< less than

> = greater than or equal to

< = less than or equal to

OR or

AND and

NIL empty value string. Selecting **NIL** will cause Adobe to search for fields with no values entered in them.

HAS includes the string that follows **HAS** in the search expression

~HAS does *not* have the string that follows **~HAS** in the search expression

HAS and **~HAS** are case insensitive; that is, **HAS microcode** will result in selection of strings writen *microcode*, *Microcode*, or any other combination of upper- and lowercase letters, and longer strings that include the word *microcode*, such as *microcoder*.

HAS and **~HAS** allow you to search for several expressions, using partial strings. For example, **HAS program** will retrieve *programmer*, *programming*, *program*, or *programmable* in the specified search field.

Selecting an operator from the pop-up menu for a particular field causes the operator to be entered in the field. You need only add the expression you want operated upon. For example, the pop-up menu for **DateTime** fields includes the operators **=**, **#**, **>**, **<**, **> =**, **< =**,**OR**, **AND**, **NIL**. By selecting **>** with the mouse and adding 1-Mar-79, you create an expression that will search for any dates later than March 1, 1979. Obviously it is just as easy to type in the **>** sign. The value of using the menu is that it displays the allowable options and prevents you from making mistakes if you are not familiar with the legal operators for each type of field.

The allowable operators for each type of field are as follows:

Numeric **=, #, >, <, > =, < =, OR, AND, NIL.** If no operator is specified, **=** is assumed. Only numbers may be entered as a search expression for numeric fields.

Bounded string **=, #, HAS, ~HAS, OR, AND, NIL.** The search expression may be any alphanumeric string.

Enumerated If this is an independent enumerated field, the menu will include **#, OR, AND, NIL,** and the possible set of values for the enumerated field.

If it is a dependent field, a set of possible values will be shown only if the query expression in the related independent enumerated field is a single value. In dependent enumerated fields you are not limited to what is shown in the menu as querying values.

DateTime **=, #, >, <, > =, < =, OR, AND, NIL.** If you specify a date without a time, the time defaults to 00:00:00 (midnight).

Search strings must be written without spaces or tabs. If you want to search for strings that include spaces, they must be enclosed in quotes, but in that case the string itself may not include quotes.

Suppose that you want to see all the AR records for software problem reports in the database system illustrated above that have these characteristics: that they are bugs in the Diagnostics system which have serious or fatal impact and which are not yet fixed, for any reports dated January 1, 1985 or later, and which have been rated as hard or very hard to solve. You would fill in the fields in the Query form subwindow as follows:

Number: >0

Date: > = 1-Jan-85

System: Diagnostics (from the pop-up menu)

Status: # complete (from the pop-up menu)

Impact: Serious OR Fatal (from the pop-up menu)

Difficulty: Hard OR Very hard (from the pop-up menu).

3.4 Query List

The Adobe Query List tool is used to perform a variety of operations on the query lists created with Adobe Query.

Adobe *Query* searches a database system for field items that satisfy your search criteria and makes a list of their AR record numbers; these are your original query lists. Adobe *Query List* enables you to manipulate your original query lists in a variety of ways to create new lists.

For example, suppose you need a list of all the employees who are retired or whose hire date was before January 1, 1960. You cannot get this information with a single query from the Query

tool. However, you can create one query list for Current Status: = retired, and another for Hire Date: = < 1-Jan-60, and use Query List's **Union!** command to combine the two for a new list that will include everyone who satisfies either or both criteria.

Remember that a query list reflects the status of the database at the time it was created. If the database system you are using has been updated since the query list was made, the query list will not be accurate and you will need to create a new one.

3.4.1 Query list window

Figure 3.4 illustrates the Query List tool using a system called Personnel Records to store employee information.

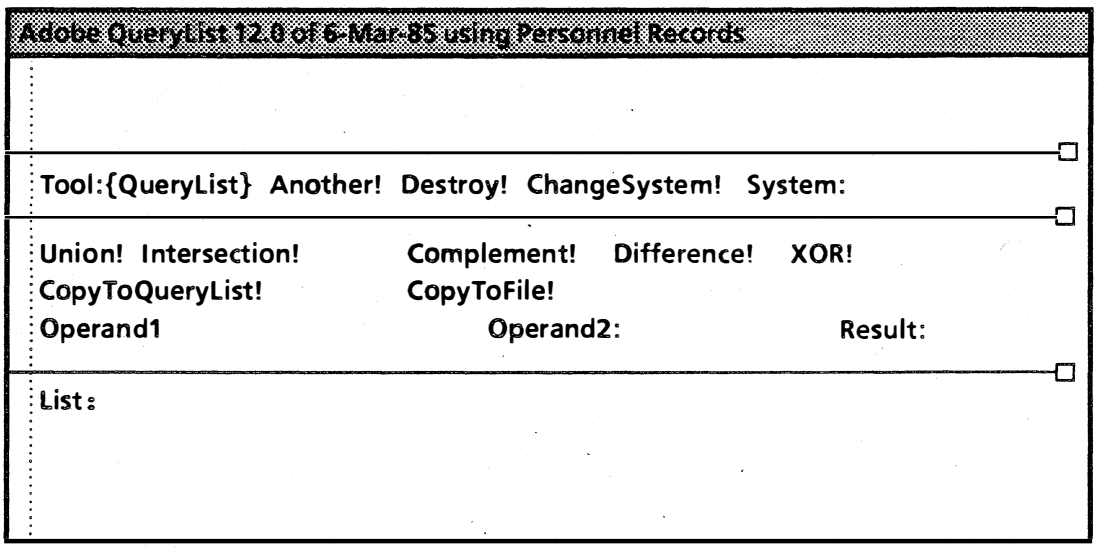

Figure 3.4 Adobe QueryList window

3.4.2 Query list operations

Query lists can be used by the following procedure:

- Bring up an Adobe Tool window.
- Set the **Tool** field in the Adobe command subwindow to **QueryList**.
- In the Adobe command subwindow, chord anywhere in the Adobe tool window and select the database you want to use from the Adobe System pop-up menu. Alternatively, you can type the name of the system you want in the **System** field and select **ChangeSystem!**

For single query list operations:

- Enter the name of the query list you want to operate upon in the **Operand1** field.

- Enter the name you want to use for the resulting list in the **Result** field.

- If you want a list of all the AR record numbers in the database that are **not** in the list named as **Operand1**, select **Complement!**

  For example, suppose you already have a query list for all the names of employees who are programmers and the name of the query list file is *Programmers.ql*. To get a list of all employees who are not programmers, simply enter *Programmers.ql* in the **Operand1** field, type a query list name such as *Nonprogrammers.ql* in the **Result** field, and select **Complement!**

- If you want to convert a query list to a text file that can be printed out or displayed on the screen, select **CopyToFile!** Enter the query list name in the **Operand1** field; the **Result** file will be your text file.

- To convert a text file back to a query list file, enter the name of the text file in the **Operand1** field and select **CopyToQueryList!** The **Result** field will be your query list. This command can also be used to copy one query list into another query list.

For two query list operations:

- Enter the name of one query list in the **Operand1** field.

- Enter the name of the second query list in the **Operand2** field.

- Enter the name you want to use for the resulting list in the **Result** field.

- If you want to join the two lists, select **Union!**

  For example, you may have a query list of all the employees who have worked on a project called SuperSpreadsheet, and another query list of all employees who have worked on a project called InfallibleFile. To get a list of all employees who have worked on either or both of the projects, enter one list in **Operand1** and the other in **Operand2** and select **Union!** Query List will join the two lists; the resulting list will contain all the names on both the original lists. Some of the employees will have worked only one of the two projects, some may have worked on both.

- If you want a list of AR numbers that are in either **Operand1** or **Operand2** but *not* in both, select **XOR!**

  In the above example, if you want a list of employees who have worked either on SuperSpreadsheet or on InfallibleFile, but not on both, you would select **XOR!**

- If you want a list that contains only the AR record numbers that are common to both **Operand1** and **Operand2**, select **Intersection!**

  In the example above, selecting **Intersection!** instead of **Union!** would result in a list of only those employees who have worked on both projects.

  For another example, suppose you already have one query list for all employees who work in Building 3, and another for all employees whose job title is *programmer*. Enter one query list in **Operand1** and the other in **Operand2**, choose a name for your **Result** field, and select **Intersection!** Adobe Query List will search for all records that are common to both query lists. The result will be a third list that contains AR record numbers of all employees who are programmers and who work in Building 3.

  In this case it does not matter which original query list is in **Operand1** and which is in **Operand2**

- If you want a list of AR numbers that are in **Operand1** and are *not* in **Operand2**, select **Difference!**

  Using the same original query lists as in the example above, enter the query list containing all employees who are programmers in **Operand1**, and the query list containing all employees that work in Building 3 in **Operand2**. Selecting **Difference!** will result in a list of employees who are programmers but who do not work in Building 3.

  Note that in this case, reversing the order in which you enter the lists in **Operand1** and **Operand2** gives a different result. Entering the list of all employees who work in Building 3 in **Operand1** and all employees who are programmers in **Operand2** and selecting **Difference!** will result in a list of all employees who work in Building 3 who are not programmers.

An explanation of Query List commands follows.

3.4.3 Query list command subwindow

Operand1: Specifies the name of a query list to be operated upon. A query list may be selected from the menu associated with this field. Chord over **Operand1** to see the menu.

Operand2: Specifies the name of a second query list to be operated upon, if any. A query list may be selected from the menu associated with this field. Chord over **Operand2** to see the menu.

Result: Specifies the name of the new query list resulting from the operations performed on **Operand1** and **Operand2**. A query list name may be selected from the menu associated with this field. Chord over **Result** to see the menu.

Union! Combines the query lists named in **Operand1** and **Operand2** to produce a third query list containing all the AR record numbers of both files.

Intersection! Produces a list of only those AR record numbers common to both **Operand1** and **Operand2**.

Complement! Produces a list of AR numbers not in **Operand1** for the database system you are using.

Difference! Produces a list of AR numbers in **Operand1** that are not in **Operand2**.

XOR! Produces a list of AR numbers that appear in either **Operand1** or **Operand2** but not in both.

CopyToFile! Copies the query list named in **Operand1** to the file named in **Result**. You should always save your query list to such a file if you intend to destroy all your Adobe windows but wish to save the query list for future use. Query lists are lost when all Adobe tool windows are destroyed.

CopyToQueryList! Copies the file named in **Operand1** to the query list named in **Result**. This is how you restore a query list previously stored to a file

3.4.4 Query list form subwindow

The Adobe Query List form subwindow (illustrated in Figure 3.4) contains only one form item, **List**. This is a field item where you may specify AR numbers or ranges of AR numbers that you want Query List to operate on.

Numbers should be kept in ascending order; runs of AR numbers should be represented with a hyphen, as in 2340-2487; and spaces should separate numbers and runs. An example entry would look like this:

List: 2310 2324 2483-2597 3096

If List is used as **Operand1** or **Operand2**, Adobe will use the numbers specified in List as a query list and perform the selected operation.

If **List** is used as **Result**, Adobe will perform the operation selected on **Operand1** (and **Operand2**, if any) and display the resulting list of AR record numbers in the form subwindow.

3.5 Report

The Adobe Report tool is used to produce a formatted report based on a query list. The report is stored in a text file. Report also produces a separate file of sort *keys*, used by Adobe Sort to sort the report file. Figure 3.5 illustrates the Report tool using a system called Personnel Records.

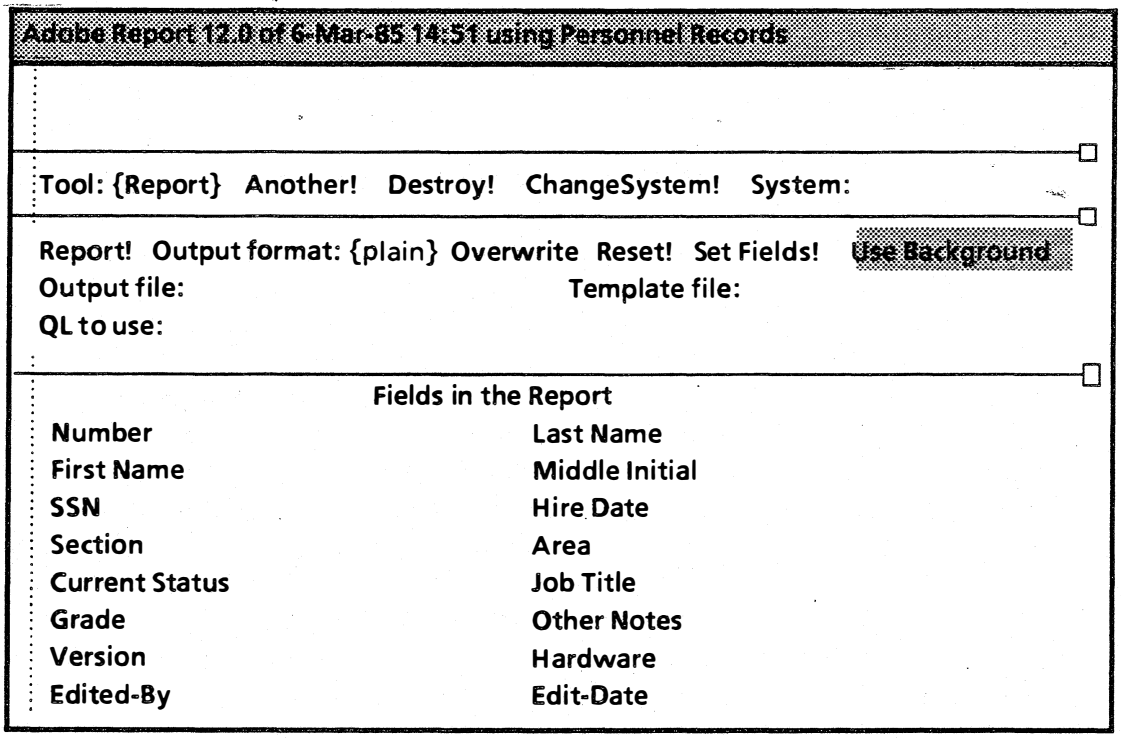

Figure 3.5 Adobe Report window

3.5.1 Generating a report

A report can be generated as follows:

- Bring up an Adobe tool window.
- Set the **Tool** field in the Adobe command subwindow to **Report**.
- Select the desired format (plain, columns, or template) for the report from the Output format pop-up menu.
- Select the query list you wish to use from the QL to use menu or enter the name of the query list in the **QL to use** field.
- Specify the fields to be included in the report as follows:

  If the **Output format** is *plain* or *columns*, specify the fields to be included in the report by selecting the desired fields in the Report form subwindow. You may select as many fields as you wish.

  If the **Output format** is *template*, enter the name of the template file you wish to use in the **Template file** field, then select **Set Fields;**
- Enter the name of the report in the **Output file** field. If a file by that name already exists and you wish to overwrite it, select **Overwrite**. If you do not select **Overwrite**, and the **Output file** name is the same as an existing file, you will get

an error message. This is to prevent your accidentally overwriting a file you wish to save.

- Select the **Report!** command to start the report.
- Select **Reset!** to clear the selected fields in the form subwindow before you run a new report.

A detailed explanation of the Report commands follows.

3.5.2 Report command subwindow

The commands and fields in the Report command subwindow are as follows:

Output format: Specifies the format for a report. This is an enumerated field that allows you to select the report format from a pop-up menu. Chord over **Output format** and select *plain, columns*, or *template*. For a complete explanation of these choices, see section 3.5.4.

Template file: Specifies the name of a text file containing a template for formatting the report. See section 3.5.5. for a complete explanation of template files.

Set Fields! Selects the fields that are specified in the template file. It is not used for any other format options. The **Template file** field must be filled in with a valid template file name before you can select **Set Fields!**

QL to use: Specifies the name of the query list on which the report is to be based. You may select the query list you want by chording over **QL to use** and selecting a name from the pop-up menu, or you may type in the name yourself. If no query list is specified, the default **SysQL** is used.

Output file: Specifies the name of the file in which the report will be stored.

Overwrite If **Overwrite** is selected, the file named in the **Output File** field will overwrite an existing file with the same name. If it is not selected, and the **Output file** name matches a file name on the disk, the report will not be generated.

Report! Creates a report on the query list named in the **QL to use** field. The report will show the fields specified in the Report form subwindow.

Reset! Clears the selected fields in the form subwindow.

Use background Allows you to process commands in more than one Adobe window simultaneously, using a background process. When **Use background** is highlighted, as in Figure 3.5, the background process is on. You can turn the background process off by clicking on **Use background**. When you want to do more than one operation at a time, **Use background** should be on (highlighted).

3.5.3 Report form subwindow

All the fields for AR records in the selected query list are displayed in the Report form subwindow. In Adobe Report the fields in the form subwindow are all on/off fields; only those fields that you select with the mouse will appear in the report.

Some of the fields in the selected database system may be unbounded string fields. If you select one of these fields for the report you will get a message saying "A report with unbounded string cannot use accelerator files." Any report including unbounded string fields will take longer to complete. Also, Adobe Sort cannot sort on unbounded string fields.

3.5.4 Output format

Every report has a header that contains the name of the file on which the report is based and the time and date that the report was started.

Plain format If you choose *plain* format from the **Output format** field menu, the report will be formatted this way:

Number: 33
Last Name: Johnson
First Name: Steve
Current Status: Active
Job Title: Associate Programmer
Grade: 6

Columns If you choose *columns* from the **Output form** field menu, the report will be formatted this way:

Number	Current Status	First Name	Last Name	Grade	Job Title
33	Active	Steve	Johnson	6	Associate Programmer

The combined width of all the columns for a particular report should be calculated carefully so that the total width does not exceed the width of the paper on which the report is to be printed. Column width and the order in which fields are printed are defined in the **[Report]** section of the **.user** file for the database system.you are reporting on. You may change these to suit your needs (see Chapter 4, section 4.2).

Templates If you choose to use a *template* file, your report will be formatted according to a design contained in the template file. Template files are useful when you want more than one field printed on a line but do not want a column format. An example of the kind of format possible with a template:

AR#:33 **Name:**Steve Johnson **Soc. Sec. Number:** 555160522
Grade: 6 **Job Title:** Associate Programmer
Current Status: Active

Setting up template files is explained in detail in the next section.

3.5.5 Template files

A template file is a text file that contains a report format for a particular database system or query list. It lists the fields that are to be included and the order in which they are to appear. The template defines the amount of space to be used for each field. Using a template allows you to insert text in the report that may not appear in the database.

Template files contain text and a few special symbols: #, @, and <>. The template file for the example shown under **Templates** above looks like this:

```
<Number>AR'#:### Name:<First Name>@ <Last Name>@ Soc. Sec. Number:<SSN>#########
Grade: ## Job Title:@
Current Status: @
```

Following is an explanation of the example template file. The special symbols will be explained in context.

<Number>AR'#:###

Number is the name of the field whose value is to be printed. But in the report, you want the value to be identified as **AR#** instead of **Number**. Enclosing the field name **Number** in less-than (<) and greater-than (>) signs will prevent the field name from being printed.

AR'#: is the text to be printed instead of the field name. However, the ' will not print. The number sign (#) is used by the template file to reserve a character space for a letter or number (when used in this way it is called a *substring definition character*). Therefore, when a number sign is used as a part of text to be printed or in a field name, as in this case, it must be preceded by an ' to tell Report it is a character to be printed, not a substring definition character. The other special symbols (**@, <>**) must also be preceded by an ' if they are part of text to be printed or part of a field name.

are not preceded by apostrophes, so they are substring definition characters. They reserve three spaces for the number of the record being printed.

In the example given above, **<Number>AR'#:###** results in the following printed output: **AR#:**33.

Name:<First Name> @<Last Name>@

Name: is text to be printed.

<First Name> indicates that the value of the field **First Name** is to be printed here, but not the field name itself.

<Last Name> indicates that the value of the field **Last Name** is to be printed here, but not the field name itself.

In the example given above, **Name:** *<First Name> <Last Name>* results in the following output: **Name:** *Steve Johnson*. Be sure to include spaces where you want spaces in the report.

Soc. Sec. Number: <SSN> ######### **Soc. Sec. Number:** is the text to be printed.

<SSN> indicates that the value of the field SSN is to be printed here, but not the field name itself.

######### are substring definition characters that reserve nine spaces for the Social Security number.

In the example given above, **Soc. Sec. Number: <SSN>** *#########* results in the following output: **Soc. Sec. Number:** *555160522*.

Grade: ## **Grade:** is the text to be printed; it is also the name of the field whose value is to be printed, so no field name in **< >** signs is needed.

reserve two spaces for the grade number.

In the example given above, **Grade:** *##* results in the following printed output: **Grade:** *6*.

Note that although two spaces are reserved for the Grade number, only one number is printed. When more spaces are reserved than are needed, the excess spaces print as blanks. However, if a grade contains three numbers and only two spaces are reserved, the last number will be cut off.

Job Title: @ **Job Title:** is the text to be printed; it is also the name of the field whose value is to be printed, so no field name in **< >** signs is needed.

@ is a variable-width substring definition character. Like the #, it reserves space in the report for a value of a given field. However, the **@** indicates that the entire value for the field is to be shown, regardless of length. Thus, Steve Johnson's Job Title may be Pilot, or it may be Executive Director of Corporate Confusion. Either title will be shown, with no extra spaces.

In the example given above, **Job Title:** @ results in the following output: **Job Title:** *Associate Programmer*.

Current Status: @ **Current Status:** is the text to be printed. Again, it is also the name of the field whose value is to be printed, so no field name in **< >** signs is needed.

@ again indicates that a variable-length field value may be shown.

In the example given above, **Current Status:** @ results in the following output: **Current Status:** *Active*.

Remember that when #, **@**, **<**, or **>** appear as part of a field name or text to be printed in the report, they must be preceded by an '.

3.6 Sort

Adobe Sort is used to sort records in a report produced by Adobe Report. The records may be sorted either numerically or alphabetically, and for any field(s) you specify.

Figure 3.6 illustratesthe Sort window using a system called Hardware Action Requests. Sort does not have a form subwindow.

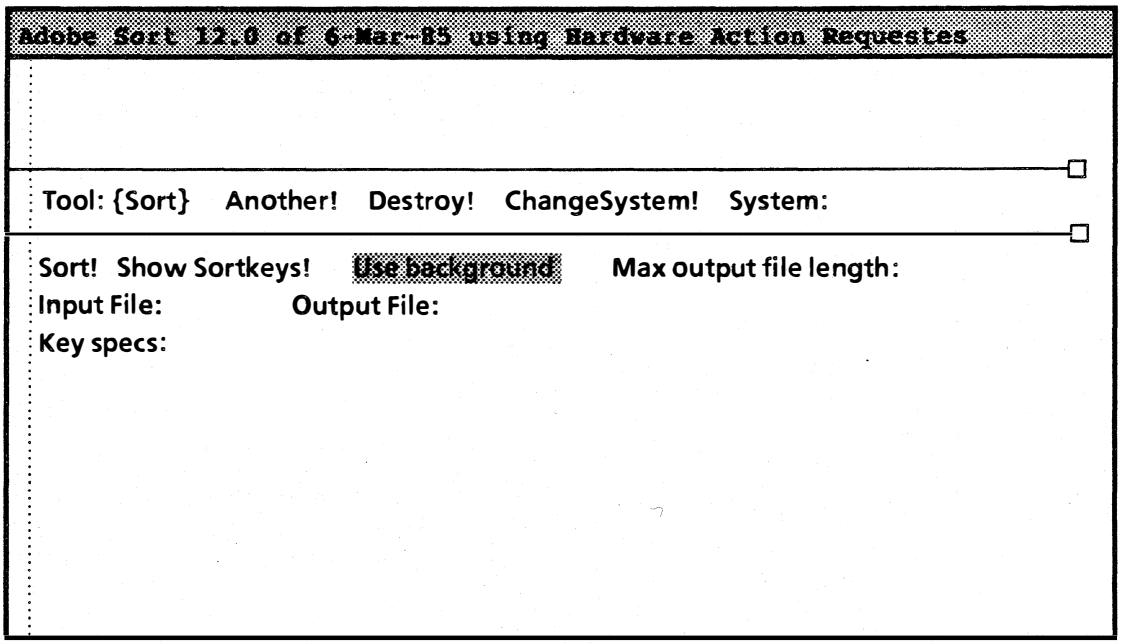

Figure 3.6 Adobe Sort window

3.6.1 Sorting a report

A report may be sorted using the following procedure:

- Bring up an Adobe tool window.
- Set the **Tool** field in the Adobe command subwindow to Sort.
- Enter the name of the file to be sorted after **Input File**
- If you wish to be reminded of the fields available to sort on, select **Show Sortkeys!** in the Sort command subwindow.
- Enter the sort specifications after **Key specs.**
- Set **Max output file length:**, if applicable.
- Enter the output file name after **Output File.**
- Select **Sort!** to start the sorting process.

A detailed explanation of the Sort commands follows.

3.6.2 Sort command subwindow

The commands in the Sort command subwindow are as follows:

Sort! Causes the report specified in the **Input File** field to be sorted, based on the sort specifications in the **Key specs** field.

Show Sortkeys! Lists the fields that can be sorted in the message subwindow. These are limited to the fields included in the report specified in the **Input File** field, excluding unbounded string fields. If the original report only includes two fields, **Name** and **Date of Hire**, you can only sort on those two fields. It is the report file that is sorted, not the database itself.

Key specs: This is where you may list the sort specifications. A report can be sorted only by the sort keys listed in the message subwindow when you invoke **Show Sortkeys!** Select the sort keys you wish to use, and enter them in the order that the sort should occur.

Sort key field names are separated from each other by a slash and a space. A *u* or *a* after a slash indicates that the field is to be sorted *u*p, or in *a*scending order. A *d* after a slash indicates the the field is to be sorted *d*own, or in *d*escending order. A single quote character should precede a slash that is part of field's name. If *u*, *a*, or *d* is not specified, the default is *u* (ascending).

For example, if you wish to sort first by **Grade/Rank**, in descending order, then by **Section**, then **Area**, and finally by **Name**, your key specs should look like this:

Grade'/Rank/d Section/ Area/ Last Name/ First Name/ Middle Initial.

None of the original order of a report is preserved during a sort. For example: suppose you have a report with entries that contain **Number** and **Grade** fields, and it happens to be sorted by **Number** because it was created that way by Report. If you then sort by **Grade** only, the **Numbers** within each group of entries of the same grade will not be sorted. Therefore, if you want the report sorted by **Grade** and **Number**, you will have to enter the **Number** field as one of your key specs.

Use background Allows you to process commands in more than one Adobe window simultaneously, using a background process. When **Use background** is highlighted, as in Figure 3.6, the background process is on. You can turn the background process off by clicking on **Use background**. When you want to do more than one operation at a time Use background should be on (highlighted).

Max output file length: Restricting the length of sorted files may be important when you need to edit or print them using some other program. The **Max output file length** field gives the maximum length in

characters that an output file from Adobe Sort can reach. Once the maximum length is reached, a new output file is started.

The output files that result from breaking up a large file will all have the same name as the original file; however, they will each have a number appended to the file name, and the numbers will indicate the proper sequence of the files.

If this form item is left blank, then the output file length is limited only by the space available on the disk.

Input File: Specifies the name of the file to be sorted. This must be a file created by Adobe Report. The input file must not have been altered since it was produced by Report.

The input file must also have an associated **.sortkeys** file on the disk. Sort key files are created automatically by Adobe Report, and normally you will not have to be concerned with them. However, if your file will not sort properly, check to see if the **.sortkey** file has been accidentally erased or damaged.

Output File: Specifies the output file name for the sorted report. This name must be different from the input file.

3.7 Tool driver

XDE provides a Tool Driver that can be used to perform very large batch operations with Adobe Tools. The average workstation user will not need Tool Driver. However, if you need to run large batch processes, you can find instructions in Chapter 7 of the *XDE User's Guide*.

Tool Driver is capable of destroying entire databases; therefore it should not be used without adequate instruction.

4.1 Creating a new database

Adobe database systems can be created only by the Adobe Services administrator. However, if you have a unique application for an Adobe system, you may design the system and take it to the Adobe Services administrator for implementation.

To design an Adobe database you must be familiar with the kinds of fields that are used for various types of information. If you are not familiar with the fields used by Adobe and have not read the **Field types** section in Chapter 1, go back and do so now.

Make a list of all the field items you want to include in the database, including field type. For bounded string fields, include the length of the field. A list of fields that might be included in a system to record requests for hardware installation or problems might be as follows:

Name of field	Type of field	Length
Number:	ARId	
Network Address:	Bounded string	25
Requester:	Bounded string	20
Location:	Independent enumerated New York Chicago Coyote Gulch Nil	
Bldg.:	Dependent enumerated for Location	
For New York	Madison Ave., 42nd St.	
For Chicago	Windy Way, East Side	
For Coyote Gulch	Shed #1, Shed #2, Shed #3	
	Nil	
Service requested:	Independent Enumerated Repair Install Move Demo Training Other Nil	

Date of request:	DateTime SystemMust (date and time of entry)	
Assigned to:	Bounded string	20
Request description:	Unbounded string	
Action:	Bounded string	150
Status:	Independent enumerated New Open Scheduled In progress On hold Awaiting parts Closed Rejected Nil	
Priority:	Independent enumerated 1-10, Nil	
System:	Independent enumerated Daisy Margo Veronica Nil	
Subsystem:	Independent enumerated Disk drive Display Ether Floppy drive Keyboard Laser Memory Mouse Orbit Paper path Power supply Software User Xerography Other Nil	
Sched. date:	DateTime	
Actual date:	DateTime	
Repair time:	Numeric	
Resolution description:	Unbounded string	
Parts used:	Unbounded string	

In designing a database system you have the option of specifying **SystemMust** fields, such as the date and time and entry for **Date of Request** in the above example. **SystemMust** defaults cannot be overridden by the user.

4.2 Setting up .user files

Each database system has a **.user** file associated with it that specifies how fields appear in the form subwindow. The location of each field item is designated by x: for its horizontal location (in pixels), and y: for its vertical location by line number. Since the form subwindow items may vary from system to system, a separate window layout is necessary for each database system. Adobe Report requires the addition of a total character width for each entry.

If you wish, you may also create your own customized window format for any Adobe database system you are using and store it on your workstation. Adobe checks for a local **.user** file before retrieving the default system **.user** file.

Below is a sample **.user** file for the database defined above.

HardwareActionRequests.user

[Submit]
Requester: [x: 0, y: line1]
Location: [x: 0, y:line2]
Bldg.: [x: 220, y: line2]
Network Address: [x:0, y: line 3]
Service requested: [x:0, y: line4]
Date of request: [x:0, y:line5]
Request description: [x:0, y:line6]

You will note that the Submit window contains only a few of the total number of fields for the database system. These fields represent the information that is available when the action request in initially received. The Edit tool is used to add information as the request is processed. Each of the windows in the examples below contains most or all of the database field items, depending on what is needed for the respective tool function.

[Edit]
Number: [x:0, y: line 1]
Requester: [x:220, y: line 1]
Net address: [x:0, y: line 2]
Location: [x:0, y:line 3]
Bldg.: [x:220, y: line 3]
Date of request: [x:0, y:line 4]
Service requested: [x:220, y:line 4]
System: [x:0, y: line 5]
Subsystem: [x:220, y: line 5]
Request description: [x:0, y: line6]
Assigned to: [x:0, y: line 7]
Priority: [x:220, y: line 7]
Action: [x:0, y: line 8]
Status: [x:220, y: line 8]
Sched. date: [x:0, y: line 9]

Actual date: [x:220, y: line 9]
Repair time: [x:0, y: line 10]
Parts used: [x:0, y: line 11]
Resolution description: [x:0, y: line 12]

The Query window could be identical, except you would not include the **Request description** or the **Resolution description**, because you cannot query on unbounded string fields.

There is no form layout for Query List or for Sort. Sort has no form subwindow items.

The fields in the form must be specified in order, from left to right and top to bottom. For example, you cannot indicate a field at x:25 y: line 4, then follow it with a field with location x:0 y: line 3. A error message will appear if this rule is violated, and a default ordering will be used with all fields displayed.

The Report form subwindow uses a different value for x than the other tool form subwindows. In Report, x = the order in which the field appears. Report also has an additional value, w, for the width of the field. If w = 0, there are no width restrictions. A value of 0 for w is used for unbounded string fields.

Report requires an additional item to define the width of the field item. The Report form subwindow may include all of the fields for the database, including unbounded strings, but we have inlcuded only enough of them below to give you an idea of how the form is formatted for field width:

[Report]
Number: [x:1, y: line 1, w: 5]
Requester: [x:2, y:line1, w:20]
Location: [x: 3, y: line2, w:12]
Bldg.: [x:4, y: line2, w: 12]
Service requested: [x:7, y: line 3, w:8]
Date of request: [x:5, y:line5, w:17]
Assigned to: [x:6, y:line6, w: 20]
Request description: [x:8, y: line 4, w:0

The width of enumerated items is set for the longest word in the menu. The width for **DateTime** items is set for 9 characters for the date, 8 for the time, and a space between them, for a total of 18.

If a value exceeds the width allowed for the field in the **.user** file, Report will use as many lines as necessary to print the whole value.

On column reports, Report breaks at the width given for a field and wraps additional text onto succeeding lines. For example:

```
AR Col. 1 Col. 2
1  xxxxx xxxxx

2  xxxxx xxxxx
   xxx   x
```

5. PROBLEMS AND SOLUTIONS

When a command cannot be carried out, Adobe helps you identify the problem through messages in the message subwindow. Some common problems and their solutions are described in the following sections.

AR already checked out

Adobe displays this message if an AR record that you need to edit is already checked out to someone else. You must wait until the AR record is checked back in before you can check it out to edit it.

Wrong version of system description

Adobe displays this message if an Adobe system description you are using has become obsolete. You are given the option of getting the new system description immediately. Confirm that you want to retrieve the new version by clicking POINT (the left mouse button).

All entries made in the Adobe tool form subwindow will be lost when a new system description is retrieved. If you wish to save your completed form, you may abort the retrieval of the up-to-date system description by clicking ADJUST (the right mouse button). Note that you may not be able to restore the form using **Get!** after you get the new system description, if the change to the old system description was severe. However, you can load the file containing the form in a window and copy values into your tool window.

Access denied

To retrieve and store files on any particular file service used by Adobe, you must be authorized to access that service. The fact that you are authorized to access one service does not mean that you can automatically access another. If an Adobe tool tries to retrieve or store a file on a service where your login name and password have not been authorized, you will get a file transfer error message. Check with your Adobe Services administrator to get the required authorization.

If you know you are authorized to access a service and you still get file transfer error messages, there may be a Clearinghouse problem or a problem on the service itself.

Adobe is busy

If you invoke a command in a tool that is already processing another command, you will get an "Adobe is busy" message in the message subwindow. You may wait until the current command is finished (indicated by a "Done" message in the message subwindow) or you may create another tool window with the **Another!** command. **Another!** can be invoked while a command is running.

Unwanted Adobe operations

You may wish to abort an operation you have started. Any operation can be aborted by pressing STOP on your keyboard.

Major crash

If a fatal error occurs, your machine will go to the debugger. If you had an AR record checked out at the time, you have effectively lost the changes that you made to that AR record.

The AR record reverts to its normal unchecked-out status. You may restart Adobe and check out the record again.

PART II LIBRARIAN

6.1 Introduction to Librarian

The Librarian Tool is a record-keeping system that can be used to prevent simultaneous reading or writing of files. This is accomplished by the use of *libjects*, which work for computer files exactly the same way that library cards work for books. That is, when you want to check out a file to work on it, you "fill out" a libject. Librarian then checks out the file to you. If someone else tries to check out the file, Librarian tells him or her that the file is already checked out. The value of such a system in a large-scale programming environment is apparent: it discourages duplication or modification of code.

Libjects are stored on a service called the Librarian. Just as every book in a library has a checkout card, every file on your system may have a libject that tells where the file resides, when it was last read or written and by whom, and the reasons for all checkouts. Each time a file is checked in or out, the libject is updated. A user typically checks out a component before attempting to modify it and checks it in when the modification is completed.

In a large networked system there are many ways in which files may be accessed. It should be clearly understood that Librarian Tool does not put locks on files. Even if a file is checked out through Librarian Tool, it may be accessed in other ways. Librarian is a *record-keeping* system, and its success as a deterrant to the simultaneous reading or writing of files depends on all persons in the organization agreeing to access files through Librarian rather than by other means available.

6.2 Initialization

If Librarian Tool is not already installed on your workstation, retrieve **Tools>LibrarianTool.bcd** from the Release Directory.

Once Librarian Tool is retrieved, you can call up the Librarian tool window by entering **LibrarianTool** at the prompt in the Executive window. If you are working at a terminal that will not display tool windows, such as TTYTajo, you can run Librarian Tool through the Executive (see instructions in section 6.3.2). On windowed terminals you may run Librarian Tool either through the Librarian tool window or through Executive, but not both at the same time.

6.3. Using Librarian Tool

6.3.1 The Librarian tool window

Figure 6.1 shows the basic Librarian tool window. The window will show additional commands and fields, depending on which **Command** is selected in the tool form subwindow: Query, CheckInOut, or Administrative. Each of the three **Command** subwindows, with their respective commands and fields, are illustrated and explained in the following sections.

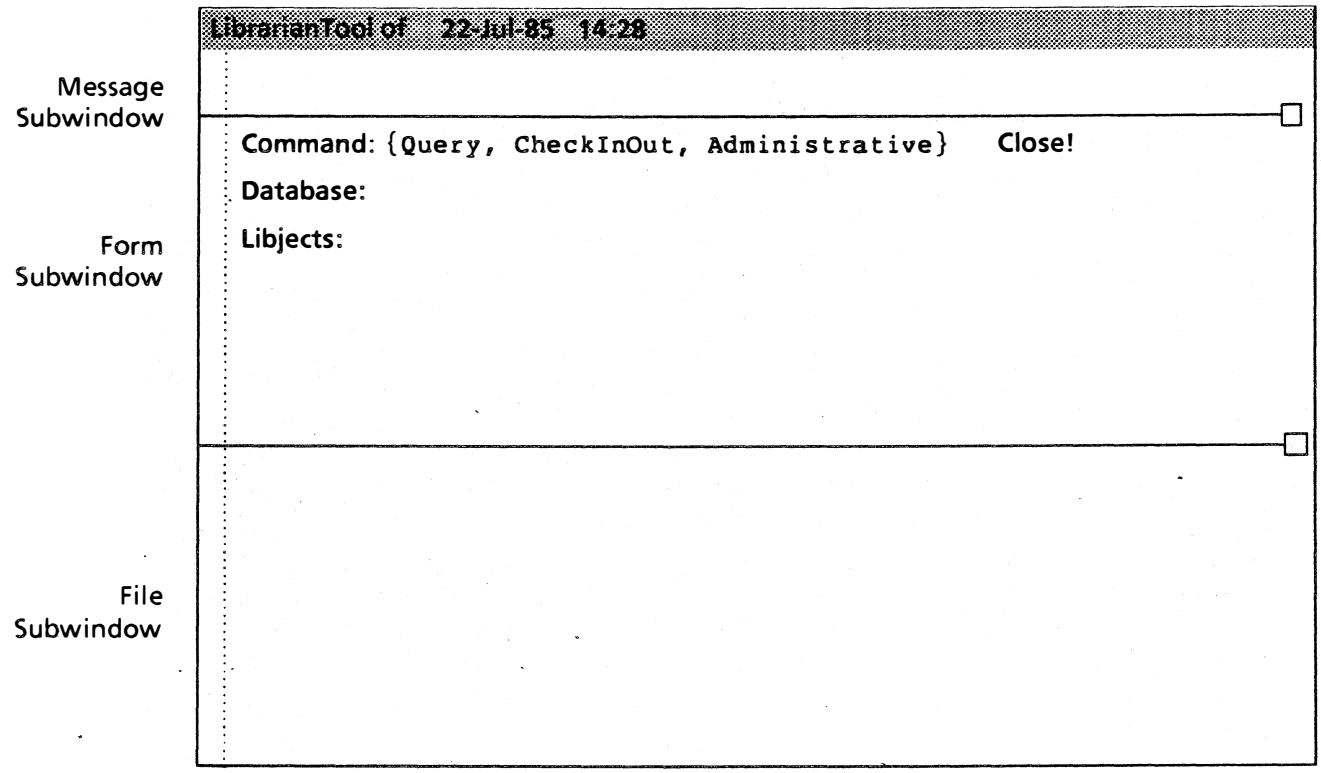

Figure 6.1: LibrarianTool window with no command selected

6.3.1.1 Librarian tool message subwindow

The Librarian Tool message subwindow displays system-level messages. Messages from Librarian Tool are displayed in the file subwindow.

6.3.1.2 Librarian tool form subwindow

The following commands and related field items (illustrated in Figure 6.1 above) are common to all the tool form subwindows.

Commands may be aborted by pressing the **STOP** key. If the results of a command may be altered by prematurely stopping it, as in the case of the **Count!** command, a warning message will appear in the file subwindow.

Command: Librarian Tool operates in three modes: **Query**, **CheckInOut**, and **Administrative**.

Selecting **Query** in the enumerated Command field enables the **Query!** and **Count!**, commands. The Query form subwindow is illustrated in Figure 6.2.

Selecting **CheckInOut** enables the **CheckIn!**, **CheckOut!**, and **Retrieve** commands. The CheckInOut form subwindow is illustrated in Figure 6.3.

Selecting **Administrative** enables the **Destroy libject!** and **Create Libject!** commands. The Administrative form subwindow is illustrated in Figure 6.4.

Database: Librarian Tool stores libjects in a database on a server. The **Database** field specifies the Librarian libject database to be used. A default database may be specified in the Librarian field of the Profile Tool, or in the Librarian section of your **User.cm** file. If no default database has been specified on your workstation, see your supervisor or the Librarian system administrator to find out what database you will be using.

Close! Closes the network connection to the database named in the **Database** field. Used when you wish to change databases. You do not have to use the **Close!** command when you deactivate the Librarian tool window; Librarian Tool will close the connection automatically.

Libjects: Specifies the libject(s) to be operated upon by Librarian tool commands. Libjects have names identical to the files they control. Therefore, the libject for a file named **Hippopotamus.doc** will also be named **Hippopotamus.doc**.

More than one libject may be entered in the Libjects field, either by entering a series of individual libject names (separated by a space), or by the use of wildcards. Wildcards are variables that allow you to search for a wide range of file names using only one search expression.

and * are wildcard symbols. # matches any single character, and * matches any multiple characters. If you want to search for all three-letter words starting with B, B## will match *bad*, *Bob*, *bib*, and so on. It will not match *bandstand* or *be*. If you want to search for all words starting with B regardless of length, use the expression B*, which will match *bandstand*, *be*, *boomerang*, *but*, *bumper*, and so on.

Either wildcard character may be used anywhere in an expression. For example, ###t will match *west*, *vast*, *belt*, etc. *t* will match *winter*, *tartness*, *onto*, etc. The symbol * by itself will match everything in the named database directory.

Wildcards are convenient when, for example, you have forgotten the name of a file you want but remember that it started with W. Listing W* in the libject field and using the **Query!** command will result in a list of all the libjects starting with W. The * and the **Query!** command will give you a list of

all libjects in the database; * and the **Count!** command will tell you how many libjects there are for a database; * and **Destroy!** will delete all existing libjects for the named database.

Wildcards are legal only for the **Query!**, **Count!**, and **Destroy!** commands.

The Query form subwindow

Selecting the Query command mode will cause the form subwindow to display the following commands and fields:

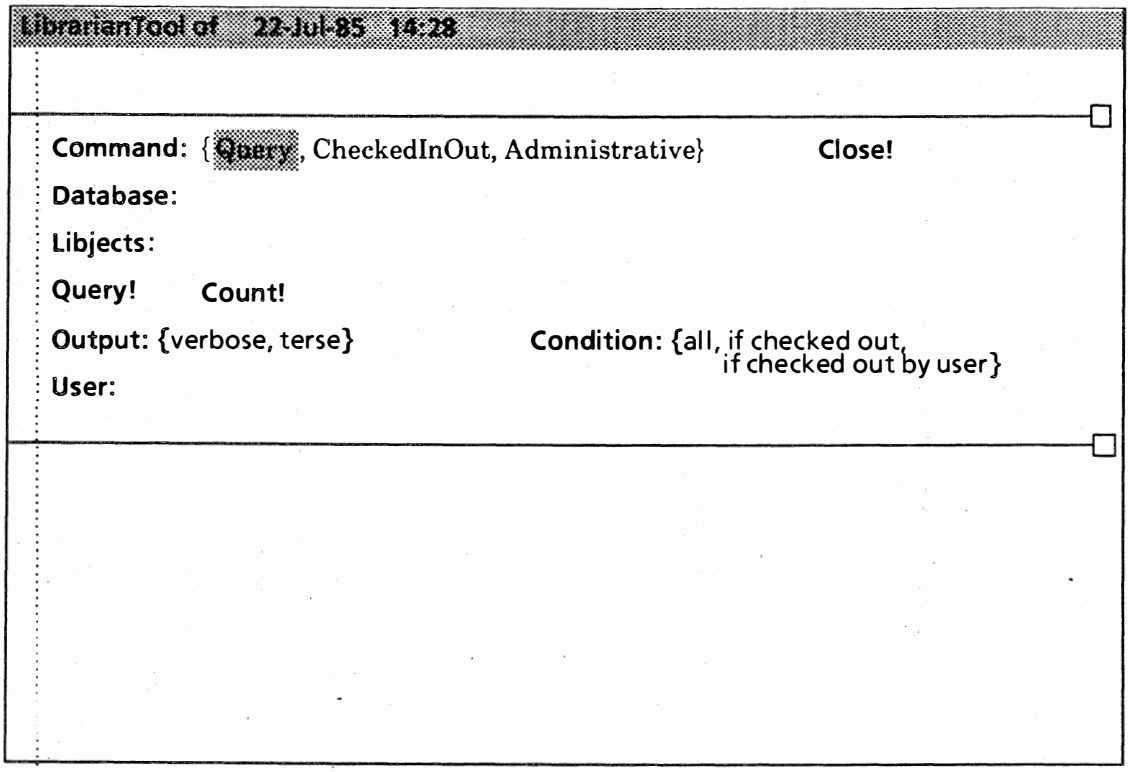

Figure 6.2: LibrarianTool window with Query command selected

Query! Queries each libject specified in the **Libjects** field and displays information on the libject(s) in the file subwindow. **Query!** will operate on libject names containing wildcards.

Query works in two modes: *verbose* and *terse*, which you may select in the enumerated **Output** field directly below **Query!** in the form subwindow

Query! in the *verbose* mode gives you all the libject information available on the file it represents. It tells you if the file is checked in or out, who has it (or who the last person was who had it), the date it was checked out (or in), the fully qualified name of the file, the date the file was created, and the reason it is (or was) checked out. The information is listed as follows, in the file subwindow:

Query of: 'Filename.ext' (checked out)
Checkout/in by: 'Francine Purcell:OSBU North:Xerox'
Checkout/in at: '2 Jul 85 16:04:53 PDT (Tuesday)
Remote file: '[Pluto:OSBU North:Xerox]<Sequoia>DF>

Windows.df!'*
Create date: *
Checkout reason: 'Bug fix'

*Note: The word 'Unknown' may appear after the remote file name and/or in the create date field. You may ignore this message; it is a field reserved for future use.

The *terse* mode is convenient for any operation in which you need only the names of the libjects. For example, using **Query!** in the terse mode, with * in the libjects field and **IfCheckedOut** selected in the **Condition** field, will result in a list of the names of all checked-out files for he named database. With **all** selected in the **Condition:** field, the same operation gets you a list of all the libjects for the named database, whether the files are checked out or not. If **if checked out by user** is selected in the **Condition** field and a fully qualified user name is entered in the **User** field, a list of files checked out to a particular person will be displayed.

A query with the wildcard Wi* in the **Libjects** field and terse mode selected might give you a report like this:

Windows.df Wisk.df WiskSupport.df
Total of 3 libjects matching the pattern: 'Wi' were found.

Output: An enumerated field with the choice of *verbose* or *terse* mode for the **Query!** command, as described above. Select the mode you want by clicking on *verbose* or *terse* The selected mode will be highlighted.

Count! Counts libjects specified in the **Libjects** field. Wildcards are legal. Libjects for files that are checked out can be counted by selecting **if checked out** in the **Condition** field. Libjects for files that are checked out to a particular user by be counted by using the wildcard * in the Libjects field, selecting **if checked out by user**, and filling in the **User** field.

Condition: An enumerated field for use with the **Query!** and **Count!** commands. When **all** is selected, the named commands will operate on all libjects in the named database. When **if checked out** is selected, the named commands will operate on checked-out items only. When **if checked out by user** is selected, the named commands will operate on libjects checked out to the user specified in the **User** field. Select the condition you want by clicking on the appropriate phrase . The selected mode will be highlighted.

User: The **User** field appears only when **if checked out by user** is selected in the **Condition** field. It is used with the **Query!** and **Count!** commands to operate only on those libjects filled out for a particular user. This field is most useful when used with wildcards in the **Libjects** field. For instance, you may use * in the **Libjects** field and your name in the **User** field to check all your own previously created libjects. You may use * in the **Libjects** field and any person's name in the **User** field to query or count libjects made out to that person. However, the user name must be the fully qualified name of the user or Librarian Tool will not recognize it.

The CheckInOut form subwindow

Selecting **CheckInOut** in the **Command** field will cause the form subwindow to display the following commands and fields:

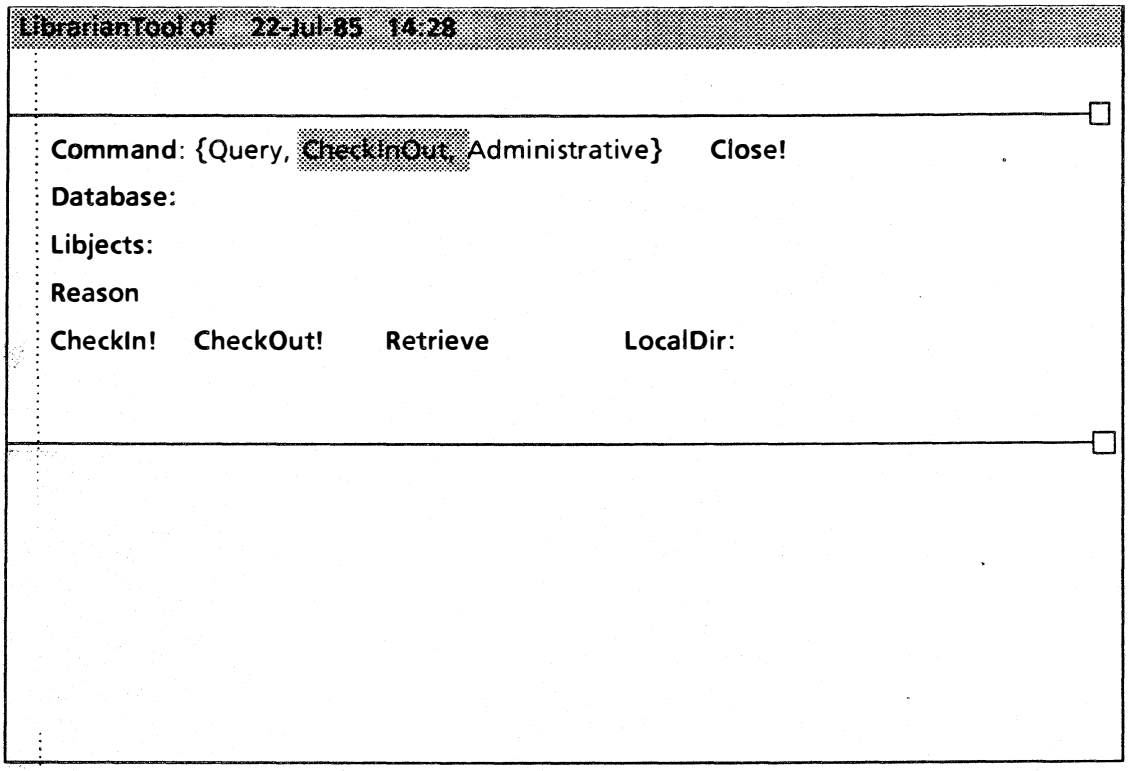

Figure 6.3: LibrarianTool window with **CheckInOut** command selected

CheckOut! Checks out each file specified by a libject in the **Libjects** field. To check out a file, you must specifiy a reason in the **Reason** field. Not doing so will result in an error message when you try to invoke **CheckOut!**

You cannot check out a file that is already checked out. If you try to do so, you will be informed by a message in the file subwindow.

You cannot check out a file for which no libject has been created. If you list a libject that does not exist in the **Libjects** field, you will be informed by a message in the file subwindow. You may then use the **Create!** command in the Administrative mode to create a libject for the file.

Retrieve When **Retrieve** is selected and the **Checkout!** command invoked, the remote file associated with the libject being checked out will be retrieved to your workstation. If no local directory is specified in the **LocalDir** field, the file will be retrieved to your default directory. **Retrieve** is highlighted when it is on. Clicking the mouse over **Retrieve** turns it alternately on and off.

LocalDir: Enter the local directory to which you wish a remote file to be retrieved. If no local directory is specified, the file will be retrieved to your default directory.

Reason: Specifies the reason for checking out a file or creating a libject. A reason is required to check out a file; a reason for creating a libject is optional. If the **Reason** field is left blank when creating a libject, Librarian Tool will supply **Initial Creation** as the reason.

CheckIn! When you have finished your work on the file you checked out, you must check it in again. Enter the libject name in the **Libjects** field and invoke **CheckIn!**. The libject must already have been checked out by the current user.

The Administrative form subwindow

Selecting **Administrative** in the **Condition** field will cause the form subwindow to display the following commands and fields:

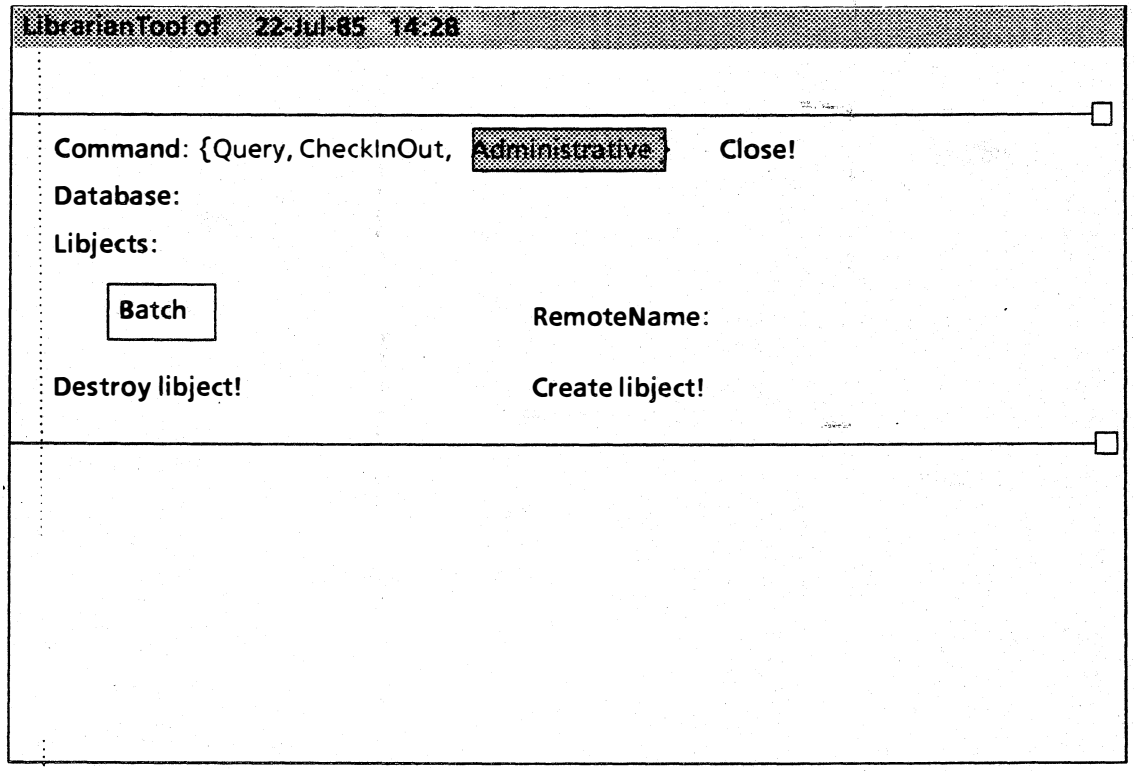

Figure 6.4: LibrarianTool window with Administrative command selected

Create! Creates the libjects specified in the **Libjects** field. You cannot create a libject with the same name as one that already exists. When the **Batch** operator is on, you may enter a series of libjects in the **Libjects** field and create multiple libjects for one remote file.

Batch An on/off operator that allows you to create multiple libjects for one remote file. **Batch** is highlighted when it is on. Clicking over **Batch** with the mouse turns it alternately on and off.

RemoteName: For use with the **Create!** command. Enter the fully qualified file name that describes where the file associated with the libject resides.

When creating a single libject specified in the **Libjects** field, this name will be used for the **Remote file** field in the libject form, as in the example under **Query!** above.

When creating several libjects using the **Batch** mode, the remote file name must be followed by the directory dilimiter '\ (or '> for backward compatibility) as follows:

RemoteName: [Server name]Directory> Filename.ext'>

The libjects will be appended to the file name by Librarian Tool. For example,if you have listed multiple libjects **A.df**, **B.df**, and **C.df** in the **Libjects** field, selected **Batch**, and invoked **Create!**, Librarian tool will create libjects **A.df**, **B.df**, and **C.df**, and fill in the Remote file field with **[Server name]<Directory>Filename.ext'>a.df**, and so on for each of the multiple libjects.

Destroy! Destroys each libject specified in the **Libjects** field. Wildcards are allowed. You can destroy all your own libjects by using * in the **Libjects** field and your name in the **User** field. Use great care in doing this, because if you forget to enter your name, all libjects in the database will be destroyed! You should never destroy libjects created by someone else.

6.3.1.3 Librarian file subwindow

The results of your Librarian tool operations will be displayed in the file subwindow, along with any messages from the tool.

6.3.2 Accessing Librarian through the Executive window

Librarian Tool can also be run in the Executive. However, the commands available are limited to **Query**, **CheckIn**, and **CheckOut**.

Wildcards may be used in Librarian tool Executive command lines, but they must be preceded by a single quote (') to prevent Executive from interpreting them as Executive syntax instead of Librarian tool syntax.

Where user names are used, they must be fully qualified. Both user names and reasons must be enclosed in quotes if they include spaces.

/l following a sring indicates a libject database. /u following a string indicates a user name. /o following a string indicates "checked out". /r following a string indicates "reason".

Multiple libjects may be operated upon by listing them just as you do in the Librarian Tool window **Libjects** field.

Query command lines:

The command to query for all checked-out libjects would be written in the following form:

Query LibrarianDatabase/l '*/o

If no Librarian database is specified, Librarian Tool will use the database specified in your Profile Tool.

The command to query for all libjects with a given extension checked out by a particular user is written in the following form:

Query LibrarianDatabase/l "UserName"/u '*.ext

For example, to search for all files in the MesaDFs database ending with the extension .df and checked out to Francine Purcell, the command is entered as follows:

Query MesaDfs/l "Francine Purcell:OSBU North:Xerox"/u '*.df

CheckOut command lines:

The command to check out a file is written in the following form:

CheckOut LibrarianDatabase/l "reason"/r libject.ext

For example, to checkout a file named BustedBeetle.df from the MesaDFs database in order to fix a bug, the command is entered as follows:

CheckOut MesaDFs/l "fix bug"/r BustedBeetle.df

CheckIn command Lines:

The command to check in a file is written in the following form:

CheckIn LibrarianDatabase/l libject.ext

To check multiple libjects in, the command is written in the following form:

CheckIn LibrarianDatabase/l $libject_1$.ext $libject_2$.ext $libject_3$.ext...$libject_n$.ext

For example:

CheckIn MesaDFs/l BustedBeetle.df CrippledCockroach.df AnxiousAnt.df

6.4 Error messages

Error messages are displayed in the file subwindow when the program is running on a Librarian tool window, and in the Executive window if the program is running from the Executive command line

6.5 References

For further information about Librarian, see the *Mesa Programmer's Manual*.

The Profile Tool section of the *XDE User's Guide* describes how to set the default Librarian database used by the tool.