FormPlus: A Form Authoring Toolkit

Ting-Jun Fan, Rona S. Machlin, Christopher P. Wang, and Ifay F. Chang

IBM T. J. Watson Research Center, P. O. Box 704, Yorktown Heights, NY 10598

Abstract

In this paper, we present the design and implementation of FormPlus, a form authoring toolkit for on-screen form design and filling. It is implemented on an IBM OS/2 Presentation Manager platform. FormPlus provides major front-end functions for form processing such as converting paper forms into electronic forms, designing forms, and filling in forms. FormPlus users live in a very friendly, WYSIWYG (What You See Is What You Get) environment, no knowledge about programming is necessary. FormPlus also provides an object-oriented API (Application Programming Interface) for programmers to develop their own applications with minimum effort. FormPlus is designed to be easily extendable to incorporate new objects and to allow data import and export.

1 Introduction

1.1 Why Electronic Forms?

Forms have become prevalent in all sectors of business and industry. They have emerged with two key properties. One is that they have become the familiar, taken-for-granted interface to business operations. The second property is that they have become a critical link between business processes and in computer integration of business applications.

A well-designed form should be easy to fill in, read, process, store, and distribute. Furthermore, the instructions for filling in the form should be easy to locate, understand, and follow. Electronic (on-screen) forms can provide better support for these tasks than traditional paper forms can. First, in electronic forms, filling instructions can be clearly displayed where needed and provided on demand. For example, illustrations relevant to a field being filled in can be brought up and displayed near the field by simply pressing a help key. With a traditional paper form, finding the relevant illustration might require a manual search through some hardcopy (paper) information resource containing extraneous information. Second, electronic forms can be filled in more efficiently and accurately. Most hardcopy forms have a few fields whose values come from some long calculation (e.g., the total price of a purchase order). This calculation is usually a bottleneck of the form-filling process. In electronic forms, the computer can automatically derive the values for such fields, so that the user need not calculate them, thus improving both efficiency and accuracy. Electronic forms can also improve accuracy by performing error detection as the user is filling in the form. Then whenever a user fills in a value that violates some constraint, the user can be informed immediately so that he can correct the error immediately. Finally, electronic forms, either blank or filled, can be efficiently distributed through electronic networks.

Two separate tools are necessary to support the capabilities just described: one for form design and one for form filling. The form design tool lets a designer create on-screen forms while the form filling tool lets a user fill information into a form. The form design tool should provide enough functionality to let the designer create various fields, arrange layouts on the screen, duplicate and imitate existing paper forms, specify actions such as the help instructions, etc. After a form has been created, it will be distributed or made available to various users, who can then use the form filling tool to bring up the designed form, fill in information, and save the filled form. These two tools should be closely related to each other because many of the functions, such as displaying the created fields, are the same for both tools; however, they have to be completely independent because they serve two different kinds of users. In fact, the form design tool should contain the form filling tool because the designer, in order to test the form being created, should be able to do all the things that a form filling user is allowed to do. Therefore, a good form design tool should allow the designer to switch back and forth between the designer mode, in which he is allowed to do things that are not permitted in the form filling tool, such as delete a created field, change the color of a field, or move a field around the screen, and the user mode, in which the designer acts as if he were a form filling user.

Form design, similar to graphical user interface applications, can be easier and more efficient if the designer can immediately view what he has created and modify it interactively. In order to achieve this, the form design tool is taking advantage of workstation displays and window interfaces rather than using conventional line-coded programming. In a window-driven environment, the designer can communicate with the underlining program through a mouse and keyboard. The created fields can be displayed immediately on the screen and the designer can move, resize, or rotate them through direct manipulation without the need to know the exact attribute values associated with these fields. Thus the form can be easily modified interactively by the designer without his having to write any programs.

1.2 FormPlus --- A Form Authoring Toolkit

In this paper, we present the design and implementation of FormPlus, a form authoring toolkit for on-screen form design and filling. Though FormPlus is primarily a tool for designing and filling forms, it can also be viewed as a graphics editor that can provide very high quality graphics output, as well as a toolkit that facilitates application prototyping: forms designed in FormPlus can be used as input screens to other applications. This latter use of FormPlus will be described in more detail later when FormPlus Application Programming Interfaces (API's) are discussed.

1.2.1 Functions Provided by FormPlus
The entire form process includes many different steps which can be separated into two general categories: the front-end, in which tasks are usually done at the workstation side, and the back-end, in which tasks are usually done at the mainframe. The purpose of FormPlus is to provide most of the front-end functions which include form design, filling, retrieving, storing, and printing. Figure 1 shows the roles of FormPlus in the entire form process:

1. Form design: on-screen design in a user-friendly environment.
2. Form conversion: conversion of paper forms into electronic forms.
3. Form storing: database support for form storing and retrieving.
4. Printing: printing on different sizes and using different color choices.
5. Form filling: on-screen filling supported by:
   a. Validation checks: automatic checking of input fields.
   b. Computation: automatic computation of field data.

1.2.2 Design Considerations

FormPlus not only has to support easy-to-use front-end functions such as form creation and filling, but also has to provide easy access to forms for back-end form filling and processing; i.e., use of these forms. Furthermore, FormPlus must easily integrate its forms with other business processes.

Because of these considerations, FormPlus was created as both a stand-alone application that provides a programming-free environment for front-end form processing and also a set of API's (Application Programming Interfaces) that support further application development and integration with other business processes. The API's are provided in two levels. The higher level API's supply functions for form access such as retrieving and storing. These functions enable the back-end programs to be able to reach and process the forms created in the front-end. The lower level API's consist of functions for individual object manipulations.

In order to support its generality, ease extensibility, and easy-to-use programming interfaces, FormPlus was designed with an object-oriented approach. Each feature in FormPlus is represented by an object. These objects include graphic objects, bitmaps, entry fields, tables, etc. Each object type is supported by its own methods. The methods are standard operations to create, delete, copy, rotate, resize, change attributes of, store, and retrieve the objects. The methods are invoked by sending messages to a top-level routine. This routine then dispatches controls according to the object types and the messages received.

In keeping with this object-oriented design, the lower level FormPlus API's allow programmers to take advantage of the already designed objects and their methods. By calling the lower-level API functions for a particular object, one gets the default methods designed into FormPlus to create, delete, copy, or paint an object of that type. This saves a tremendous amount of work in writing routines for these methods from scratch. One can still write one's own applications to decide how to handle the user interactions that initiate these methods and how to use the objects. Of course, a programmer can also modify or replace any default method by capturing the corresponding message before it is sent to the top-level routine and redirecting it to a routine he himself has written.

2 Survey of Related Work

PerFORM [1] is an electronic business form processor running on the IBM PC, XT, AT, PS/2 or compatibles under the DOS operating system. It is easy to learn and easy to use and is based on a WYSIWYG (What You See Is What You Get) interface. It provides a form design tool and a form filling tool. PerFORM provides a programming-free environment for the designer to design forms using the mouse or keyboard, and also allows people to scan existing forms and trace them on the screen. In its form filling tool, PerFORM provides data entry validation including numeric checks and range selection (which have to be specified in the form design tool), context sensitive help, automatic date and time stamping, and automatic calculation from other fields.

InterView [2] is a toolkit developed at Stanford University that offers a rich set of composition mechanisms and a variety of predefined objects, which make it easy to implement complex user interfaces. It supports three basic composition mechanisms: Interactor composition, which allows interactors (i.e., managers of some area of potential input and output on a workstation display) to be composed, providing for tiling, overlapping, stacking, and encapsulation of object components, Graphic composition, which provides a direct manipulation editor that allows the designer to directly manipulate graphical representations of familiar objects, and Text composition, which allows the designer to directly manipulate text objects and textual interfaces. InterView is a general application development toolkit which is not restricted to form design alone.

There are many other areas of research related to our work [3-6]. In general, a general purpose user interface design tool such as InterView and HyperCard is not restricted to form design; therefore, many of the special fields such as editable tables or multi-line entry fields which are very useful for form design are not included. On the other hand, systems focusing on form design, such as PerFORM, do provide many features specially useful for form design; but not provide programming interfaces to facilitate convenient use of their functions to develop form based or related applications.

3 FormPlus

3.1 System Overview

FormPlus is implemented with Presentation Manager (PM) and OS/2 and operable on a PS/2 model 60 or above with a VGA or BGA display. It is a stand-alone application requiring no pre-installation to run. FormPlus itself serves as a user interface between the PM window system and the form designer. A form generated by FormPlus is itself an application which is also a user interface between PM and the one who fills in forms.

Figure 2 shows the system structure of FormPlus which contains two modes. The designer mode lets the designer create on-screen forms, and the user mode lets the user fill in information in the designed form. The user mode also allows the designer to test the form being de-
signed. The designer is free to switch between these two modes; however, the form end user is allowed only in the user mode.

3.2 Objects And Forms

In FormPlus, a form is a combination of various objects. Each object has different properties and provides different functionality. FormPlus provides five types of objects: graphics objects (including circles, ellipses, rectangles, triangles, diamonds, lines, arrows, text, and barcodes), bitmaps (or images), checkboxes, entry fields, and tables.

One can start designing forms by creating objects in the designer mode. Each object created can then be treated independently and can be modified at any time during the design process. Several objects can be grouped as one object. The designed form can be stored, retrieved, modified, printed, or filed in by a user in the user mode. Figure 3 shows a typical form created by FormPlus. It should be noted that this form was originally created in full color and can be displayed on a color monitor and printed on a color printer. The form contains two bitmaps (one, a picture of fruits and the other, an image of teeth), a barcode, a table, a frame rectangle and several other smaller ones, 11 checkboxes, many lines of text (using different fonts and background colors), and many entry fields. A user can check/uncheck any of the checkboxes by clicking a mouse button on top of it. A user can also fill information into any one of the entry fields by clicking a mouse button on top of it and then typing characters at the keyboard. Some of the entry fields such as those labeled "Serial" and "Date of Accident" are assigned particular types which cause automatic input validations to be performed each time their contents are changed by the user. The entry field prefixed by "Total fee charged" at the bottom of the form is assigned a numerical formula, specifying that its value should be the sum of the contents of the last column (i.e., the entries under "Fee"). In general, when an entry field is assigned a formula, the formula is used to automatically calculate a value for that entry field. Whenever the value of any field mentioned in the formula is changed, the result of the formula is recalculated and the entry field assigned the formula is updated to reflect the new result. So in this case, the field labeled "Total fee charged" will always contain the sum of the "Fee" column, and whenever any value in that column is changed, the contents of this entry field will be updated to the new sum of values in the "Fee" column.

3.3 User Friendly Interfaces

FormPlus provides a completely programming-free WYSIWYG (What You See Is What You Get) environment in which one can easily and efficiently create and fill in on-screen forms. Everything on the FormPlus main window can be manipulated directly using the mouse and keyboard or indirectly via menus and dialogs. Figure 4 shows a FormPlus main window.

3.3.1 Menus and Dialogs

Displayed at the top of the main window is the name of each menu bar. Each menu bar consists of several menu items, each of which is labeled by one or two words describing the action or sequence of actions which can be invoked by selecting this menu item. When an action requires more than one step, a dialog box is usually used. For example, changing the background color of the client window requires at least two steps: one to select the menu item initiating this action and one to choose the new background color.
3.3.2 Designer Mode and User Mode

Since the two modes serve different users, their functionalities are quite different. In general, the user mode is more restricted than the designer mode. For example, a form user is only allowed to fill in or change data in certain areas and is not allowed to create, delete, resize, or rotate any of the objects in the form. FormPlus is designed to provide two software copies: one for the designer which will contain both modes and one for the form user which will contain only the user mode. The designer's copy will allow the form designer to switch between two modes via a single menu selection.

3.3.3 Object Manipulation

Manipulating objects in FormPlus is very easy. Once an object is created, it acts like a solid object that one can drag, resize, rotate, cut, or duplicate with simple mouse and keyboard manipulations. The results of such actions are animated and displayed immediately on the screen. For example, an object can be selected as the current object by clicking anywhere inside the bounding rectangle of this object. The bounding rectangle of an object is the smallest rectangle that encloses the object. The current object is always highlighted and can be moved, resized, edited, etc. For example, to rotate an object, one should follow the steps listed below:

1. Select that object.
2. Choose the "Rotate" menu item under the "Arrange" menu bar.
3. Click the left mouse button anywhere on the client window to specify the center point of the rotation, or click the right mouse button, in which case the center of the object will be chosen as the rotation center. (Two concentric circles will be displayed.)
4. Click and hold the left mouse button between the two circles and drag the mouse. (The circles will revolve and the boundary of the object will be rotated accordingly, as shown in Figure 5.)
5. Release the mouse button, the circles will disappear and the object will then be rotated accordingly.

![Figure 5. Rotating an object.](image)

3.4 Editing Capabilities in FormPlus

3.4.1 The Entry Field

Entry fields are small text editors. As with the other objects, designers can interactively create them, move them, resize them, etc. In addition, designers, as well as users, can enter texts into entry fields. This can be done by first activating an entry field by clicking a mouse button on it and then typing at the keyboard. When an entry field is active, a cursor is displayed to show the current character position. Any text typed in will be inserted at that position. In FormPlus, entry fields are supported by strong editing capabilities which include formatting and reformattting of text, input validation checks, and automatic formula computation.

Formatting and Reformatting

As text is entered into an entry field, it is automatically reformatted to prevent it from going outside the entry field in the horizontal direction. In doing this, FormPlus tries not to split a word over different lines except when an entered word is longer than the entry field width.

When the text in a line becomes longer than the entry field width, this line is split at word boundaries. This is, the line is truncated word by word starting from the last word until it fits in the window, and the truncated words are moved down to the next line. Then the next line is reformatted, then the line after that, etc. When a character is deleted from the current line, the first word of the next line may now fit at the end of the current line. If so, this word is moved up to the end of the current line, and then reformattting is propagated through the following lines. However, if the user hits an "Enter" key, a carriage return is inserted, and a new line is always enforced.

Input Validation

FormPlus provides automatic type validation of entry field input. After creating an entry field, the designer can specify the entry field type. Once a particular type has been assigned to an entry field, the contents of the entry field will be checked automatically every time the user finishes typing into that field. If the contents violate the defined type, the user will be warned and forced to go back to the entry field until he enters data of the right type. For example, an entry field with type "date" can only contain a string representing a valid date; however, the format of the date is very flexible. For instance, to input a date indicating the 31st of October, 1989, one can type "October 31, 1989", "10-31-89", "Oct. 31 1989", "10/31/89", etc. Dates entered are also checked for valid month-date and leap year relationships. For example, the dates "02-30-89" and "02-29-89" are both illegal, but "02-29-88" is acceptable since 1988 was a leap year.

Automatic Computation

FormPlus also provides automatic computation of entry field contents by allowing the designer to specify an expression, or a formula, for any numeric entry field. The formula can be any numeric expression that contains numbers, operators (+, -, *, /, ^, (, )), and symbolic names representing other entry fields. After a formula is assigned to an entry field, the contents of the entry field will be equal to the result of this formula. If the formula contains a symbolic name of another entry field, the result of the formula will be updated automatically each time the contents of that entry field are changed.

![Figure 6. Editing a formula.](image)
placed into entry field "D". We can type a formula into the dialog box as shown in the figure:

\[ A + B + C \]

Then, whenever the content of any of the other three entry fields is changed, the given sum is automatically recomputed and the result placed into "D".

If one would type something illegal, for example, "A + E" where "E" did not exist, one would be prompted for a correction. If any of the entry fields in the formula was not numeric, one would be warned and asked if forcing it to be numeric would be desired. It should also be noted that, in the above example, "D" could not appear in the formula of "D" itself for obvious reasons.

3.4.2 The Table

Tables are structured, higher-level objects provided to expedite form design and filling. They contain entry fields organized into rows and columns. Each column has a column header and the entire table has a title. The designer can easily add, delete, resize, or move columns and add, delete, or move rows through simple direct mouse manipulations. A user can easily enter data into the table fields, by moving back and forth across rows or up and down columns.

Figure 7 shows an example of how to move a table column. Suppose one has a table with four columns as shown in Figure 7 (a) and one wants to switch Column 2 and Column 3. To accomplish this, one can take the following steps:

1. Click and hold the left mouse button on the double line below the column 2 header. (A tracking rectangle will appear around the column, as shown in Figure 7 (b)).
2. Move the mouse rightward while holding the button down until the tracking rectangle sits between column 3 and column 4, as shown in Figure 7 (c).
3. Release the mouse button, column 2 will be moved between columns 3 and 4, and column 3 will be moved down to where column 2 was, in effect switching columns 2 and 3, as shown in Figure 7 (d).

The description so far has assumed that the entire table is always visible. In many form based applications, it is not unusual for tables containing database data to have hundreds of rows. In fact, tables designed with FormPlus are rarely entirely visible on the screen either limited by the display screen size or dictated by the form layout. In these situations, a view onto the table is shown. A view contains some block of currently visible rows and columns, along with the table title and the column headers for the visible columns.

When the table is activated, vertical and horizontal scroll bars are provided to scroll the view. Scrolling the view horizontally moves different columns into and out of view. When the view is scrolled horizontally, the column headers move along with the columns, but the table title stays in place. Scrolling the view vertically moves different rows into and out of view. With vertical scrolling, the column headers stay in place, along with the table title. This is a desirable feature, since in general, it is helpful to be able to see the appropriate column headers no matter where one is in the table. The screen displays in Figure 8 illustrates the fully functional scrolling: clicking on the up-down or left-right arrows with the mouse scrolls the view up-down or left-right one column; moving the scroll bar thumbwheels with the mouse scrolls the view proportionally in the appropriate direction, in real time, clicking on the scroll bar to the left-right or to the top-bottom of the thumbwheels scrolls the view left-right by one page or up-down by one page. Also, when entering data, tabbing beyond the current view (with either the tab/back-tab keys or the up arrow/down arrow keys) causes the view to automatically scroll in the appropriate direction. This makes it very easy to enter an entire row or column of data.

When the designer changes the table by adding/deleting/resizing columns/rows, the view size remains constant if possible (sometimes the view size must be changed in order to keep the table consistent). The designer can change the view size by choosing "Resize" under the "Arrange" menu bar. When he clicks and holds the mouse button, a tracking rectangle will appear, and as he holds down and moves the mouse, the tracking rectangle will resize to show the current view size. Releasing the mouse button will change the view size appropriately. Note that the view size cannot get larger than the full table size, nor smaller than the size necessary to show one row of the widest column. Also, the view will always contain the table title and appropriate headers.

Figure 7. Changing table column order.

(a) Original
(b) Click at double lines of Col. 2
(c) Move the tracking rectangle
(d) Result

Figure 8. An example of scrolling.

(a) A large table before vertical scrolling
(b) The table in (a) after vertical scrolling

<table>
<thead>
<tr>
<th>Personal Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
</tr>
<tr>
<td>106</td>
</tr>
<tr>
<td>107</td>
</tr>
<tr>
<td>108</td>
</tr>
<tr>
<td>109</td>
</tr>
<tr>
<td>110</td>
</tr>
<tr>
<td>111</td>
</tr>
<tr>
<td>112</td>
</tr>
</tbody>
</table>
3.5 Creating Forms from Scanned-In Paper Form Images

In addition to creating forms from scratch, one can also use FormPlus to create electronic forms from scanned images of existing paper forms. This function provides a smooth transition from the paper form processing to the complete electronic form processing in a friendly way.

The scanned images are treated as bitmaps in FormPlus and can serve as backgrounds to the electronic forms to be created. Though scanned paper form images sometimes look just like forms designed from scratch, no data can be entered into them -- their checkboxes and entry fields are still just images. In order to be able to fill data into these forms, areas into which the user will input data must be converted to active fields. This can be done by creating checkboxes, entry fields, and tables and superimposing them on top of the appropriate areas of the image. Of course, parts of the paper image containing descriptive text, lines, and bitmaps can be left as its familiar images, which simplifies the creation process. Figure 9 (a) and (b) show an example of a paper form scanned in and an electronic one converted from the form image, respectively.

It should be noted that, as shown in Figure 9 (a) and (b), due to noise and resolution problems introduced by the scanning process, a form created from a scanned paper form image may not appear as clean or crisp as one generated completely from scratch. However, from the application point of view it is often desirable to retain the familiar paper form image in the electronic form until ready for complete conversion or redesign to electronic form for automation.

![Scanned paper form image](image1)

![Electronic form created based on the form image](image2)

4 Conclusion

In summary, we have implemented a toolkit, FormPlus, which provides an easily usable environment for electronic form design and form filling. FormPlus has the following major capabilities that make electronic forms more advantageous than conventional paper forms, and yet complementary to the existing paper form processing environment:

1. It can be used to convert paper forms into electronic forms, by loading in scanned-in form images and creating data entry fields wherever data will be entered. This process is extremely simple, since only a few objects need to be created. Also, any form created this way will be familiar, since most of the image of the original paper form is retained.

2. It provides a programming-free environment in which a designer can easily create complex forms, and a user can easily fill them in, through simple, direct manipulations using just the mouse and keyboard.

3. It provides strong editing capabilities for fields, including input validation and automatic formula computation.

The philosophy in the design of FormPlus bears the automations of form processing in mind. The system is designed to be easily extendable and to be the base for integration with many applications.

Reference


