A GUIDED TOUR OF FORMS/3

Judith G. Hays
hays@research.cs.orst.edu

Margaret M. Burnett*
burnett@cs.orst.edu

Department of Computer Science
Oregon State University
Corvallis, OR 97331

Technical Report #95-60-6
June 1995
Revised January 1997

Abstract: Forms/3 is a declarative visual programming language that aims to provide general-purpose programming language capabilities in a simple, form-based approach. This report provides an example-driven introduction to programming in Forms/3.

* To whom correspondence should be directed

TABLE OF CONTENTS

LIST OF FIGURES

TYPOGRAPHICAL CONVENTIONS

The following types of formatting in the text identify special information.

Bold type	Words or characters you type. For example, if the instructions say to type **(run)**, you type a left parenthesis followed by run followed by a right parenthesis.
[item,...]	Items specified inside braces are optional
<u>Cancel</u>	Underlined words indicate buttons on the screen which should be clicked with the mouse or the name of important areas within a form.
Tips	Sections which are shaded and enclosed in a box contain technical information which can be skipped on the first reading, or are of interest to more technical users.

INTRODUCTION

Forms/3 is a form-based Visual Programming Language (VPL). Its goal is to provide computational and expressive power in a language featuring a simple, concrete programming style with immediate feedback.

This document provides a view of Forms/3 from a user's perspective. It is by no means a complete language description. Rather, it simply presents a picture of what it is like to use Forms/3 through a set of examples. When used with a live system, it provides a guided tour. However, we have tried to make the descriptions complete enough to be useful even without a live system.

EXAMPLE 1 - AREA OF A SQUARE: CONCRETENESS AND IMMEDIATE VISUAL FEEDBACK

OVERVIEW

Forms/3 borrows the spreadsheet ideas of cells and formulas as the basic way that programming is done. This example shows how to use this method to calculate the area of a square.

DETAILED INSTRUCTIONS FOR CALCULATION OF AREA OF A SQUARE

Load and execute Forms/3 as described in Appendix A. The Forms/3 main window, which is shown in Figure 1, should appear on the screen. The System form which is listed in the Forms Loaded: box always loads at this time.

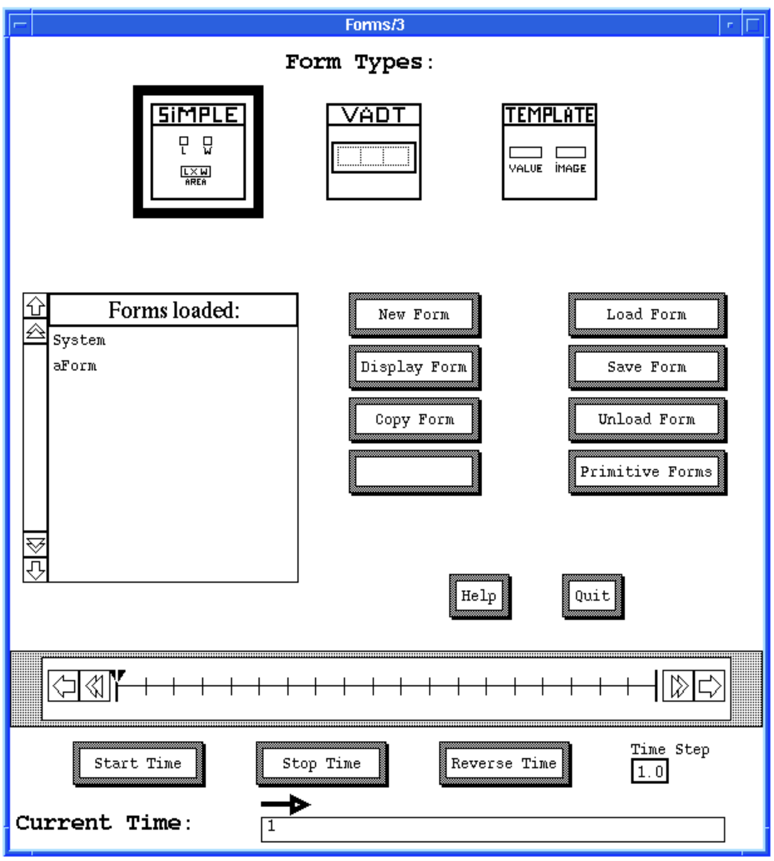

Figure 1 - Forms/3 Main Window

1. Start by creating a form to work on. Click on the <u>New Form</u> button. A dialog box appears that prompts for the name of the new form. See Figure 2.

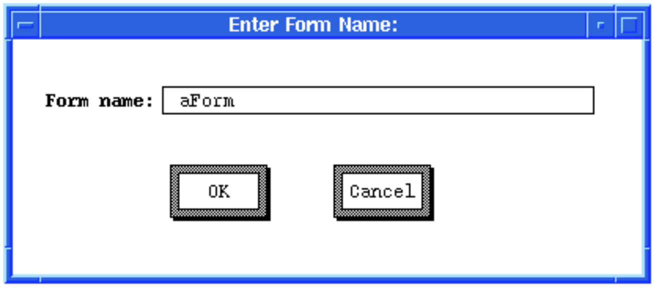

Figure 2 - A New Form Dialog Box

A default name appears in the dialog box. Change the name of the form to **Area-Box** and click <u>OK</u>. A new form will appear called <u>Area-Box</u>.

2. Select the cell tool from the palette to the left with the left button. (The left button is used for selecting, and for many other tasks. The right button is always used for creating new objects). Click in the upper left corner of the <u>Area-Box</u> form with the **<u>right</u>** button. A cell will now appear. This cell consists of an outline with black and clear handles and a formula tab. The black handles can be used to resize the cell, and the white handles can be used to move the cell on the form. See Figure 3.

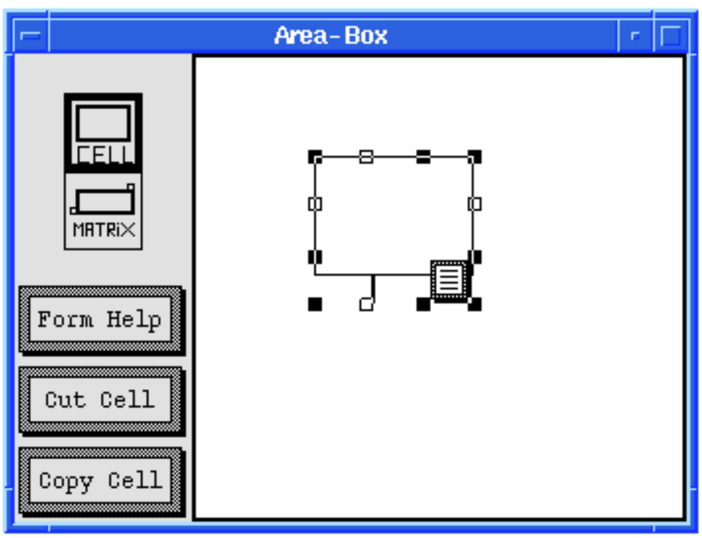

Figure 3 - A Form Containing a New Cell

3. Click to the left of the formula tab and slightly below the black line at the bottom of the cell but above the cell selection handles. When the cursor appears, (shown in Figure 3 as a **|**) type in the name of the cell -- **Abox** in this case.

4. Double click on the formula tab. A dialog box appears. In the formula entry area in the lower area of the dialog box (see Figure 4) click as close to the upper left-hand corner as possible and when the data entry cursor appears type the formula, in this case the number **5**. Click <u>Accept</u>. Note that in the

interior of the cell the evaluated formula is displayed (in this case 5). This is an example of immediate feedback - whenever you enter a formula, all affected values are automatically displayed. Refer to Appendix B for a full list of Forms/3 operators.

5. Repeat steps 2-3 above and create another cell to the right of the Abox cell and call it **Area**.

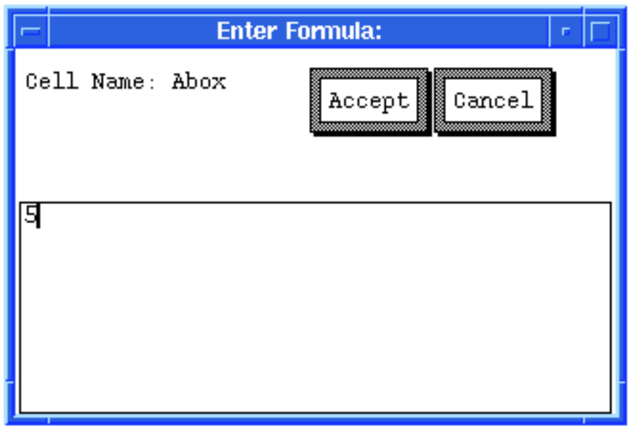

Figure 4 - A Formula Dialog Box with a Formula Entered

6. Now the area of the square will be calculated. Double click on the formula tab for the Area cell When the formula dialog box appears type **Abox** and a **space** and a * and a **space** and then click on the Abox cell. This shows the two ways to refer to a cell - either by clicking on it, or by typing its name. Click Accept.

7. Note that the Area cell now displays the evaluated formula, the number 25. See Figure 5.

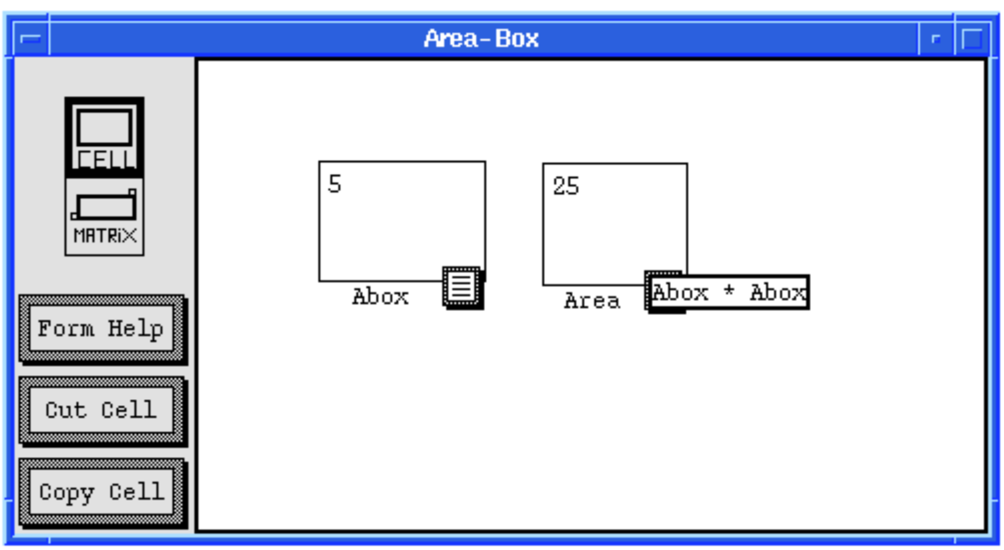

Figure 5 - The Completed Area-Box Form

8. Double click on the formula tab of cell Abox and change the 5 to a 6. Note that the Area cell now has the number 36 displayed.

9. Now save this form for later use. Refer to the Forms/3 main window and in the Forms loaded box click on Area-Box to select it. Click on the button Save Form. A Save Form File dialog box appears which asks you to specify a location in which to save the form. The location is a standard UNIX file path name. A default name is given which is ~Forms/form. If you do not have a Forms directory in your home directory then type another path name for your file or open another window on your X-terminal and make a directory called Forms in your home directory. Click on the Save button.

10. If you want to stop now click the Quit button on the Forms/3 main window.

BASIC OPERATIONS ON CELLS AND FORMS

Continue using the Area-Box form which was saved in step 9 above. If you do not have this form on your screen follow the instructions in the following section, How to Load a Form. Otherwise skip to the section, How to Change the Size of a Cell.

How to Load a Form

From the Forms/3 main window click on the Load Form button. A dialog box appears which prompts for the path name of the file in which the form was saved. A default path name is given. Type in the name of the form as a standard UNIX pathname. When the path name has been typed in click Load.

How to Change the Size of a Cell

When Defining a New Cell

On the Area-Box form below the Abox and Area cells create a new cell by clicking and not releasing the *right* button of the mouse. As you drag the mouse around the size of the cell changes.

After Cell has been Defined

Click on the Area cell. Small black and clear handles appear around the cell. (See Figure 3 above) . Click and position the pointer of the mouse in the black handle in the lower left hand corner and without lifting the mouse drag the handle toward the bottom of the form. The Area cell should now have changed size vertically. Other size changes can be made by moving different black handles.

How to Move a Cell on the Form

Select the cell by clicking on it. Position the pointer of the mouse in the one of the clear handles (see Figure 3) and without releasing the mouse move the cell to another position on the form.

How to Change the Cell Attributes

Several options are available to change the attributes of a cell on a form. Double click in the Abox cell within the black outline of the cell while holding down the Control key on your keyboard. A Cell Attribute Window appears. See Figure 6. Buttons which are darkened describe the current attributes of the Area cell. Underneath the Border attribute click Hidden, underneath the Formula Tab attribute click Hidden and than Accept. You should now see the Abox cell with no border or formula tab and with only the name Abox and the evaluated formula displayed.

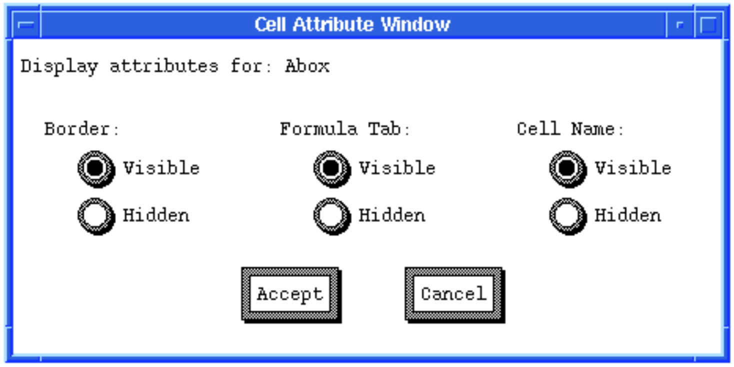

Figure 6 - A Cell Attribute Dialog Box

How to Permanently Display a Formula

There are times when it is convenient to see the formula of a cell as well as the answer. Place the cursor over the formula tab of the Area cell and click and hold the mouse down. While the mouse is held down the formula is displayed beneath the formula tab. When the mouse is released the formula disappears. To leave the formula displayed, click on the formula tab and do not release the mouse. Drag the mouse down inside the formula box and then release the mouse The formula will move up over the formula tab and remain there until you click on it.

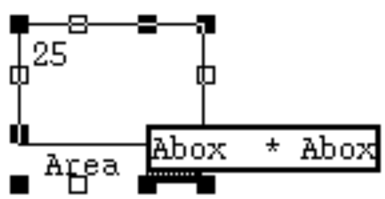

Figure 7 - A Formula Permanently Displayed

How to Delete a Cell

A cell can be deleted from a form by selecting it and then clicking on the Cut cell button on the tool pallet to the left of the form.

Help

The Help button on the Forms/3 main window gives basis information about running Forms/3, and the Help button on every form gives useful information about that particular form.

How to Quit

Click on the Forms/3 main window and then click on the Quit button.

TIPS: SELECTED ITEMS ON THE FORMS/3 MAIN WINDOW

Form Selection

On the main window the user will note at the top of the form there are three form types to choose from. A large black border around the left most form type indicates that the default Simple form will be chosen whenever a new form is created.

Cell Type Selection

On a form in the tool pallet in the upper left-hand corner two types of objects can be chosen. The large black border around the top object indicates that a cell rather than a matrix will be chosen.

EXAMPLE 2 - GRAPHICS

OVERVIEW

In this example the form Area-Box from the previous exercise will be used. The Abox cell's formula will be changed so that the box can be represented by a graphical box rather than a number.

DETAILED INSTRUCTIONS

1. Begin by doing one of the following:

   a. continuing from the end of example 1 with the Area-Box window displayed.

   b. loading the previously saved Area-Box form.

   c. preparing an Area-Box form by following the instructions in Example 1 above.

Once this form is completed it should look similar to Figure 5.

2. Move the Area-Box window out of the way so that the Forms/3 main window can be seen. Create a new form and name it **Graphic-Box**.

3. In the upper right hand corner of this form create a new cell, name it **H** and enter the formula **50**.

4. To the left of the cell defined in step 3 create a new cell called **W** and enter its formula as **50**.

5. In the upper left hand corner create a new cell, call it **AnewBox** and for its formula enter:

   box W H

Notice that a box appears inside the AnewBox cell.

TIPS: THE PRIMITIVE FORMS

The use of the Box W H command is a short cut for use of the primitiveBox form. To explore options on this form, click the Primitive Forms button on the Forms/3 main window and select the primitiveBox from the list.

By clicking on the Help button on the primitiveBox form more information is available.

6. Now change the formula of cell Abox on the Area-Box form to refer to the graphical box just created. To do this change cell Abox so that it is a reference to the AnewBox cell. Delete the current

formula, click on AnewBox and then Accept the formula). Note that a box now appears in the AnewBox cell and also that references can be made across forms.

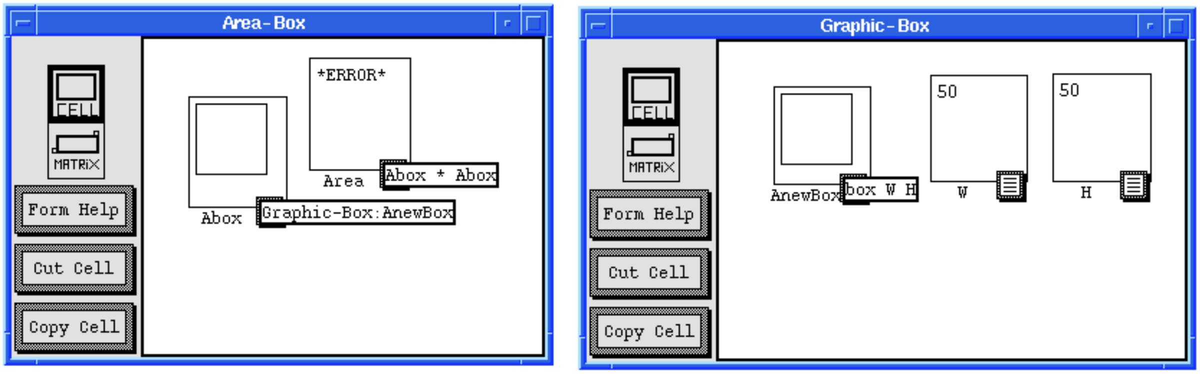

Figure 8 - Graphic-Box and Area-Box Forms

The Area cell is now displaying an *Error* for the formula. This is because there is no way to multiply boxes together. Open the formula for cell Area and change to:

(width Abox) * (height Abox)

Notice that the Area cell now contains the correct area. Move to the Graphic-Box form and change the formula in the W cell to 30 and the formula in the H cell to 30. Note that the changes are propagated across both forms. This ends the exercise. Click Exit on the Forms/3 main window.

EXAMPLE 3 - USER DEFINED GRAPHICAL DATA TYPES

OVERVIEW

Forms/3 allows the user to define their own data types. Part of this definition includes how user-defined types should appear on the screen.

DETAILED INSTRUCTIONS

1. Start this exercise with a predefined VADT form. From the Forms/3 main window load form:

~burnett/Resrch/Forms/examples/hays/VADT-Person

2. The VADT-Person form should look like Figure 9 below.

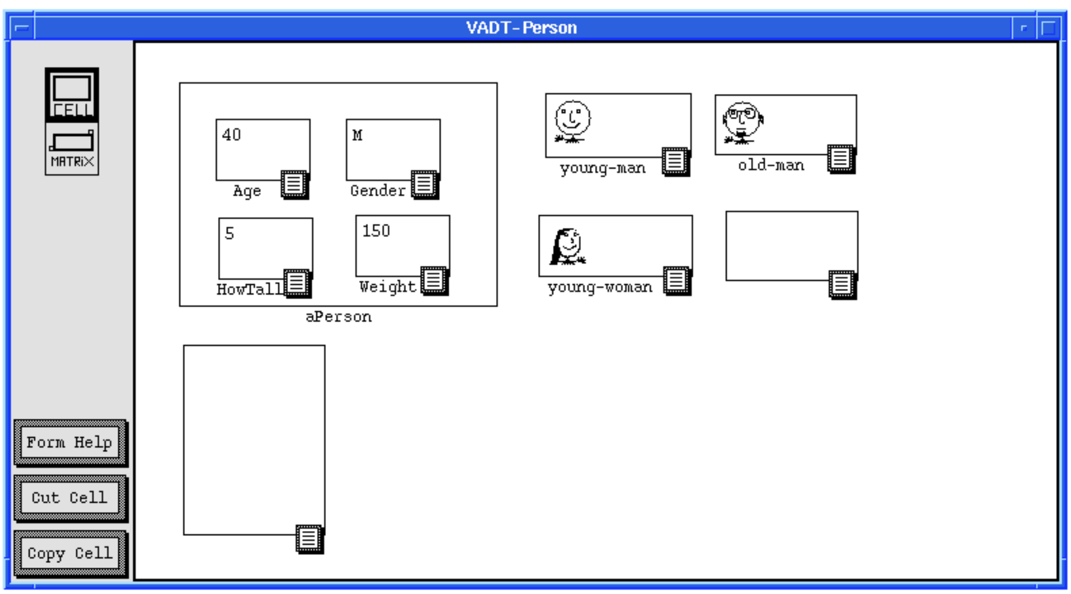

Figure 9 - The VADT-Person Form

The VADT-Person form is a form that defines an abstract data type person. The user has specified that a person is implemented from the composite of its Age, Gender, Height (HowTall) and Weight. This exercise will define the appearance of an instance of type aPerson. To help with this task the cells at the right of the form, which contain bitmaps that were created with an ordinary X-Windows bitmap editor, will be used. Pull down the formula tabs to explore the formulas of the cells on the form. Note that one of the cells on the right is empty.

3. Name this cell old-woman and for the formula type:

glyph "~burnett/Resrch/Forms/video/oldwoman.bitmap"

4. Now create two additional cells, one to calculate the face and one to calculate the body. Do this below the aPerson VADT and to the right of the empty cell in the lower left-hand corner. Put a smaller cell on top of a longer cell as in Figure 10. The upper cell should be named Face and the lower cell Body.

5. The face will depend on aPerson's age and gender so the formula of the Face cell type should be entered:

```
if (Person[Age] < 20)
    then (if (Person[Gender] = "M") then young-man else young-woman)
    else (if (Person[Gender] = "M") then old-man else old-woman)
```

Tip: The Forms/3 parser works in a left to right manner and so parentheses are needed in complex expressions. For example in the above expression, the parentheses after the then and else but before the if statements must be present.

6. The formula for the body should refer to a graphic box. For the formula for Body enter:

box (Person[Weight] / 3) (Person[HowTall] * 10)

7. Move the Body cell directly under the Face cell. Change the attributes of the Face and Body cells to Hidden for Border, Formula Tab, and cell Name.

8. Now make some changes to aPerson to see how they affect the appearance. Change Age to 4. The young-man face should appear. Change Gender to "F" and the young-woman face should appear. Change HowTall to 10 and a tall person should appear.

9. Now the face and body will be put together. Name the empty cell in the lower left-hand corner and to the left of the Face and Body cells Image. For the formula of Image enter:

Compose Face at (0,0) with Body at (0, (Height Face))

If you prefer, you can demonstrate the above formula instead of typing it. To do so, first open the formula entry window as usual, then use the mouse to visually arrange the Face cell and the Body cell the way you want them, and finally rubberband the arrangement with your mouse.

In the Image cell notice that the young woman's face with a tall body (or whatever formulas you had last entered in the Face and Body cells) appears.

Figure 10 below shows what the completed form looked like after the user arranged the Face cell and Body cell one directly above the other, and hid the names and borders of those two cells.

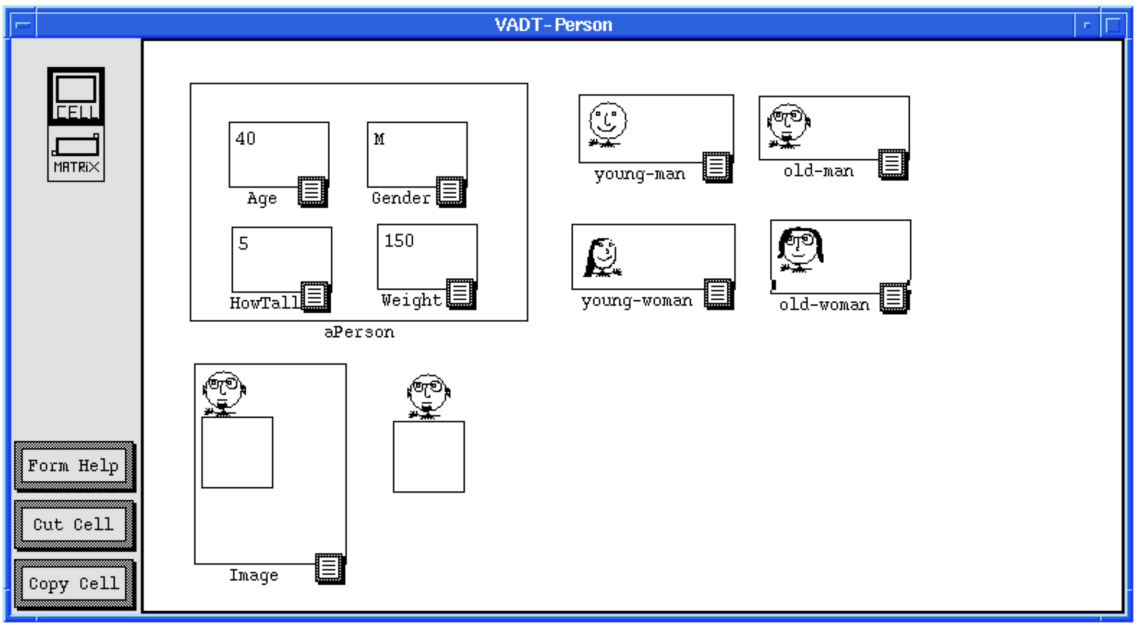

Figure 10 - The Completed VADT-Person Form

10. Change some of the values in aPerson, for example change HowTall to 10, and watch the Body change, both in the Body cell and in the Image cell.

At this point, the abstraction box aPerson contains an instance of type VADT-Person. Outside of this form, which implements this user-defined type, every instance of this type will be encapsulated as an indivisible value, and will appear according to the specifications you have programmed in the Image cell's formula.

11. Open a new form, create a cell, and refer to aPerson on the VADT-Person form. Notice how the resulting value is displayed.

12. There are several example programs based on other implementations of people, some of which are time-based, in **~burnett/Resrch/Forms/examples/personExamples**. You may wish to look at Example 5 in this report before you explore these other examples. When you are ready to proceed with one of these, find a "package" file to load, then press the Start Time button on the Forms/3 main window to watch it run. When the program quits changing, press the Stop Time button and move the slider back to Time 1 before moving on to other examples.

EXAMPLE 4 - RECURSION: FROM CONCRETE FORMS TO REUSABLE ABSTRACTIONS

OVERVIEW

A form allows the user to collect related calculations together into abstractions that can be reused, serving the same purposes as procedures, functions, or type definitions in other languages. In the following example, a form will be set up to concretely specify a reusable abstraction to calculate the n'th number in the Fibonacci sequence.

Recall that if we start counting at 0, the 0'th number in the Fibonacci sequence is 1, followed by another 1, and then each number after that is the sum of the preceding 2 numbers in the sequence. The program will use a recursive approach.

DETAILED INSTRUCTIONS

1. Create a new form and call it Fib. On this Form create two cells, the first on the upper-left should be named N with a formula of 5 and the second on the upper-right should be named Ans.

2. Create two more cells called N-1 and N-2. N-1 should be on the lower left-hand corner and N-2 on the lower-right hand corner of the Fib form.

3. N-1 cell's formula should be entered as **N - 1** and N-2 cell's formula should be **N - 2.**

4. To make the rest of the exercise easier make the Fib window smaller.

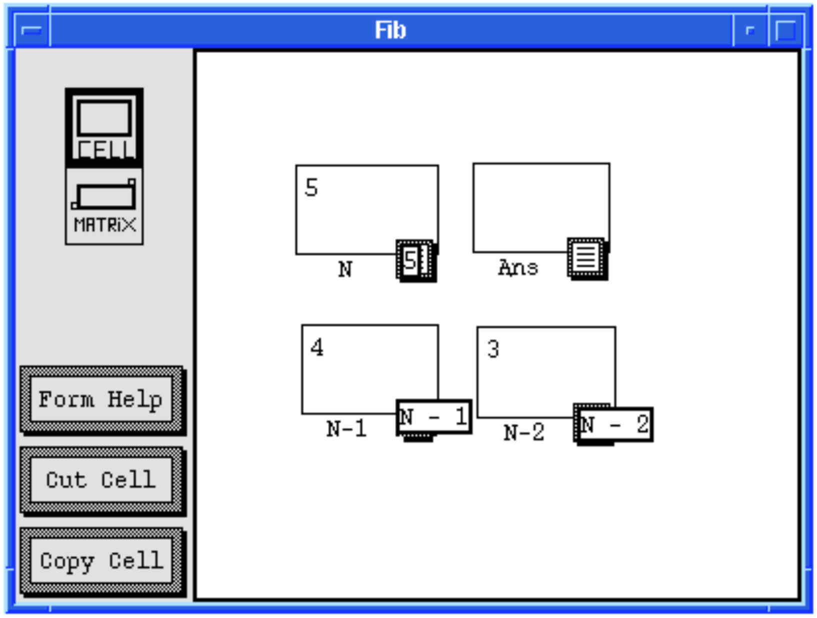

Figure 11 - The First Fib Form

5. Move the Fib form aside so that the Forms/3 main window can be seen. In the Forms loaded section click on the Fib form to select it and then click on the Copy Form button. Note that a form, Fib1, is produced and all of the cells and formulas from Fib have been propagated into it. Fib1 is gray to indicate that it has been copied.

6. Copy form Fib again to produce form Fib2 and note that again the cells and formulas have been propagated into Fib2.

7. On the Fib1 form change the value of the formula in the N cell to refer to the N-1 cell on the Fib form. Delete the 5 and then click on the N-1 cell of the Fib form. (The idea is that Fib1 will calculate the N - 1st Fibonacci number). If you have done this correctly the formula for the Fib1 cell should be Fib:N-1. Notice that the interior of the cell is now white to indicate that the formula on this copy has been changed.

8. On the Fib2 form change the value of the formula in the N cell to refer to the N-2 cell on the Fib form. Delete the 5 and click on the N-2 cell of the Fib form. The formula of the N cell of the Fib2 form should now say Fib:N-2.

9. Notice that appropriate values have been changed in the various cells on all forms.

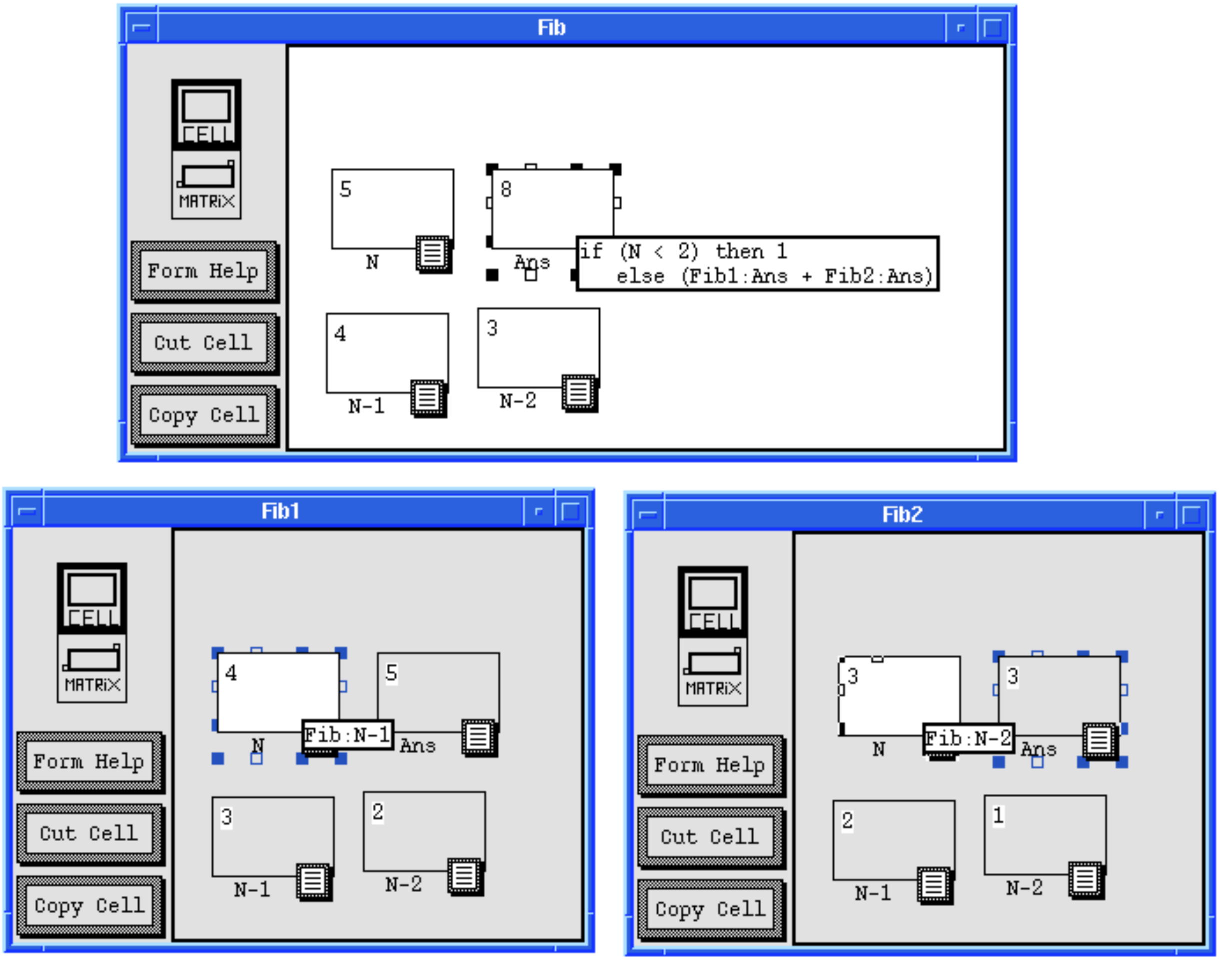

Figure 12 - The Completed Fib Forms

10. Now for the recursive part. Move to the Fib form and in the formula of the Ans cell enter (by typing and clicking on cell references):

if (N < 2) then 1
else (Fib1:Ans + Fib2:Ans)

This formula for answer is exactly what we want for this cell, but when it is "accepted" it will be propagated to the 2 copies. But the formula is too concrete for the copies (Fib1 and Fib2). It must be generalized because the formula for the Ans cell on the Fib1 form will seem to be circular. This will also be true of the formula for the Ans cell on the Fib2 form.

11. The Forms/3 system will address this problem so click Accept. It will analyze the relationships among the cells and automatically derive a generalized, reusable version of the formula.

12. If you want to save these forms you can do so by selecting Fib from the Forms loaded: area of the Forms/3 main window and then clicking on the Save Form button. It is not necessary to save Fib1 and Fib2 as they are automatically regenerated as needed by the recursive definition of Fib.

13. Now unload the Fib Forms. Select Fib in the Forms loaded area of the Forms/3 main window. Then click on the Unload Button. Notice that the Fib form and its associated Fib1 and Fib2 forms are all unloaded at the same time.

EXAMPLE 5 - FACTORIAL: USE OF TIME

OVERVIEW

In Forms/3, a cell's formula can be used to define a sequence of time values for a cell. This example shows how to do this, and how to use the Time Control Buttons and Time Slider to navigate through time.

DETAILED INSTRUCTIONS

1. Open a new form and call it factorial.

2. In the upper-left hand corner of the form create a cell called N and enter the formula 5.

3. In the upper-right hand corner of the form create a cell called Counter and in the formula box enter:

1 fby ((earlier Counter) + 1)

"fby" stands for "followed by", and is one way to define a sequence of values over time. Notice that after the entry the number 1 is displayed. This is because we are currently positioned at the beginning of the time sequence.

4. In the center of the form create a cell called Fact and enter the formula:

1 fby ((earlier Fact) * Counter) until (Counter >= N))

At this stage your form should look like Figure 13 below.

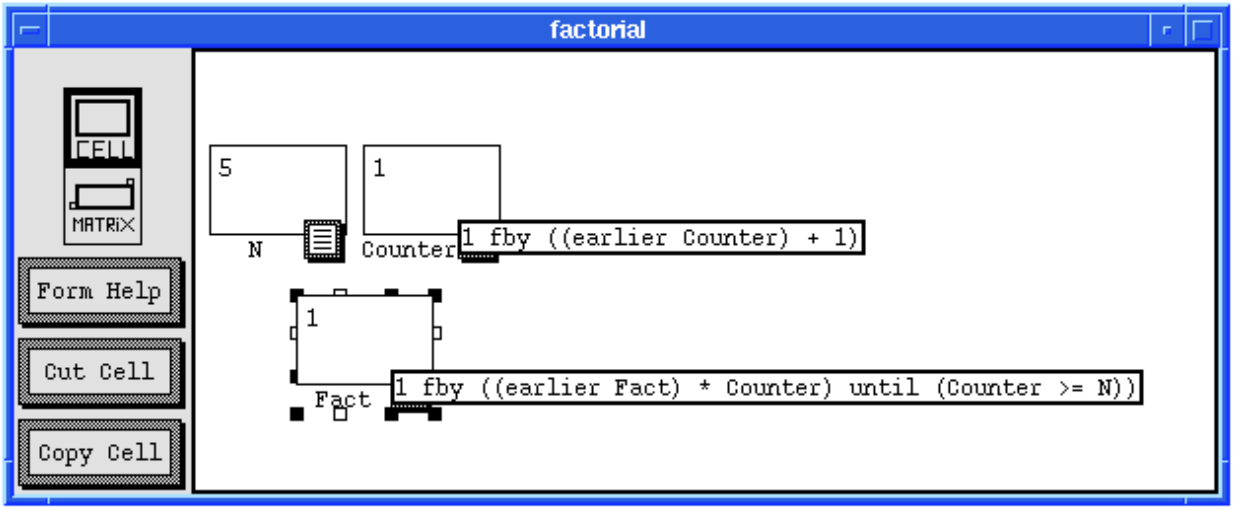

Figure 13 - Completed factorial form

5. Now bring the Forms/3 main window to the front (keep this form and the factorial form on the screen in positions where you can observe then simultaneously). Refer to Figure 1 for the rest of the discussion on this exercise. Notice at the bottom of the Forms/3 main window the Start Time, Stop Time, and Reverse Time buttons. Click on the Start Time button and watch the values changing on the factorial form. Notice that the value keeps changing in the Counter cell but has stopped at the value of 120 in the Fact cell.

6. Click on the Stop Time button to stop the progression of time. Now click on Reverse Time and then Start Time. Observe the factorial form redisplaying the saved values in reversed sequence. Notice on the main window the triangular icon on the time slider. It is moving slowly towards the left side of the slider. Also in the Current Time: Box at the bottom of the main window the numbers are decreasing by 1.

7. Drag the triangular icon near to the left-most vertical tick mark on the slider and release it. Watch as the triangular icon reaches the left edge and then automatically reverses and starts moving to the right.

8. Press Stop Time. Now explore the factorial values at various moments in the sequence by navigating through time. You can do this by dragging the triangular icon, or clicking directly on the slider bar at the desired position.

9. To the left and right of the slider are arrows and side-ways double triangles. The arrows can be used to move the triangular icon one small step in the desired direction. The side-ways double triangles can be used to move the triangular icon in larger steps.

10. A default value of 1.0 is given for the Time Step box in the lower-left corner. This can be edited to increase or decrease the speed that the triangular icon moves through time.

EXAMPLE 6 - A WHEEL ON A RAMP: AN INTRODUCTION TO ANIMATION

OVERVIEW

Formulas can be used in Forms/3 to define animations over time. This example uses an animated rolling wagon wheel program to show how to do "on-the-fly" exploratory programming or debugging by exploiting the fact that any Forms/3 program can be executed forwards or backwards.

DETAILED INSTRUCTIONS

1. Load the following Forms/3 package:

~burnett/Resrch/Forms/examples/animation/wagonWheel/wheelPackage

2. The following forms load: several eventReceptor forms, an animationForm, a wheelInput and a wheelOutput form. The animationForm (see Figure 14) is the rolling wheel part of the program. The wheelInput form (see Figure 15) allows you to specify the slope of the ramp (and other parameters which can be seen by pulling down and enlarging the window.) The wheelOutput form (see Figure 15), combines the rolling wheel, ramp, and user controls to start and stop the rolling wheel.

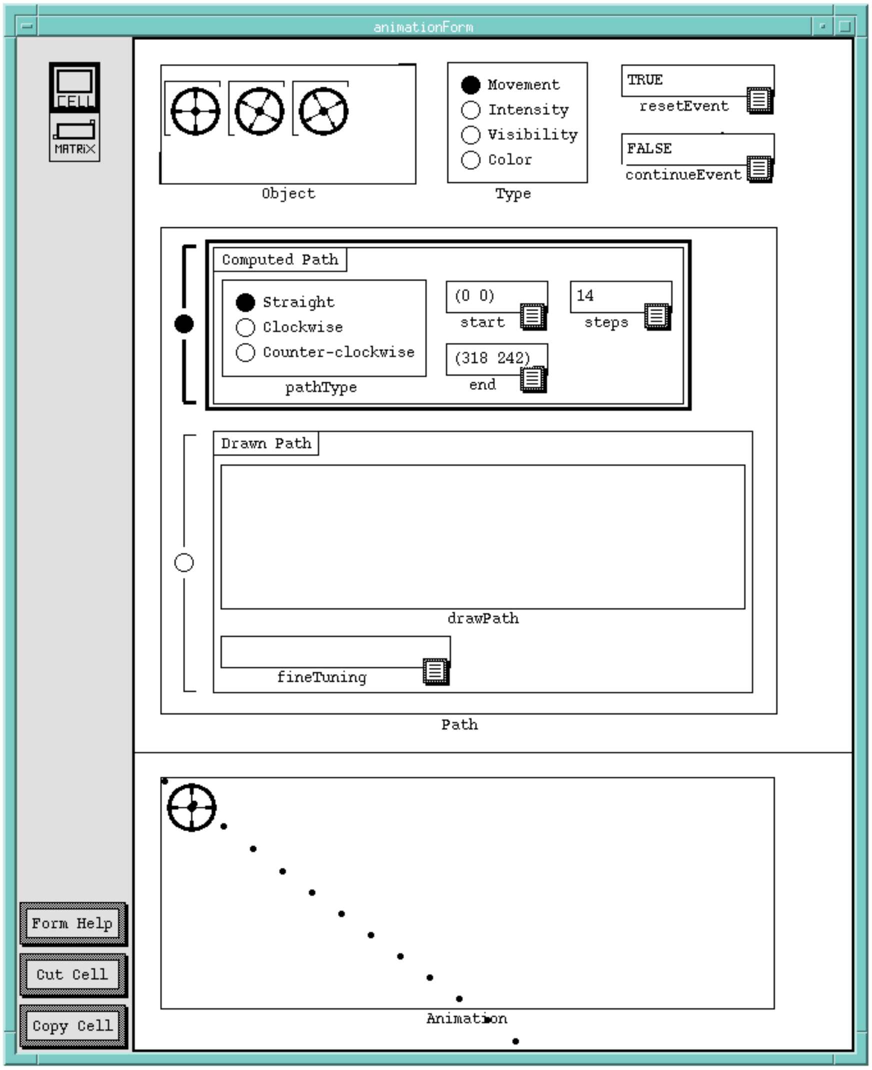

Figure 14 - The animationForm Window

3. Iconify all the eventReceptor forms.

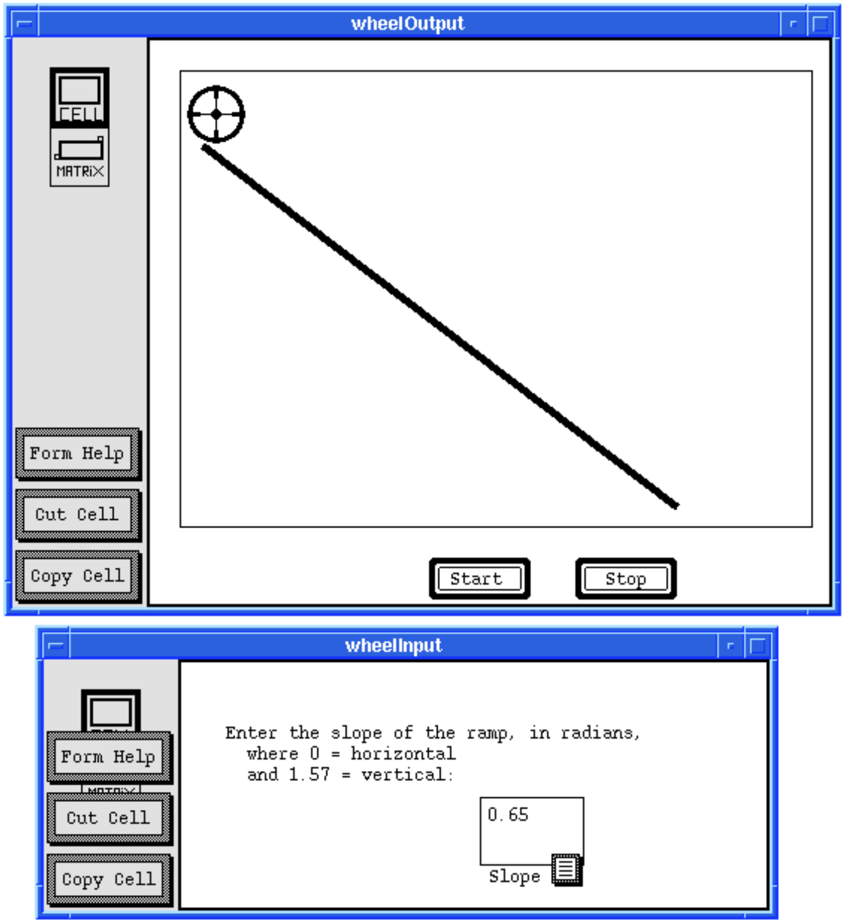

Figure 15 - The wheelInput and wheelOutput Forms

4. Start by watching the animation. Go to the Forms/3 main window and under the Forms loaded: select wheelOutput and click on the Display Form button. wheelOutput should now be the front window. Move it aside so that you can see both it and the Forms/3 main window at the same time. Animation is a time based operation, so start time on the Forms/3 main window. Now click on the Start button on the wheelOutput form and the wheel will move down the ramp until the you press the Stop button. Note that time has changed in the Forms/3 main window. In the Forms/3 main window use the triangular icon on the timeslider and drag the time to the left and watch the wheel retrace its route.

5. Move time to about 5, and change the slope on the wheelInput form to **0.80**. In Forms/3 an animation can be modified mid-execution and will give immediate feedback about the effects of those modifications. Note that the wheel's position and all related values have been updated to reflect the same point in time that the animation was at before the modification was made.

6. Any Forms/3 program can be executed in either a forward or backward direction. The direction can be toggled at any time by simply clicking the Stop Time, Reverse Time and Start Time buttons, permitting you to examine any portion of the execution as many times as desired without restarting the entire program. Click on Stop Time, Reverse Time, and Start Time now, and watch the wheel roll back up the ramp.

7. Stop Time and move the time slider back to time 1. Now move to the animationForm and click the button next to the Drawn Path. Using your mouse (while any mouse button is down inside the drawPath cell) drag a path in the drawPath window. When the mouse button is released, the list of offsets constituting the path is placed in the fineTuning cell, where you can make precise adjustments (0 0 is the upper left hand corner). Start the wheel moving on the path by following the sequence of events in step 4, above.

TIPS: THE ANIMATIONFORM

Animation is visually programmed in Forms/3 by specifying the animation parameters on this form (see Figure 14). The Object matrix, the Type, the resetEvent, the continueEvent, and the Path cells are the inputs to the animation, and the Animation cell at the bottom of the form renders the transition of the object along the path. The Object matrix contains the object(s) to be animated. An object can be any type, including primitive objects such as boxes, bitmaps, and strings. When the Object matrix contains more than one object, as in Figure 14, the animation repeatedly cycles through the matrix, displaying one object per animation frame. Here, the differences between the three glyphs of the Object matrix cause the wagon wheel to rotate as it moves down the ramp.

The Type cell contains the transition type of the animation. The programmer uses the radio buttons to select from the four primitive transition types.

Whenever the resetEvent cell is true, the animation is restarted from the beginning of the path. Whenever the continueEvent cell is true, the next animation frame in the transition is rendered in the Animation cell. The resetEvent cell has precedence over the continueEvent cell.

The value of the Path cell is the relative path that the animation follows. The programmer indicates, by selecting the appropriate radio button, whether the path is to be computed or drawn. More information can be obtained by referring to the animation - related references in Appendix D.

EXAMPLE 7 - ALGORITHM ANIMATION: A SORT ALGORITHM

OVERVIEW

Algorithm animation has been seamlessly integrated into Forms/3. This is not a special-purpose visual algorithm animation system for animating algorithms implemented in other languages, but rather an animation system for algorithms programmed in Forms/3.

DETAILED INSTRUCTIONS

1. From the Forms/3 main window load the form:

~burnett/Resrch/Forms/examples/animation/sort.matrix/matrixSort.package

A Sort form, four primitiveAnimation forms and a sortOutput appear. Iconify the four primitiveAnimation forms by clicking in the upper-menu bar in the second from the right hand button. This leaves the sortOutput, Sort and Forms/3 windows open. On the Forms/3 main window click on the Start Time button and watch the sorting process. Click on the Stop Time button at any time to observe values in cells or to navigate manually through time. More information can be obtained about this example and algorithm animation from the references in Appendix D.

APPENDIX A: HOW TO EXECUTE FORMS/3

• On your X-terminal start-up your window manager and in the login window type

xhost +hostname

(some computers will require that the xhost command be entered. For example **xhost +gold**. On some computers the xhost path needs to be typed in, e. g. /usr/local/X11/xhost).

• Open an X window and from it rlogin to the sapphire, grace, or gold computer.

• at the UNIX prompt in this new window type:

setenv DISPLAY your workstation:0
(for example setenv DISPLAY 128.193.99.99:0 or twiddle:0)

cd ~burnett/Resrch/Forms/src

• at the UNIX prompt type

GarnetForms

You will see some comments printed and then the > prompt.

• Upon receipt of the > prompt type

(load "RUN")

You will see several comments printed and then the > prompt.

• Upon receipt of the > prompt type

(run [:keyword option])

The keyword options are given in Appendix C.
Normally you won't use these options, and will just type: **(run)**

The Forms/3 main window should now appear.

APPENDIX B: SYNTAX FOR CELL FORMULAS

Type	Format	Examples, Explanation
Text	quotedString	“Hi” Any leading blanks before the initial quote mark will be removed before the text is displayed.
Algebraic Expressions	integer and floating point numbers are accepted.	• Operators include + - * / • The mod function is allowed with format: (numerator **mod** denominator). The numerator and denominator may be numbers or cell references. • Operators and numbers (variables) must be separated by at least one blank. 3*4 will cause an error. 43 43 + 57 (cell reference * (100 - 53))
Cell references	Click on the cell or type the name of the cell. Case is ignored.	This feature works similarly to a spreadsheet program.
Logical Expressions	A op B not A not (A and B). A and B are numbers, cell references, boolean expressions or more complex expressions	• A space must be placed between each expression and the operator. • The operators not, and, or are to be used for booleans variables. Their use on other types of data gives unpredictable results. • The operators =, <, >, <=, >= can be used with numbers • Starting expression with an equal sign (as in Excel) will return a true or false value.
Boolean Variables	t, f, true or false	
Box	box Width Height or Box may be capitalized.	• Example: box 20 (10 * 2) • A box Width by Height appears within the cell on the form.
Line	Line or line delta_x delta-y	• Example: line 20 30 • Displays a line from (0,0) to (50,60)
Width	Width (reference to a graphical object)	width X

Height	Height (reference to a graphical object)	height X
Glyph or Bitmap	glyph bitmap-filename	glyph “~/examples/fish.bm” Bitmaps can be created using X’s “bitmap” utility. Please give the full pathname for the bitmap file .
ComposeCentered	composeCentered X with Y	• ComposeCentered is one word. • Center Y over X horizontally and vertically. Example: ComposeCentered (box 20 200) with (box 10 10)
Compose	compose X at (x1,y1) with Y at (x2,y2) with Z at (x3, y3) with ...	Compose one object at a certain location, or compose two or more things together. Examples: compose box1 at (10, 10) compose (box 10, 10) at (50, 50) with (circle 8) at (25, 25) with “hey” at (15, 15).
if/then/else	if (cond) then E1 else E2	if (N > 0) then 1 else (N = 2)
fby	0 fby 1 Count fby 1 until done	Initial value of 0, followed by 1 at the next moment in time Initial value of Count, followed by 1 until done
earlier	earlier (N + 1) initially 0 until (countDone) earlier (N + 1) initially 0 earlier (N + 1)	Suppose this is the formula for cell N. Then N's initial value is 0, and then increments by 1 over time until countDone is true
prev	prev (N + 1) initially 0 until (countDone) prev (N + 1) initially 0 prev (N + 1)	

Misc.	sin x	(x is in radians)
	cos x	(x is in radians)
	round x	round (12/5)
	-number	(this is unary minus) e.g., -3 + X
	+number	(this is unary plus) e. g., +3 - X

APPENDIX C: KEYWORD OPTIONS FOR THE (RUN) COMMAND

(run [:keyword option])

Keyword	Comment	Options
dm	Direct manipulation	T - point/click cell reference and text entry (DEFAULT) NIL - Text entry only
bigfont	Big font used in the video tape	NIL - default font (DEFAULT)
memoryhog	Memory intensive runs. Bumps Lisp's memory from 40MB to 100 MB	NIL- 100 MB (DEFAULT) t - 40MB
timeadvance	Time moves forward (or backward); one logical second approximately every physical second	T - Time starts moving forward immediately. NIL - You can move time manually using the time slider, or you can start the automatic movement at any time. To start time, enter (inter:start-animator $TimeAdvance). To stop time, enter (inter:stop-animator $TimeForward nil) To make time run backwards, enter (setf $TimeForward nil). NOTE: If time is running backwards and it gets to time ' (1), it will automatically reverse and run forwards.

APPENDIX D: FORMS/3 REFERENCES

DISSERTATIONS

1. "Abstraction in the demand-driven, temporal-assignment, visual language model", Margaret Burnett, PhD Dissertation, University of Kansas, 1991.

JOURNAL PAPERS

2. "Interactive Visual Data Abstraction in a Declarative Visual Programming Language", Margaret Burnett and Allen Ambler, *Journal of Visual Languages and Computing* 5(1), 29-60, March 1994.
3. "Visual Forms of Iteration that Preserve Single Assignment", Allen Ambler and Margaret Burnett, *Journal of Visual Languages and Computing*, Vol. 1 No. 2, Academic Press, 159-181, June 1990.

CONFERENCE PAPERS

4. "Integrating Algorithm Animation into a Declarative Visual Programming Language", Paul Carlson and Margaret M. Burnett, 1995 IEEE Symposium on Visual Languages, Darmstadt, Germany, September 1995.
5. "From Concrete Forms to Generalized Abstractions through Perspective-Oriented Analysis Of Logical Relationships", Sherry Yang and Margaret Burnett, 1994 IEEE Symposium on Visual Languages, St. Louis, Missouri, 6-14, October 4-7, 1994
6. "Types and Type Inference in a Visual Programming Language", Margaret Burnett, 1993 IEEE Symposium on Visual Languages, Bergen, Norway, 238-243, August 24-27, 1993.
7. "Is It Easier to Write Matrix Manipulation Programs Visually or Textually? An Empirical Study", Rajeev Pandey and Margaret Burnett, 1993 IEEE Symposium on Visual Languages, Bergen, Norway, 344-351, August 24-27, 1993.
8. "A Declarative Approach to Event-Handling in Visual Programming Languages", Margaret Burnett and Allen Ambler, 1992 IEEE Workshop on Visual Languages, Seattle, IEEE Computer Society, 34-40, September 1992.
9. "Generalizing Event Detection and Response in Visual Programming Languages", Margaret Burnett and Allen Ambler, Advanced Visual Interfaces: an International Workshop, Rome, Italy, May 27-29, 1992, World Scientific Series in Computer Science 36, (T. Catarci, M. F. Costabile, S. Levialdi, eds.), World Scientific Press, Singapore, 334-347, 1993.
10. "Efficiency Issues in a Class of Visual Languages", Margaret Burnett and Allen Ambler, 1990 IEEE Workshop on Visual Languages, Skokie, Illinois, IEEE Computer Society, 209-214, October 1990.
11. "Visual Languages and the Conflict Between Single Assignment and Iteration", Allen Ambler and Margaret Burnett, 1989 IEEE Workshop on Visual Languages, Rome, Italy, IEEE Computer Society, 138-143, October 1989.

VIDEOS

12. "A Taste of Forms/3", Margaret Burnett, Sherry Yang, and John Atwood, 13-minute video, July 1994

TECHNICAL REPORTS

13. "Algorithm Animation in a Declarative Visual Programming Language", Paul Carlson and Margaret M. Burnett, Technical Report 95-60-2, April 1995.
14. "Exception Handling in the Visual Programming Language Forms/3", Pieter van Zee and Margaret M. Burnett, Technical Report 95-60-1, Oregon State University, Department of Computer Science, March 1995.
15. "Lazy Marking: A Lazier Implementation of Functional I/O for Graphical User Interfaces", Margaret Burnett and John Atwood, Technical Report 94-60-9, Oregon State University, Department of Computer Science, December 1994.