NASA Contractor Report 191460

REFERENCE MANUAL FOR A REQUIREMENTS SPECIFICATION LANGUAGE (RSL), VERSION 2.0

G. L. Fisher
California Polytechnic State University
San Luis Obispo, CA

G. C. Cohen
Boeing Defense and Space Group
Seattle, WA

Contract NAS1-18586
November 1993

National Aeronautics and Space Administration

Langley Research Center
Hampton, Virginia 23681-0001

Table of Contents

1. Introduction

This manual describes tools to support RSL (Requirements Specification Language). The reader is assumed to be familiar with the basic concepts of RSL. For a complete description of the language, see the companion language reference manual. There are three RSL tools that can be used individually or in concert:

1. The Basic RSL Translator
2. The RSL Text Browser
3. The RSL Graphic Browser

The Basic Translator provides facilities similar to a programming language compiler -- syntax analysis, type checking, and interactive interpretation. The browsers provide two different views of an RSL document. The Text Browser allows users to navigate through an RSL specification using a number of menus and textual links. The Graphic Browser allows users to view supporting pictures that aid in the understanding of the text.

The tools are built to execute under the UNIX operating system. The Basic Translator can be executed on a standard terminal. The Browsers require the X Windows System.

The remaining sections of the manual describe the details of tool use. Section 2 describes the operation of the Basic Translator, including conversational interpretation. Section 3 describes the Text Browser and Section 4 the Graphic Browser. Section 5 explains how to construct "browseable" RSL documents by using the browsers in edit mode. Sections 6 and 7 describe details of text editing and graphics editing, respectively. Section 8 explains the details of the RSL dataflow editor. Section 9 summarizes the tool execution environment.

2. Basic Translator

The RSL translator accepts input in the form of RSL source files and outputs messages regarding the outcome of the translation. The translator performs syntactic analysis, type checking, and interactive interpretation.

There are two means to invoke the translator: from the UNIX command line or from the file selection menu within the browsers. Command line invocation is described here; invocation from inside the browsers is described in later sections of the manual.

Translator invocation from the UNIX shell has the following format:

```
rsl [options] file ...
```

Each file should be one of the following:

Filename Extension	File Contents
.rsl	RSL requirements specification
.rsi	companion rsl implementation module(s)
.rsg	companion rsl graphics module(s)

The .rsl files contain the formal document text that defines the base specification. The .rsi files contain optional implementation modules, as described in Section 9 of the RSL Language Reference Manual. The .rsg files contain graphical views of the specification. These views are created with the RSL graphic editor, discussed in Section 5 of this manual. If no .rsi or .rsg files are listed explicitly on the command line, the translator will automatically search for them. That

is, if f.rsl is the only command line file, the translator will search for f.rsi and f.rsg during translation.

As each .rsl file is translated, messages of the following form are printed to stdout:

```
Checking Module M
  Checking object o1
  Checking object o2
   ...
  Checking operation o1
  Checking operation o2
   ...
```

The checking of objects and operations proceeds in the lexical order encounted in the input files. If one or more errors occurs, an error message is output to stderr. Each error message indicates the text file, line number, and column number where the error was detected.

The command-line options are the following:

-m Turn off the checking messages that are normally output.

-b Invoke the browsers after processing the command line input files. If any errors are detected during translation, no browsers will appear. If .rsg files were processed, then both text and graphics browsers appear. If no .rsg files were processed, then only the text browser appears.

-e Invoke the browsers in edit mode, after processing the command line input files.

-i Enter the interactive interpreter after processing the command line input files. In the present version of the system, the interpreter and browsers cannot be run simultaneously.

-oo Perform checks to ensure that a fully object-oriented specification is defined. This entails checking that each defined operation appears in the `operations` attribute of exactly one object, and that all of the operations listed in an `operations` list appear in the same module as the object in which they are listed.

-t Dump a parse tree for the input files; this option is intended primarily for system testing.

-s Dump a symbol table for the input files; this option is intended primarily for system testing.

3. Text Browser

Figure 1 depicts a typical browser configuration, with a single RSL file open for browsing. In general, a browser display consists of one or more windows. A main control window is always displayed. This window lists the objects and operations that are currently active and provides other browser control functions. In addition to the control window, zero or more text windows are displayed. These text windows contain the UNIX files in which the active entities are defined. The main control window contains the following subwindows:

- the pulldown command menu, located across the top of the control window

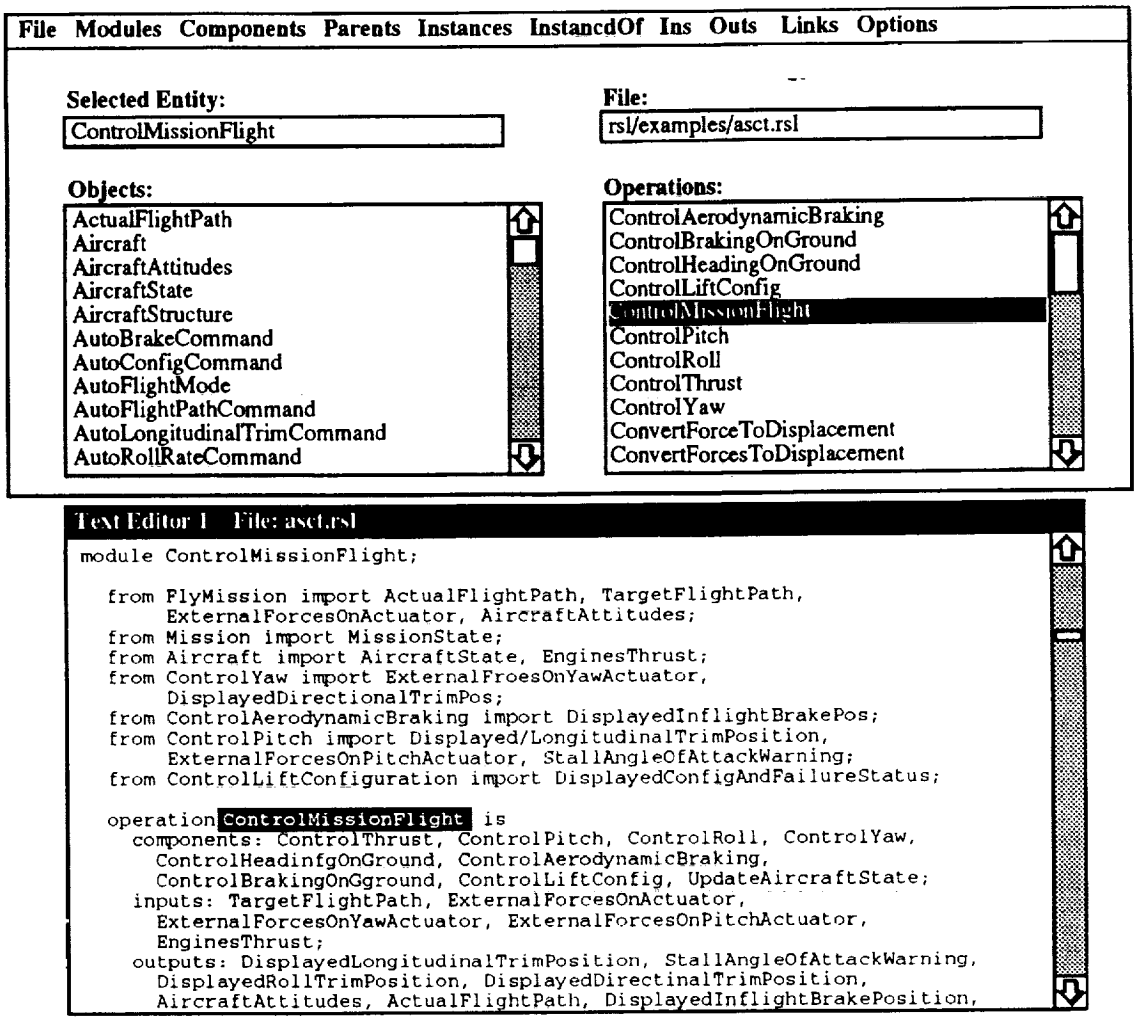

Figure 1: Typical Browser Configuration.

- the 'Selected Entity' subwindow, which lists the name of the current entity (object, operation, or module)
- the 'File' subwindow, which lists the name of the text file in which the current entity is defined
- scrollable 'Objects' and 'Operations' subwindows, which list the names of all objects and operations defined in the currently open text file(s)

Details on the manipulation and use of these subwindows follow.

The primary function of the browser is to locate object and operation definitions within RSL text files. The most direct means to locate a definition is by clicking on one of the names listed in the 'Objects' or current, which in turn leads to an update of the browser display. For example, Figure 1 depicts the state of the browser when the user has clicked on operation "ControlMissionFlight" inside the 'Operations' list.

In addition to clicking in one of the scrollable lists, entities can be located by selecting from appropriate pulldown menus, or by typing directly in the 'Selected Entity' subwindow. Details of these alternative means of selection will be described shortly.

Whenever a browsing command causes a new entity to become current, the context of the entire browser is updated accordingly. Specifically, the following display updates are made:

- The menus in the command menubar are updated appropriately.
- The name appearing in the 'Selected Entity' subwindow is changed to the current entity.
- The name appearing in the 'File' subwindow is changed to the name of the file in which the current entity definition appears.
- The 'Objects' or 'Operations' subwindow is scrolled to make the current entity visible and highlighted, if it is not already; the name of the previously selected enity, if any, is dehighlighted.
- The text window containing the current entity definition is moved to the foreground, if necessary; the text window is scrolled to the position of the current entity definition and the name text is highlighted.

In its initial state, the browser shows only three menus in the command menubar: 'File', 'Modules', and 'Options'. As browsing selections are made, new menus appear as appropriate. Specifically, new menus appear for the browseable attributes of the current entity. If the current entity is an object, the additional top-level menus are:

- Components
- Parents
- Instances
- InstanceOf
- Operations
- Links

If the current entity is an operation, all of the above menus apply, except that the 'Operations' menu is replaced by menus for 'Inputs' and 'Outputs'.

The items appearing in the pulldown menus reflect the content of the current entity's definition. For example, if the current entity is defined to have components A, B, and C, then these names appear under the 'Components' command menu. If the current entity has inputs X, Y, and Z, then these names appear under the 'Inputs' command menu. By selecting a name listed in one of the command menus, the entity of that name becomes current, and the browser display is updated accordingly.

As noted above, the 'Selected Entity' subwindow is editable. The user may type the name of an entity directly into this window. Pressing the 'Return' key within the 'Selected Entity'

subwindow causes the typed name to become current. It should be noted that the name appearing in the 'Selected Entity' subwindow must always be prefixed with its class: "object", "operation", or "module".

It is possible during browsing to select an undefined entity. This can occur if an RSL specification is incomplete or if the user types an undefined name into 'Selected Entity'. Whenever an undefined entity is selected, a small error dialog box appears to alert the user. After dismissing the alert, the context of the browser is left unchanged.

The subsections that follow describe further details of each of the top-level command menus.

3.1. File Menu

The File menu contains the following entries:

- Open ...
- New
- Close ->
- Expose ->
- Save
- Save As ...
- Save All
- Rescan
- Search Forw
- Search Back
- Goto Line
- Quit

These selections are used to view an RSL specification at the level of a UNIX text file (.rsl extension). Text files can be opened, textually searched, and displayed in multiple windows.

Selecting 'Open' produces a file chooser dialog box. The user may select a file from the chooser and the file will be included with the currently active files in the browser. Before inclusion, the file is run through the RSL translator. If successfully translated, the newly opened file is displayed in a new text window, the file becomes current, and entities in the file can be located via browser commands. If translation is unsuccessful, an error reporting window appears in which the translator error messages are listed. After unsuccessful translation, the opened file is displayed in a new text window, but browsing is disabled and the system is set to 'Edit Mode'. Within 'Edit Mode', the erroneous portions of the specification can be repaired, and the files can be rescanned, as described in Section 5.3 of the manual.

The 'Close' and 'Expose' selections both display a submenu that lists the currently open windows by the name of the file contained in each. Selecting one of the filenames under 'Close' closes the file and removes the window in which it was displayed. Selecting one of the filenames under 'Expose' causes its window to become current. This includes making the window fully visible if it is not already. Subsequent browsing commands will apply to the newly exposed window.

The three 'Save' selections and the 'Rescan' selection are active only during edit mode. Details of these menu selections are given in Section 5.3 below.

The 'Search Forw' and 'Search Back' selections perform textual search within the current text window. Both commands present a dialog box in which the string to be searched is typed by the user. The dialog box contains standard buttons to confirm or cancel the search.

The 'Goto Line' selection moves through the current text window to a specified line number. The line is centered and highlighted within the text window. The 'Quit' selection terminates execution of the browser.

3.2. Modules Menu

The 'Modules' menu lists the names of all modules defined in the RSL specification. Selecting one of the names from the 'Modules' menu causes the definition of the module to appear in the current text window. There is no enforced correspondence between an RSL module and a UNIX file. Each file can contain zero or more modules. The only constraint that exists is that a single user-defined module cannot span multiple files.

As described in the RSL language manual, an RSL specification need not contain any modules. That is, the specification can consist of a collection of object and operation definitions appearing at the top level of a file. For specifications with this structure, the system creates a module named "Main" in which the loose objects and operations implicitly appear. Loose objects and operations are those that are defined outside of any RSL module definition. If the name "Main" appears in the 'Modules' browser menu, it indicates that the RSL definition contains loose definitions.

3.3. Components and Instances Menus

The 'Components' menu lists the names of the components of the currently selected entity. These are the entities that are listed in the `components` field of the current entity. Selecting one of the names on the 'Components' list causes the selection to become the current entity.

The 'Instances' menu lists the names of instances of the selected entity. These are the entities defined as `'instance of'` the current entity. Selecting one of the names on the 'Instances' list causes the selection to become the current entity.

3.4. Parents and InstanceOf Menu

These menus list the names of hierarchical and class parents of the current entity. Hierarchical parents are those entities in which the current entity appears as a `component`. Class parents are those entities of which the current entity is declared an `instance of`. Selecting one of the parent or instance-of names causes the selection to become current.

3.5. Inputs and Outputs Menus

These menus appear when the current entity is an operation. The menus list the names defined as `inputs` and `outputs` for the current entity. Selecting one of the input or output names causes the selection to become current.

3.6. Operations Menu

This menu appears in place of the 'Inputs' and 'Outputs' menus when the current entity is an object. The 'Operations' menu lists the names defined as `operations` in the current entity. Selecting one of the operation names causes the selection to become current.

3.7. Links Menu

This menu lists the names of the user-defined relational attributes for the current entity. As defined in the language reference manual, a relational attribute defines a connection between two or more entities in an RSL specification. Consider the following example:

```
object Obj is
    components: ... ;
    attr1: obj1, obj2, obj3;
    attr2: obj3, obj4, op1, op2, op3;
    attr3: obj5, op7, obj6
end o;
```

The configuration of the 'Links' menu for this example is shown in Figure 2. The figure illustrates the two-level form of the 'Links' menu. The pull-down menu lists the names of all relational attributes. Each pull-right submenu lists the name(s) of the entity(ies) to which the relation is defined. Selecting one of the entities causes the selection to become current.

Of particular note in the links menu is the 'Pictures' submenu. This menu provides the interface between the text and graphics browsers. It lists the names defined in the `pictures` attribute of the current entity. Selecting one of the picture names causes the picture to appear in a graphic browser window. Further details of graphic browsing are defined in the next section of the manual.

3.8. Options Menu

The 'Options' menu contains the following selections:

- Edit/Browse Mode
- Transient Menus On/Off
- Auto Graphics On/Off
- Redraw

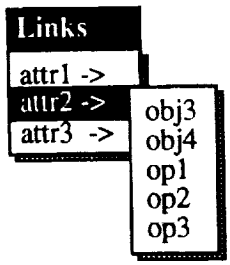

Figure 2: Sample Links Menu.

'Edit/Browse Mode' toggles between the read-only browser and the writable browser/editor. Details of editing are discussed in Section 5 of this manual. The default mode is 'Browse', unless the -e command line option is selected.

'Transient Menus' toggles the manner in which the menubar is configured. With transient menus on, only non-empty menus appear in the menubar. For example, if the current entity contains components, but no instances or parents, then the 'Parents' and 'Instances' menus will not appear in the menubar. The default for 'Transient Menus' is Off.

The 'Auto Graphics' toggle relates to the maintenance of a companion Graphic Browser, if one is currently active. If 'Auto Graphics' is On, the companion Graphic Browser is automatically updated when the current entity is changed in the Text Browser. Specifically, the preferred picture of the newly selected entity becomes the current selection in the Graphic Browser. The "preferred" picture is the name listed first in the `picture` attribute of the current entity. If no picture attribute is defined for the current entity, then the state of the Graphic Browser is not updated.

The 'Redraw' menu selection redraws the screen in the (rare) case that some damage occurs.

4. Graphic Browser

Figure 3 depicts a sample graphic browser display. This figure shows a dataflow diagram corresponding to Figure 1 above. The graphic browser provides three menus in the top-level menubar: 'File', 'Links' 'Options'. Unlike the text browser, no additional menus appear as the browser context changes.

The graphic browser is intended primarily as an adjunct to the text browser. The text browser provides the primary means to navigate through an RSL specification. The graphic browser shows supporting picture information. Picture information is presented in two formats:

- Free-form drawings
- Dataflow diagrams (DFDs)

As in the Text Browser, the Graphic Browser maintains a *current selection*. In the case of the graphics window, the selection is chosen by clicking directly on one of the displayed graphic objects in the screen.

The subsections below describe the details of the Graphic Browser menus and drawing formats. The next major section of the manual describes the creation and editing of graphics.

4.1. File Menu

The File menu contains the following entries:

- Open
- Search
- Goto Position
- Close ->
- Expose ->
- Quit

These selections are used to view pictures at the level of an RSL-graphic file (.rsg extension). Graphic files can be opened, searched by name, and displayed in multiple windows.

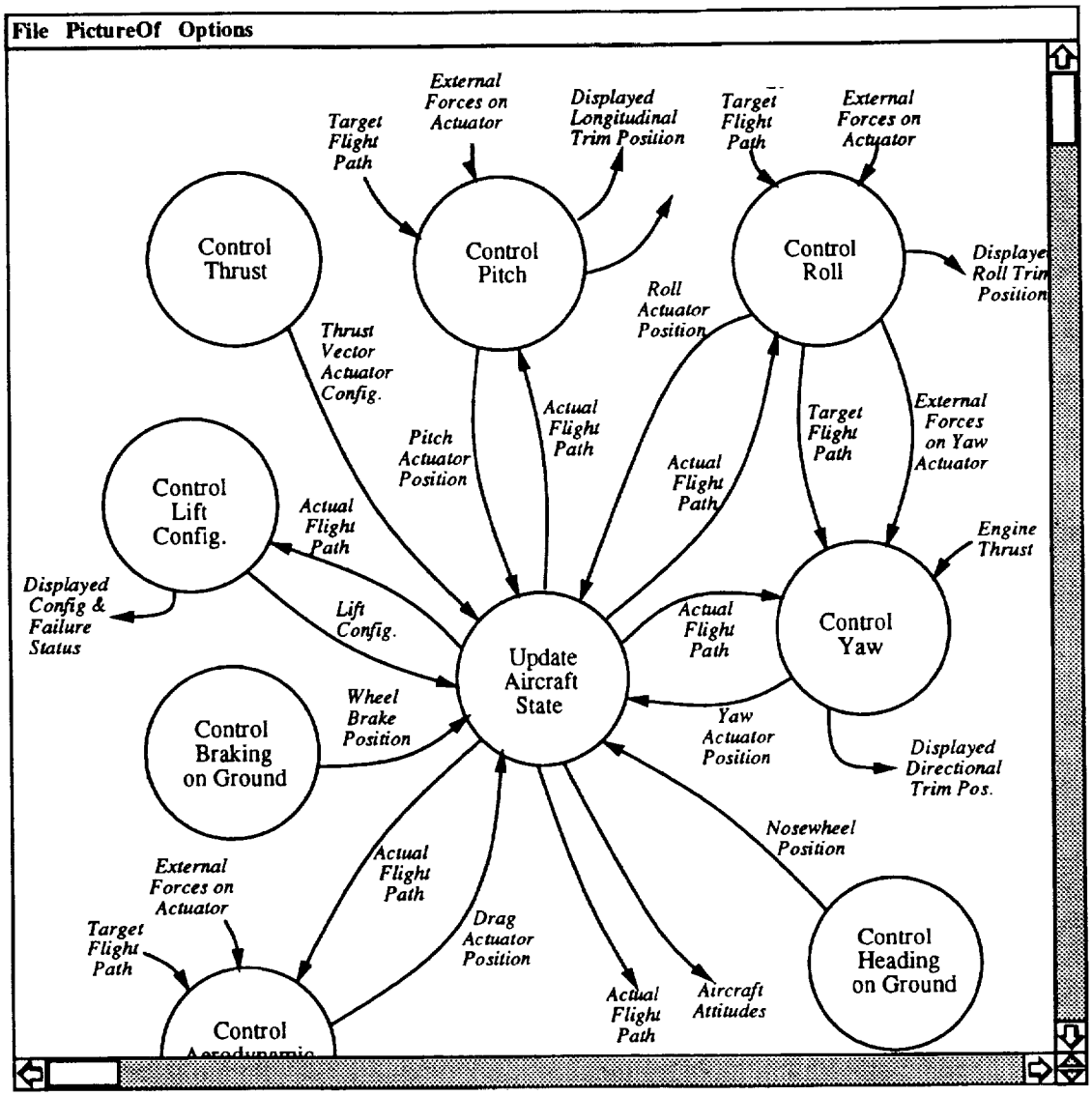

Figure 3: Sample Graphic Browser Display.

Selecting 'Open' produces a file chooser dialog box. The user may select a file from the chooser, which will be included with the currently active files in the browser. Before inclusion, the file is run through the RSL translator. If the translator detects any errors, an error reporting window appears. The error window indicates that an error in graphic translation occurred, and

the file is not included into the browser. If successfully translated, pictures in the file are display in a new graphic canvas. It also is possible to view the graphic display in a stand-alone fashion, without an accompanying Text Browser; this is done using the 'Graphics Only' option described below.

The 'Search' selection performs a search for a graphic object by its name. The selection presents a dialog box in which the string to be searched is typed by the user. The dialog box contains standard buttons to confirm or cancel the search. If the named picture is found, the graphic window is scrolled so that the picture appears centered and highlighted in the window.

When a .rsg file is constructed, each object in the display is given a name, either explicitly by the user or by system default. These names can be used to search for objects. The names are also used to establish picture links from entities visible in the Text Browser. A further discussion on graphic naming is given in the next section of the manual.

The 'Close' and 'Expose' selections perform the same functions for graphics canvases as described above for text windows.

The 'Goto Position' selection moves the graphics window such that the specified x,y position appears in the center of the window. X,y positions can be specified in pixels or points. A pixel is the smallest unit of graphic resolution available on the physical display screen. A point is 1/72 of an inch, which is the standard unit of measure for text.

The 'Quit' selection terminates execution of the Graphic Browser only. It does not terminate execution of the Text Browser, if one is currently active.

4.2. PictureOf Menu

This menu lists the names of RSL entities to which a picture link exists, from the currently selected graphic object. The link information is taken from the `pictures` attribute in an RSL specification. Consider the following example:

```
operation Op1 is
    components: ... ;
    pictures: Op1_Node, Op_Concept_Diagram;
end Op1;

operation Op2 is
    components: ... ;
    pictures: Op2_Node, Op_Concept_Diagram;
end Op2;
```

This defines two pictures each for the operations named `Op1` and `Op2`. Suppose the operations are defined in the file `op.rsl`. When `op.rsl` is translated, the translator looks in `op.rsg` for each of the objects named in Op's picture attributes. When these graphic objects are located, a two-way picture link is established between the RSL text and graphics. These links are followed from the Text Browser using the 'Pictures' menu. The picture links are followed from the Graphic Browser using the 'PictureOf' menu.

The picture links can be many-way in both directions. An RSL entity can have picture links to multiple pictures. A graphics object can be the picture of one or more entities.

4.3. Options Menu

The 'Options' menu contains the following selections:

- Edit/Browse Mode
- Auto Text On/Off

- Graphics Only On/Off
- Redraw

'Edit/Browse Mode' toggles between the read-only browser and the writable browser/editor. Details of editing are discussed in Section 5 of this manual. The default mode is 'Browse', unless the -e command line option is selected.

'Auto Text' is the analog of the 'Auto Graphics' option in the Text Browser. If 'Auto Text' is On, the companion Text Browser is automatically updated when the current object is changed in the Graphic Browser. Specifically, the entity of which the current graphic object is the "preferred" picture becomes the current selection in the Text browser. The "preferred" picture is the name listed first in the `picture` attribute of an entity. If the current graphic is preferred by more than one entity, then a dialog box will be used to resolve the ambiguity. If the current object is not the preferred picture of any entity, then the state of the Text Browser is not updated.

The 'Graphics Only' option allows the text browser to be used in stand-alone mode, without an accompanying Text Browser. With 'Graphics Only' set On, the most recently opened .rsg file will automatically be displayed in the graphic window.

The 'Redraw' menu selection redraws the screen in the (rare) case that some damage occurs.

4.4. Picture Formats

As noted above, there are two picture formats available in the Graphic Browser. Free-form drawings can be linked to any RSL entity. They represent a a form of "graphical comment" that aids the user in understanding a specification. A dataflow diagram is a specialized form of picture, depicting the input/output relationships between operations.

Both picture formats are created using the Graphic Browser, in editing mode. This mode is described in the next section of the manual.

5. Creating Browseable Documents

In the default mode of the browsers, displayed text and graphics are not editable. With the 'Edit Mode' option selected, the text and graphics can be created and modified.

Externally, there is no immediate difference in the Text Browser when editing is enabled. The operational difference is that the text windows are editable, using a subset of the commands available in the emacs text editor. For the Graphic Browser, two new palettes are enabled:

- A drawing palette, containing tools for drawing and manipulating graphic objects.
- A dataflow diagram palette, containing tools for creating dataflow nodes and links.

The following subsections describe the basics of text and graphic editing. Sections 6, 7, and 8 describe details of text, graphics, and dataflow editing.

5.1. Basic Editing

An RSL text document is represented in fully standard ASCII text. Given this, RSL text can be edited in either the RSL Browser or any standard text editor. For large RSL documents, it is generally more convenient to edit in a standard UNIX editor, such as emacs or vi, since the browser editing facilities are not as powerful as those of other UNIX text editors. It is also convenient to use a standard text editor when X Windows support is not available to run the RSL Text Browser. Users who do not desire the power of a UNIX editor can create and edit RSL text documents entirely within the Text Browser.

An RSL graphic document must be created using the editing facilities of the Graphic Browser. Other X Windows drawing editors cannot be used, since they do not provide the means to name graphic objects such that the objects are linkable to RSL text.

To facilitate the creation of graphics, the editing facilities of the Graphic Browser are based on the *idraw* graphics editor. Idraw is a popular, full-feature graphics editor, available on many UNIX workstations. The specialized RSL editor is called *ridraw*. It includes the following enhancements to standard idraw:

- The ability to name graphic objects, in order to establish RSL picture links.
- Specialized facilities for constructing dataflow diagrams.

Given the collection of available tools, the following is the most practical methodology for creating browseable RSL documents, consisting of both text and graphics:

- Create an RSL text document using a standard UNIX text editor. The emacs editor provides the most power, since an RSL-mode has been written for emacs.
- Create an RSL graphic document using the ridraw graphic editor. Provide names for graphic objects that correspond to names specified in RSL `picture` attributes.
- Run the translator on the text document to perform checking.
- Use the browser to navigate through the document.
- Use edit mode in the browsers to make minor updates to the RSL text and/or graphics.

5.2. Establishing Picture Links

Picture links are defined using a simple, symbolic connection technique. This allows RSL text documents to be 100% standard text, and facilitates a simple representation for RSL graphics. To create picture links, the following steps are performed:

1. In the RSL text document, define `picture` attributes for all entities for which picture support is desired.
2. Within the `picture` definition, provide unique picture names for each linkable picture.
3. From Edit Mode of the Graphics Browser, provide matching names for all those names specified in `picture` attributes. This is accomplished by selecting 'Name Object' from the 'Option' menu in the Graphics browser. A dialog box will appear, into which the object name is entered.

5.3. The Rescan and Saving Selections of the File Menu

Once edit mode is enabled, the potential exists to invalidate the linkage information upon which the Browsers rely. Since much of the linkage is based on textual positioning within files, even a small amount of editing can cause the Browsers' information to become invalid. This is the reason that all of the browsing commands become inoperable in Edit Mode. The inoperability is made apparent to the user by removal of the browsing menus.

To reenable browsing, the current collection of RSL files must be rescanned by the translator. This is accomplished using the 'Rescan' selection of the 'File' menu. This selection was added to the 'File' menu upon entry to 'Edit Mode'. When 'Rescan' is chosen, the translator

performs its normal actions to check the RSL specification. If any errors are encountered, a scrollable text window is displayed, containing a list of standard translator error messages. Given errors, the browsing commands remain disabled. The errors can be repaired by additional editing, whereupon 'Rescan' is selected again.

Rescanning can also be accomplished by selecting 'Edit/Browse Mode On' from the 'Options' menu. When this selection is made, a dialog appears asking for confirmation of the rescan. If 'OK' is selected in the dialg, the rescan is performed and Edit/Browse Mode is set off, providing that all files translated successfully. If one or more files failed to translate, a warning dialog is displayed and the system remains in Edit mode.

Three other Edit-Mode menus are 'Save', 'Save As', 'Save All'. These are used to save the contents of an edit window in a UNIX file. With 'Save', the current window is saved in the file from which it was originally opened. This is the file that is displayed in the 'File' subwindow on the browser screen. With 'Save As', a file chooser is displayed, from which the save file can be selected. The file selected with 'Save As' becomes the currently displayed file.

'Save' and 'Save As' save only the one file that is current. If more than one text window is open, the 'Save All' command can be used to save the text in all windows. The filename associated with each text window appears in the top-most banner of the window. 'Save All' save each window to its associated file. There is no 'Save All As' command.

6. Editing the Text Window

The editing facilities available in text windows are based on the Simple Text Editor (sted) available within InterViews library package. Sted editing is in turn based on the emacs text editor.

6.1. Editing Model

The text edit window has a single text selection that is displayed with reversed colors. If the selection is empty, an insertion caret is displayed instead. Characters typed into a window replace the current contents of the selection or add characters at the insertion point.

An emacs-like "minibuffer" is available at the bottom of the screen for entering commands textually. Legal minibuffer commands are those available in the Text Browser 'File' menu. The commands are precisely the same as those in the menu -- the minibuffer simply provides an alternative means to invoke the commands, for those users who prefer not the use the mouse while text editing.

6.2. Mouse Bindings

The left mouse button is used to select text. Clicking the left button in the text window selects a new insertion point; dragging the button selects a range of text. If the mouse is dragged outside the window, the display is scrolled to keep the selection point in view.

The middle and right mouse buttons invoke different ways of scrolling with the mouse. Of course, scrolling can also be performed using the scrollbar in the normal way. The middle mouse button does ´grabber´ scrolling. Clicking and dragging the middle button causes the text to move along with the mouse. This scrolling style is useful for fine control over small scrolling ranges. The right mouse button does 'rate' scrolling. Clicking with the right button and 'pulling back' on the mouse causes the text to scroll upwards at a rate dependent on the position of the mouse. 'Pushing forward' causes downwards scrolling. This style of scrolling is useful for rapidly scanning through a large document.

6.3. Keyboard Bindings

For convenience, many operations can be performed from the keyboard. These operations include scrolling, cursor movement, and command execution. The keyboard bindings mimic similar bindings in standard emacs:

Key	Binding
ESC-V	backward page
CTRL-V	forward page
CTRL-P	backward line
CTRL-N	forward line
CTRL-B	backward character
CTRL-F	forward character
ESC-<	beginning of buffer
ESC->	end of buffer
CTRL-A	beginning of line
CTRL-E	end of line
CTRL-D	delete character
DEL	delete backward character
ESC-=	goto
CTRL-S	search
CTRL-X-CTRL-V	visit
CTRL-X-CTRL-F	file
CTRL-X-K	close
CTRL-X-CTRL-C	quit

7. Editing the Graphics Canvas

The graphics editing facilities available for the 'Graphic Canvas' are based on the idraw editor, available in InterViews library package. As noted above, the two idraw enhancements are object naming and dataflow diagram creation.

The idraw-based editor contains four windows, as shown in Figure 4:

- A Tools palette, initially positioned at the left.
- A Dataflow palette, initially to the right of the 'Tools'
- A horizontal command menubar, along the top.
- A Drawing Canvas, in which the graphic objects are drawn.

Details of the tools palette and command menubar are described in this section. Dataflow editing is described in Section 8.

7.1. Drawing Tools

The drawing tools are:

- Select
- Move
- Scale
- Stretch

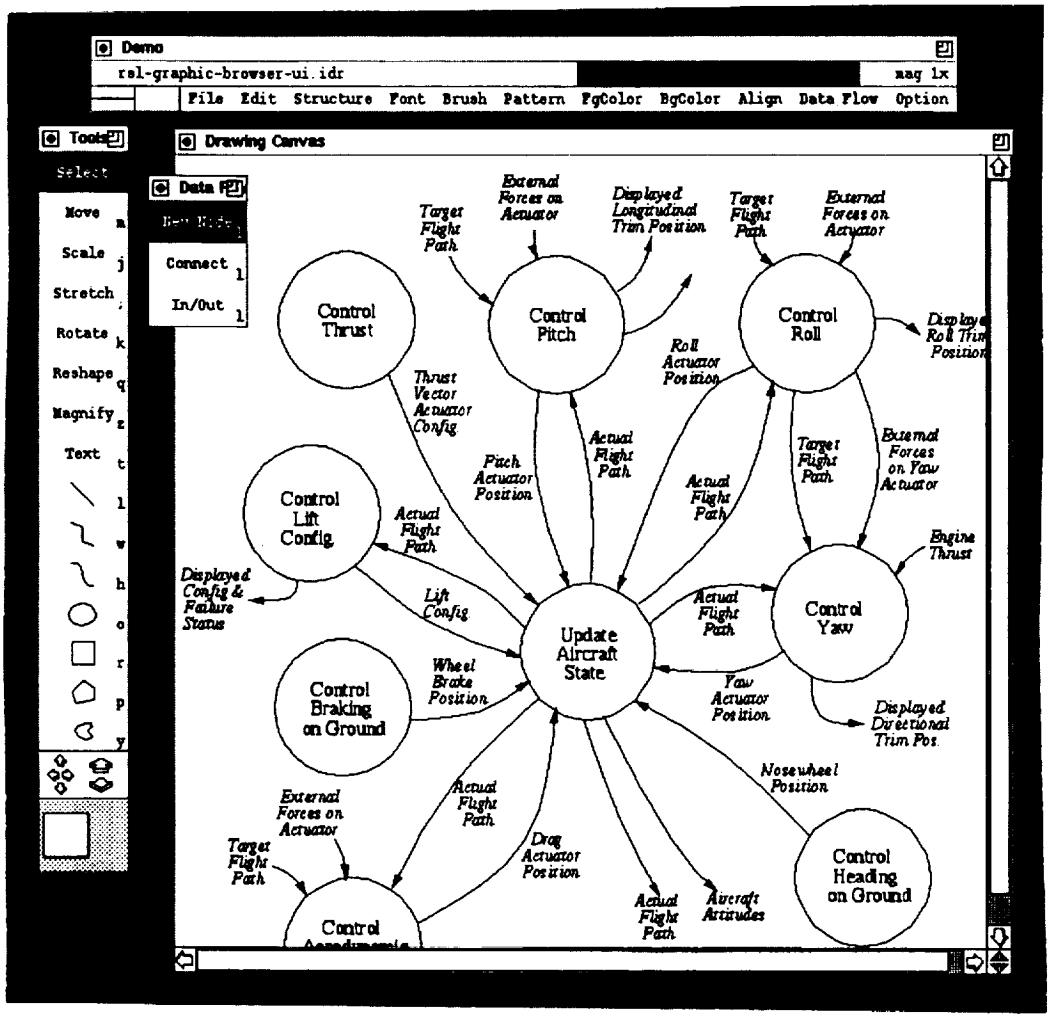

Figure 4: The Idraw-Based Graphics Editor.

- Rotate
- Reshape
- Magnify
- Text
- Line
- Multi Line
- Open Spline
- Ellipse

- Rectangle
- Polygon
- Closed Spline
- A zoomer panner

The first six tools ('Select' through 'Reshape') manipulate existing graphics. 'Magnify' makes a part of the view expand to fill the entire view. The following eight tools ('Text' through 'Closed Spline') create new graphics. At the bottom of the tools palette is a zoomer/panner device.

The shift key constrains the last group of drawing tools if pressed at the same time as the mouse click that begins drawing. The current drawing tool is selected by clicking on its palette item or typing its associated character. The current drawing tool is highlighted when selected. Clicking the left mouse button in the drawing area invokes the current drawing tool. Details of drawing tool operation follow.

7.1.1. Select

'Select' selects a graphic whether or not it is already selected. All other selections are deselected. The selection is indicated by the appearance of small squares around the enclosing rectangle of the selected graphic. These are the selection handles.

By holding down the shift key, 'Select' retains the previous selections while including an unselected graphic or excluding a selected graphic. If the button is pressed outside any graphic, all of the selections are cleared. A group of objects can be selected by dragging a rubberband rectangle around the group. As a shortcut, the right mouse button invokes 'Select' while the mouse is in the drawing area.

7.1.2. Move

'Move' moves graphics from one location in the drawing canvas to another. As a shortcut, the middle mouse button invokes 'Move' while the mouse is in the drawing area. A selection of multiple graphics is moved in the same manner as a single graphic.

7.1.3. Scale, Stretch, and Rotate

These tools perform the graphic transformations suggested by their names. 'Scale' scales graphics about their centers. 'Stretch' stretches graphics vertically or horizontally while tying down the opposite edge. 'Rotate' rotates graphics about their centers according to the angle between two radii: the one defined by the original clicking point and the one defined by the current dragging point.

7.1.4. Reshape

To invoke 'Reshape', the user moves one of a graphic's selection handles. The handle will be moved and the affected area of the graphic will be reshaped accordingly. Nothing happens with reshape applied to an ellipse or circle because these objects do not have handles suitable for dragging. Reshape on text enters text editing mode, in which the text can be changed using emacs-like keystrokes. Text edit mode is terminated when the mouse is used to perform some other command.

7.1.5. Magnify

'Magnify' enlarges the part of the drawing specified by dragging a rectangle to fill the entire view. Idraw limits magnifications to powers of two between predefined minimum and maximum magnifications.

7.1.6. Text

'Text' creates text. The user clicks at the text insertion point in the drawing canvas. Emacs-style keystrokes are used to edit the text as well as enter it. Text mode is terminated when the mouse is used to perform some other command.

7.1.7. Line, Multi Line, Open Spline, Ellipse, Rectangle, Polygon, Closed Spline

These tools draw the graphic shape suggested by their names. The drawing is initiated by clicking the left mouse button on the drawing canvas and then dragging the mouse to another point. Four of the tools draw objects containing multiple points: 'Multi Line', 'Open Spline', 'Polygon' and 'Closed Spline'. To draw these shapes, the user drags the mouse to each successive point and clicks the left mouse button. Any button except the left mouse button terminates the drawing.

For 'Line' and 'Multi Line', the shift key constrains the line segments to lie on either the vertical or horizontal axis. For 'Open Spline' and 'Closed Spline', the shift key constrains each control point to lie on either the vertical or the horizontal axis with the preceding point. For 'Ellipse', the shift key constrains the ellipse to the shape of a circle. For 'Rectangle', the shift key constrains the rectangle to the shape of a square. For 'Polygon' the shift key constrains each side to lie on either the vertical or the horizontal axis.

7.1.8. Zoomer Panner

At the bottom of the drawing tools palette is a "Zoomer/Panner" control device. The device contains the following controls

- four small arrows, for panning
- two larger arrows, for zooming
- a white box over a grey background, for panning

The panning controls move the drawing within the boundaries of the physical window in which the drawing is displayed. Each of the four panning arrows moves the drawing a small amount in the indicated direction. The panning arrows are an alternative to the horizontal and vertical scroll bars attached to the drawing canvas. Panning automatically updates the scroll bars, and vice versa.

The zooming buttons enlarge or shrink the drawing by powers of two. The upward facing zoomer enlarges (zooms in); the downward facing zoomer shrinks (zooms out).

The panning box shows the relative size of the drawing with respect to the viewing window. The grey area in the panning box shows the amount of the drawing that cannot currently be seen. The white area in the panning box shows the size of the window. Moving the window box over the grey area pans the window over the drawing. If a drawing is fully visible within the current window, then the panning box is entirely white. Zooming and resizing the window affect the panning box by enlarging or reducing the relative sizes of window and drawing.

7.2. Command Menus

The pulldown menus appear in the horizontal menubar initially placed above the drawing canvas. The menus are:

- File
- Edit
- Structure
- Font
- Brush
- Pattern
- FgColor
- BgColor
- Align
- Option

These commands are executed by pulling down the menu and releasing the mouse button on the command or by typing the character associated with the command.

7.2.1. File Menu

The 'File' menu contains the following commands to operate on files. The keystroke shown in parentheses after the command name is the keyboard shortcut for the command.

- New (CTRL-N)
- Revert (CTRL-R)
- Open (CTRL-O)
- Save As ... (CTRL-A)
- Save (CTRL-S)
- Open Dataflow (<)
- Save Dataflow (>)
- Print ... (CTRL-P)

'New' destroys the current drawing and replaces it with an unnamed blank drawing. 'Revert' rereads the current drawing, destroying any unsaved changes. 'Open...' displays a file chooser, which is used to browse the file system looking for an existing drawing to open. 'Save As...' saves the current drawing in a file chosen in a dialog box. 'Save' saves the current drawing in the file it came from.

'Open' and 'Save Dataflow' operate on dataflow diagrams. Details are given in Section 8 of the manual.

'Print...' prints the current drawing by sending it through a UNIX pipe to a command entered in a dialog box. For example, "lpr -Pps" will send the file to the printer named "ps" (presumably the local PostScript printer). The bold rectangular outline (called the "page boundary") appearing in the drawing area indicates the portion of the drawing that will appear on the printed page.

7.2.2. Edit Menu

The **Edit** menu contains the following commands to edit graphics:

- Undo (U)
- Redo (R)
- Cut (x)
- Copy (c)
- Paste (v)
- Duplicate (d)
- Delete (CTRL-D)
- Select All (a)
- Flip Horizontal (_)
- Flip Vertical (|)
- 90 Clockwise (])
- 99 CounterCW ([)
- Precise Move ... (M)
- Precise Scale ... (J)
- Precise Rotate ... (K)

'Undo' retracts the last change done to the drawing. Successive 'Undo' commands undo earlier and earlier changes back to the last stored change and then do nothing. 'Redo' restores the last undone change to the drawing. Successive 'Redo' commands redo later and later changes up to the first change undone by 'Undo' and then do nothing. Any new changes cause all undone changes to be forgotten.

'Remove' extracts the selected graphics from the drawing and places them in a file called ".clipboard" in the user's home directory for later retrieval. 'Copy' copies the selected graphics into the clipboard. 'Paste' pastes copies of the graphics in the clipboard into the drawing. Together, 'Cut', 'Copy', and 'Paste' allow the user to transfer graphics between multiple instantiations of idraw simply by cutting graphics out of one view and pasting them into another.

'Duplicate' duplicates the selected graphics and adds the copies to the drawing. 'Delete' destroys the selected graphics. 'Select All' selects every graphic within the drawing canvas.

'Flip Horizontal' flips the selected graphics into their mirror images along the horizontal axis. 'Flip Vertical' flips the selected graphics into their mirror images along the vertical axis. '90 Clockwise' rotates the selected graphics 90 degrees clockwise. '90 CounterCW' Rotates the selected graphics 90 degrees counterclockwise.

'Precise Move...' moves graphics by exact amounts, which are typed in a dialog box. Movements are specified in units of screen pixels or points. 'Precise Scale...' scales graphics by exact amounts, which are typed in a dialog box. Scalings are specified in terms of magnifications along the axes. 'Precise Rotate...' rotates graphics by exact amounts, which are typed in a dialog box. Rotation is specified in degrees.

7.2.3. Structure Menu

The 'Structure' menu contains the following commands to modify the structure of the drawing, that is, the order in which graphics are drawn:

- Group (g)
- Ungroup (u)
- Bring To Front (f)
- Send To Back (b)
- Number of Graphics (#)

'Group' nests the selected graphics in a newly created group. A group is just a graphic that contains other graphics. 'Group' allows the user to build hierarchies of graphics. 'Ungroup' dissolves the selected groups. 'Ungroup' reverses a 'Group' operation. If a group contains several subgroups, each successive ungroup dissolves each nested group, in reverse order of group creation.

'Bring To Front' brings the selected graphics to the front of the drawing so that they are drawn on top of (after) the other graphics in the drawing. 'Send To Back' sends the selected graphics to the back of the drawing so that they are drawn behind (before) the other graphics in the drawing.

'Number of Graphics' counts and displays in a popup window the number of selected graphics, recursively adding the number of graphics nested inside groups among the selected graphics. 'Select All' followed by 'Number of Graphics' counts the total number of graphics in the drawing.

7.2.4. Font Menu

The 'Font' menu contains a set of fonts with which to print text. When the current font is set from the menu, the font of all the selected graphics objects in the drawing canvas is also set. A font indicator in the upper right corner displays the current font.

7.2.5. Brush Menu

The 'Brush' menu contains a set of brushes with which to draw lines. When the current brush is set from the menu, the brush for all the selected graphics objects in the canvas is also set. The nonexistent brush draws invisible lines and nonoutlined graphics. The arrowhead brushes add arrowheads to lines', multi lines', and open splines' starting point, ending point, or both. These graphics can be freely stretched or scaled without affecting the arrowheads' size. A brush indicator in the upper left corner displays the current brush.

7.2.6. Pattern Menu

The 'Pattern' menu contains a set of patterns with which to fill graphics but not text. Text always appears solid, but a different color than black can be used to obtain a halftoned shade. When the current pattern is set from the menu, the pattern is also set for all the selected graphics objects in the canvas. The nonexistent pattern draws unfilled graphics while the other patterns draw graphics filled with a bitmap or a halftoned shade.

7.2.7. Fgcolor and BgColor Menus

These menus contain a set of colors with which to draw graphics and text. When the current foreground or background color is set from the FgColor or BgColor menu, the foreground or background colors are set for all the selected graphics objects in the canvas. The "on" bits in the bitmaps for dashed lines and fill patterns appear in the foreground color while the "off" bits appear in the background color. A black and white printer will print a halftoned shade of gray for any color other than black or white. The brush, pattern, and font indicators all reflect the current colors.

7.2.8. Align Menu

The **Align** menu contains the following commands to align graphical objects with other graphical objects:

- Left Sides (1)
- Right Sides (2)
- Bottoms (3)
- Tops (4)
- Vertical Centers (5)
- Horizontal Centers (6)
- Centers (7)
- Left To Right (8)
- Right To Left (9))
- Bottom To Top (0)
- Top To Bottom (-)
- Align To Grid (.)

The first graphical object selected stays fixed while the other objects move in the order they were selected, according to the type of alignment chosen. The last Align command, Align To Grid, aligns each selected object's lower left corner with the nearest point in a grid of points spaced every 8 printer's points (one ninth of an inch).

7.2.9. Option Menu

The **Option** menu contains the following commands:

- Reduce (i)
- Enlarge (e)
- Normal Size (n)
- Reduce To Fit (=)
- Center Page (/)
- Redraw Page (CTRL-R)
- Gridding on/off (,)
- Grid visible/invisible (?)
- Grid spacing ... (S)

- Orientation (+)
- Name Object (O)

'Reduce' reduces the magnification by a factor of two so the drawing's size decreases by half. 'Enlarge' enlarges the magnification by a factor of two so the drawing's size doubles. 'Normal Size' sets the magnification to unity so the drawing appears at actual size. 'Reduce to Fit' reduces the magnification until the drawing fits entirely within the view. 'Center Page' centers the view over the center of the 8.5 by 11 inch page.

'Gridding on/off' toggles the grid's constraining effect on or off. When gridding is on, new graphical objects will use only grid points as their points, existing objects will move or scale only in grid-sized units, and so forth. 'Grid visible/invisible' toggles the grid's visibility on or off. When visibility is on, idraw draws a grid of equally spaced points behind the drawing. 'Grid spacing...' changes the grid spacing to a value (in units of printers' points of which there are 72.07 to an inch), entered in a dialog box.

'Orientation' toggles the drawing's orientation. If the editor was formerly showing a portrait view of the drawing, it will now show a landscape view of the drawing and vice versa.

'Name Object' provides a name for the selected object in the drawing canvas. As described earlier, this name is referenced in RSL `picture` attributes to establish text-to-graphics links.

8. Editing Dataflow Diagrams

Figure 4 depicts the interface display to the dataflow tool, which is integrated within the general-purpose drawing editor. The dataflow tool allows a variety of dataflow diagrams to be developed and edited. The user is allowed to specify the shape of the nodes in a diagram, connect nodes with splines or straight lines, and annotate the diagram with graphics and text using the normal features of idraw. Diagrams can be organized into levels on multiple display canvases. "Next level" and "parent" links can be established to nodes at different levels.

When connections are made between nodes, the system retains the connections by moving the endpoint(s) to the edge of the node(s) to which they are connected. When a node is moved, the edges connected to the node are automatically modified so that they remain connected to the node. Also, the connector labels are moved so that they retain their relationship to the connections that they label.

As shown in the figure, the dataflow editing is integrated with the graphics editor, This allows the user to annotate the dataflow diagrams with explanatory graphics and text.

The dataflow palette contains three functions that are initiated by clicking the mouse on the display canvas. The 'Dataflow' pulldown menu is located in the menubar between 'Align' and 'Option'. The menu contains five operations that are part of a pull-down menu at the bottom of the palette.

8.1. Dataflow Functions

There is a current function associated with the dataflow palette. The current function is always highlighted, and can be changed by clicking the mouse on an unhighlighted palette item. Following is a brief description of each of the dataflow functions.

8.1.1. New Node

To create a new node, the user selects 'New Node' in the dataflow palette, and clicks anywhere on a drawing canvas. A node appears containing a text cursor that prompts the user to enter a name for the node. If no name is entered (i.e., some other operation is immediately performed), then a default name is given to the node. The user can later edit the name using the "Text Edit" function in the drawing palette. The name can also be relocated within the node, but the system will not allow the name to be moved outside of the boundaries of the node.

8.1.2. Connect Nodes

To connect two nodes, the user selects 'Connect Node' in the dataflow palette and clicks the left mouse button within the boundaries of a node. The user then drags the mouse, with the left button down, and releases the left button at the location of each point desired in the connecting spline. When the final point of the spline is defined, the user clicks the middle mouse button. This final point must be within the boundaries of a node, or the spline disappears. If a valid connector has been drawn, a cursor appears prompting for connector name. As with the 'New Node' function, if no name is entered, a default name is generated for the connector.

8.1.3. In/Out Connector

To define an input or output connector for a node, the user selects 'In/Out' in the dataflow palette, and clicks the left mouse button on the display canvas. The user then draws a spline in the same manner as with the 'Connect Nodes' function. However, the 'In/Out' function requires that exactly one of the endpoints of the spline be within the boundaries of a node, not two, and if this requirement is not met the spline will disappear. If a valid connector has been drawn, a cursor appears prompting for a connector name. As with the 'New Node' and 'Connect' functions, if no name is entered, a default name is generated for the connector.

8.2. Dataflow Operations

The 'Dataflow' menu contains dataflow operations that do not require mouse gesturing on the display canvas. Some of these operations are dependent on what object is selected on the display canvas. A dataflow object is selected with 'Select' from the tools palette, in the same manner that a standard graphics object is selected.

8.2.1. New Level

The 'New Level' operation opens a new display canvas and allows the user to locate the canvas on the screen. If a node is selected in the original display canvas, a virtual "next level" connection is made between that node and the new canvas. Next level connections allow users to traverse through a multiple-canvas diagram in a hierarchical manner.

8.2.2. Levelize

When display canvas is becomes crowded, the user can choose the 'Levelize' command to move a group of nodes and connectors to another canvas at the next level of detail. When 'Levelize' is chosen, a new canvas is opened, and all selected nodes and connectors on the original canvas are moved to the new canvas. The transplanted nodes are replaced by

a single "parent" node, and the user is prompted to enter a name for the parent. All nodes still on the original canvas that were connected to transplanted objects are connected to the parent node. A next level connection is automatically made between the parent node and the new canvas.

8.2.3. Goto Next Level

The user can traverse a next level connection by selecting the 'Goto Next Level' operation. When this operation is performed, the system checks the next level connection of the node selected on the display canvas. If the next level connection exists, the canvas containing the next level of dataflow is made visible (if necessary), deiconified (if necessary), and raised to the forefront of the screen.

8.2.4. Set Next Level

The 'Set Next Level' command allows the user to explicitly set the next level connection of the object selected on the display. When this operation is chosen, a dialog box appears, prompting the user to enter the name of the canvas that should be connected to the selected object.

8.2.5. Set Default Node

The user can change the shape of the node that is created with subsequent 'New Node' commands. This is performed by drawing a graphic with drawing tools on the graphics palette, selecting the object using the 'Select' function, and choosing the 'Set Default Node' command. When this operation is chosen, the system copies the internal representation of the object selected on the display and uses it as the shape of all new nodes created. This default object can be a rectangle, an ellipse, or a polygon, and can have any size and color.

8.3. Saving and Loading Diagrams

The idraw 'File' menu contains two additional commands to save and load dataflow diagrams. Diagrams can be saved in two formats: a textual format that can be reloaded into the dataflow tool, or in postscript format for printing purposes. The 'Save Dataflow' command saves files for subsequent reloading with the 'Load Dataflow' command. The normal idraw 'Save' and 'Save As' commands create postscript files. Postscript files can also be loaded into the system, but the document loaded will be raw graphics, with no special relationships between nodes and connectors.

9. Summary of the Execution Environment

The RSL tools have been developed to run under the UNIX operating system. The translator is written in ANSI C, using standard UNIX versions of the YACC and LEX compiler tools. To date, the translator has been compiled on four UNIX platforms: SUN3, SUN4, AIX, and HPUX. It is reasonable to expect that the translator can be compiled on other UNIX systems, given ANSI C, YACC, and LEX. It should also be possible to port the Basic Translator to a non-UNIX system, given that ANSI C, YACC, and LEX are available.

The browsers execute under the X Windows system. The browsers are written in C++, using Version 2.6 of the InterViews interface library. To date, the browsers have been compiled on the SUN3 and SUN4 platforms. It is reasonable to expect that the browsers

can be compiled on other UNIX systems, as long as the InterViews 2.6 library is available.

When X Windows support is not available, the basic translator can run without the browser. This allows RSL text to be composed and checked on a plain ASCII terminal.

10. Concluding Remarks

This manual has described the toolset for RSL. Development of the language and its tools are ongoing. Tool enhancements scheduled for future release include the following:

- Support for mechanized verification of RSL specifications, based on transformation of RSL to an underlying formal language for which verification support exists. Candidate underlying languages include EHDM [Rushby 91] and HOL [Gordon 85].
- Execution via term-rewriting of equational specifications, as provided in OBJ [Goguen 88].
- Support for the automated generation graphical user interfaces, as described in [Fisher 88, 92] and [Wolber 91a, 91b].

References

[Fisher 88] G. L. Fisher, "An Overview of a Graphical, Multilanguage Applications Environment," *IEEE Transactions on Software Engineering*, June 1988.

[Fisher 91] G. L. Fisher and D. A. Frincke, "Formal Specification and Verification of Graphical User Interfaces," *Proceedings of the Hawaii International Conference on System Science*, January 1991.

[Fisher 92] G. L. Fisher and D. E. Busse, "Adding Rule-Based Reasoning to a Demonstrational Interface Builder", *Proceedings of the Fifth Annual ACM SIGGRAPH Symposium on User Interface Software and Technology*, November 1992.

[Fisher 93] G. L. Fisher and G. C Cohen, "Reference Manual for a Requirements Specification Language (RSL), Version 2.0", NASA Contractor Report 191460, September 1993

[Goguen 88] J. A. Goguen and T. N. Winkler, "Introducing OBJ3", SRI International Technical Report, Palo Alto, CA, August 1988.

[Gordon 85] M. Gordon, "A Proof Generating System for Higher-Order Logic", University of Cambridge Computer Laboratory, January 1987.

[Rushby 91] J. Rushby, "The EHDM Reference Manual", SRI International Technical Report, Palo Alto, CA, 1991.

[Wolber 91a] D. A. Wolber and G. L. Fisher, "Developing User Interfaces By Stimulus-Response Demonstration," *Proceedings of COMPSAC*, July 1991.

[Wolber 91b] D. A. Wolber and G. L Fisher, "A Demonstrational Technique For Developing Interfaces With Dynamically Created Objects", *Proceedings of the Fourth Annual ACM SIGGRAPH Symposium on User Interface Software and Technology*, November 1991.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November, 1993	Contractor Report

4. TITLE AND SUBTITLE
Reference Manual for a Requirements Specification Language (RSL), Version 2.0

5. FUNDING NUMBERS
C NAS1-18586
WU 505-64-10-07

6. AUTHOR(S)
Gene L. Fisher*
Gerald C. Cohen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Boeing Defense and Space Group
P.O. Box 3707, M/S 4C-70
Seattle, WA 98124-2207

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
NASA CR-191460

11. SUPPLEMENTARY NOTES
Langley Technical Monitor: Sally C. Johnson
Task 11 Report
*California Polytechnic State University, San Luis Obispo, CA

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Unclassified - Unlimited

Subject Category 60

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report is a Reference Manual for a general-purpose Requirements Specification Language, RSL. The purpose of RSL is to specify precisely the external structure of a mechanized system and to define requirements that the system must meet. A system can be comprised of a mixture of hardware, software, and human processing elements. RSL is a hybrid of features found in several popular requirements specification languages and includes constructs for formal mathematical specification.

14. SUBJECT TERMS
Requirements Specification
Formal Specification

15. NUMBER OF PAGES
31

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		