What's NeXT After 1-2-3?

Lotus's spreadsheet for the nineties, Improv, has a startling new look

Tom Yager

You know the old saying about reinventing the wheel. And the one about not fixing it if it isn't broken. Both of those philosophies seem to apply to spreadsheets because practically every one that has been produced in the last several years looks and acts like Lotus 1-2-3. Can it be that there is simply no other way to make a spreadsheet?

Lotus doesn't think so. Improv is a completely redesigned spreadsheet program that borrows nothing from past Lotus efforts. Nearly everything about it is new to Lotus, and some elements of the interface have never been used in *any* program. That's what's so exciting about it. In these days of lawsuits over duplicated look and feel, Lotus has come up with something *new*.

B2, or Not B2

To call a program a spreadsheet, it must have rows and columns; that is the only sensible way to arrange tabular data. Every spreadsheet has this much in common, but Improv diverges rapidly from the pack after that.

The most obvious difference is that Improv runs exclusively, at least for the time being, on the NeXT Computer. Lotus had its own reasons for selecting that platform, not the least of which is the killer development environment that comes standard on the NeXT.

Lotus plans to port Improv to other environments, but the first such target has not yet been chosen. The package is entirely dependent on a graphical interface, so Windows, the X Window System, and the Macintosh are all likely candidates. Shipments to customers are scheduled to start late this year.

Once you get over the shock of having to buy a NeXT to run Improv, you begin to notice the important things that make it different. First, the stifling ABC/123 row and column names are dead. You can name rows and columns whatever you like. Big deal, right? Ah, be patient—that's only the beginning.

In Improv, row and column names are not just ornaments; they are identifiers, which Lotus calls *items*. To locate a specific cell in an Improv worksheet, you specify the intersecting items (e.g., `March:Profit`). For clarity, that beats the heck out of C21. Item names can contain mixed capitalization, even spaces and punctuation.

To add clarity and utility, you can assign types to items through *categories*. These are brief, usually one-word, descriptions of the type of information represented by the items. For example, if you had row items representing months of the year, and column items named for the things you sell, you might define categories of `Months` and `Products`. But you could tag them anything you like. Improv doesn't care what names you use.

Now, look at photo 1. This is an Improv window with a somewhat typical worksheet. You'll notice that the items appear where that ABC/123 stuff used to be, and categories appear on *tiles* placed near the rows and columns they describe. Improv starts up with one category each for rows and columns, but that might not be enough. In the Unix benchmark example in photo 1, I needed two sets of column categories: `Results` to hold the timing and index data, and `System` to hold the system names.

In 1-2-3, it might have been simplest to keep typing "Result" and "Index" across the columns until I had as many as I needed, but Improv won't allow that. Because it uses row and column names to identify cells, the names must be unique. That might seem like a burden, but it isn't. If you have repetitive data, you need to create a new category to contain it. That's easy enough: Click on an existing category tile and press Enter on the numeric keypad (there's also a menu option).

With the `Results` column category, each cell in photo 1's worksheet is distinctively identifiable, even though there are three sets of columns with the same names. To reference the C compiler index for the NeXT, you'd ask for `NeXT:C Compiler:Index`. As you add categories, the number of items needed to refer to a specific cell increases. For this reason, it's best to avoid using additional categories simply to group data for readability.

Improv provides a separate mechanism for this, called (of course) the *group*. Any number of items can be collected into a group, and group names can be used wherever a range is called for. Ranges still exist, and they are specified in a 1-2-3-like fashion with ".." between the bounding item names.

Getting (Data) into It

When you start Improv, you get a one-cell spreadsheet. Improv requires that

continued

THE FACTS

Improv $695
Lotus Development Corp.
55 Cambridge Pkwy.
Cambridge, MA 02142
(617) 577-8500
Inquiry 1185.

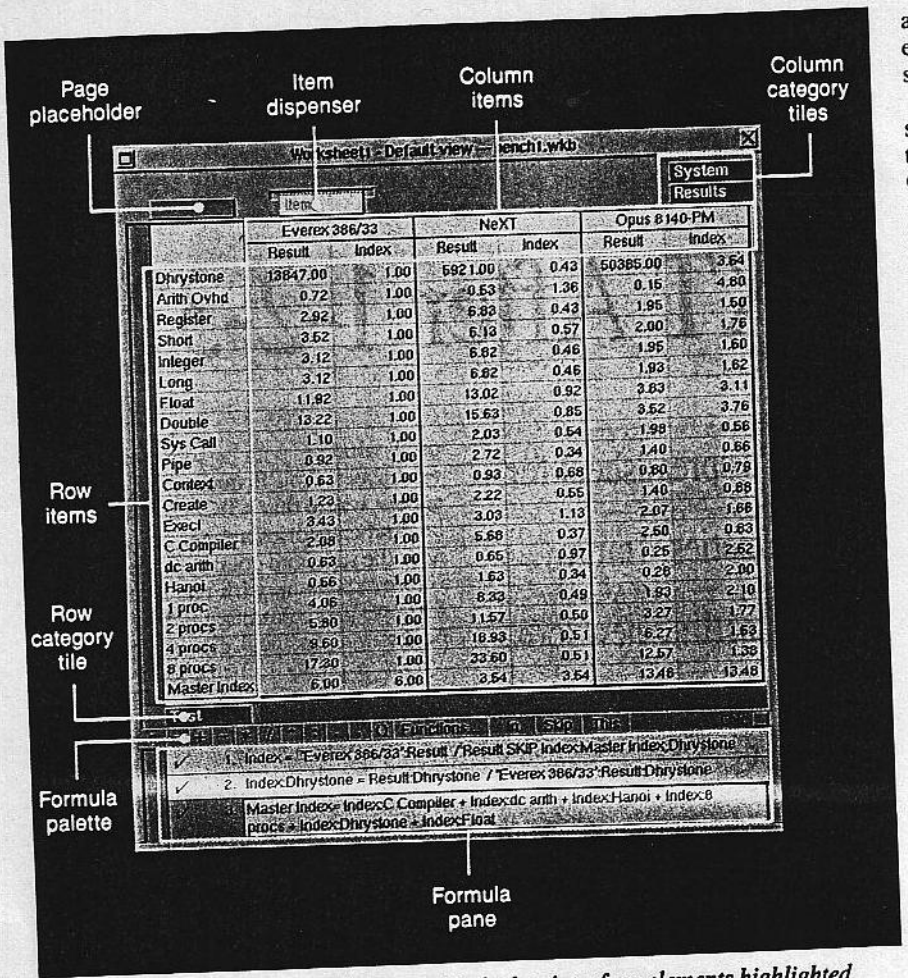

Photo 1: *A sample Improv worksheet with the key interface elements highlighted.*

you create cells as you need them. This is a good idea—it cuts down on the clutter—but adding cells as you go takes some getting used to. The keypad Enter key, as described above, is the global "make another one of whatever is selected" key, and it creates new cells, too. Cells are not added one at a time, of course. With the benchmark worksheet, I created all the column categories and items first. Then, stepping down the rows with the Enter key, I created and named the row items. Each time a new item is added, empty cells are created.

For those who are in a hurry, Improv offers two shortcuts to the creation of new cells. First, if you highlight an item and start drumming on the Enter key, it will create new items (and, therefore, cells) named after the parent category. If the category is named `Month`, the default item names will be `Month1`, `Month2`, and so on.

The other approach is more fun. At the upper left of the window lies the *item dispenser*, a nifty device that works like a roll of paper towels. You select an item, click on the "towel" sticking out of the item dispenser, and drag the mouse down until you have the number of items you want. You then tear off the sheet of items and drop the items on the worksheet. This is illustrative of Improv's interface philosophy: The most common operations are the ones that are the easiest to perform.

Facing the Interface

When you sit down with Improv, you must be prepared to be completely lost for a few minutes. There is no 1-2-3 interface compatibility—no menu comes up when you press "/." But there is a simplicity to it, and a continuity that makes it quick to learn and endearing to use. The interface is simple enough that, once you understand the hierarchy of categories, groups, items, and cells, the rest follows naturally.

Nowhere is the uniqueness of Improv's interface more apparent than in *flexible views*. Take a moment to study the layout of the data in the worksheet in photo 1. It is laid out properly for rapid data entry, and you can see the bottom line clearly enough, but it takes time to compare results between systems.

In an ordinary spreadsheet, you could set aside a group of cells below the main table to present the data more clearly. Getting the data into those cells would involve tedious copying or the creation of formulas that referred back to the original data.

In Improv, the worksheet in photo 2 was created with a single action: drag and drop. By moving the `Systems` category tile from the row to the column area, I instantly created a revised view of the worksheet. Systems got stacked together in adjoining rows, making comparisons a breeze. The data doesn't change, and you can shuffle the category tiles any way you like. If you drop a category tile over the page placeholder, Improv will devote one display page to each item under that category.

The data hierarchy in Improv doesn't end with pages. Each worksheet can hold multiple *views*, where each view presents the data differently. A single Improv file (called a *model*) can hold multiple worksheets (with unique data) as well. Cells in other worksheets (within the same model) can be referenced in any formula by prefixing the cell address with the worksheet name followed by two colons (e.g., `Profit` and `Loss::FY 90:Q1:Net Income`).

The Secret Formula

There's really only one reason to reference a worksheet cell or group of cells, and that is to perform some calculation on them. Lotus innovated again with respect to formulas. In 1-2-3, the norm is to define one formula in each cell that needs calculating. It is not unusual to find large spreadsheets with hundreds of formulas, woven in a fragile web of interdependence.

Improv breaks the link between cells and formulas. Formulas are entered in the *formula pane*, a resizable subwindow of the worksheet window. Each formula consists of a right side that describes the calculation, and a left side that specifies the placement of the results. I say "results" because a single formula can produce and place multiple data values in a worksheet. If you create a worksheet with items `Cost`, `Quantity`, and `Total`, the formula `Total=Quantity*Cost` will calculate *all* the totals in one operation.

Entering formulas almost never requires using the keyboard. Double-clicking in the formula pane places you in edit mode and brings up a palette of formula construction buttons. Clicking in the

worksheet adds the appropriate identifier to the formula. Category, group, and item names can all be used to select data for formulas.

Erroneous formulas are trapped immediately, and an explanatory error message that briefly describes the specific problem is displayed. A mouse-click brings up a dialog box with a detailed description of the error and a pointer to the location in the formula where the calculation failed.

Formulas can also overlap, placing their results in the same cells. Improv flags these, too, and, instead of blindly following the "last formula takes precedence" rule, it lets you select which formula will prevail.

The 1-2-3 functions are duplicated in Improv, and 1-2-3 spreadsheets can be imported to and exported from Improv. When importing spreadsheets, you can determine whether label names will be converted to items. Also, Improv will filter through the formulas, reducing them to the minimum set required to derive the desired result. The reduction varies, but most large spreadsheets, loaded with copied formulas, can be duplicated in Improv with only a handful of formulas.

Another important factor lending to the reduction in the number of formulas is the *recurrence formula*. A single expression can replace countless copied formulas. `Quarter[THIS]:Cost=Quarter[PREV]:Cost*1.25`, for example, escalates costs by 25 percent each quarter, starting from a constant value and stopping when there are no more cells. `FIRST`, `LAST`, and `NEXT` can also be used inside the brackets, as can an integer index. `Prices[3]` refers to the third cell under the item `Prices`.

Formulas can be restricted to operating on certain cells with the `IN` and `SKIP` clauses. `IN` restricts calculations to deal only with specified cells. `SKIP` operates similarly, instructing a formula *not* to mess with the specified cells. The result is that formulas can have scope, and portions of a worksheet can be made to calculate differently from others.

Gone is the age-old problem with clobbering formulas by accidentally typing over them. Improv will not allow data entry in any calculated cell. Selecting a formula will highlight the cells it affects.

The symbolic arrangement of data in Improv makes formulas easier to create, easier to read, and, therefore, easier to maintain. With well-chosen item names, most formulas read like English; look closely at the formulas in photo 1 and see if you understand what's going on.

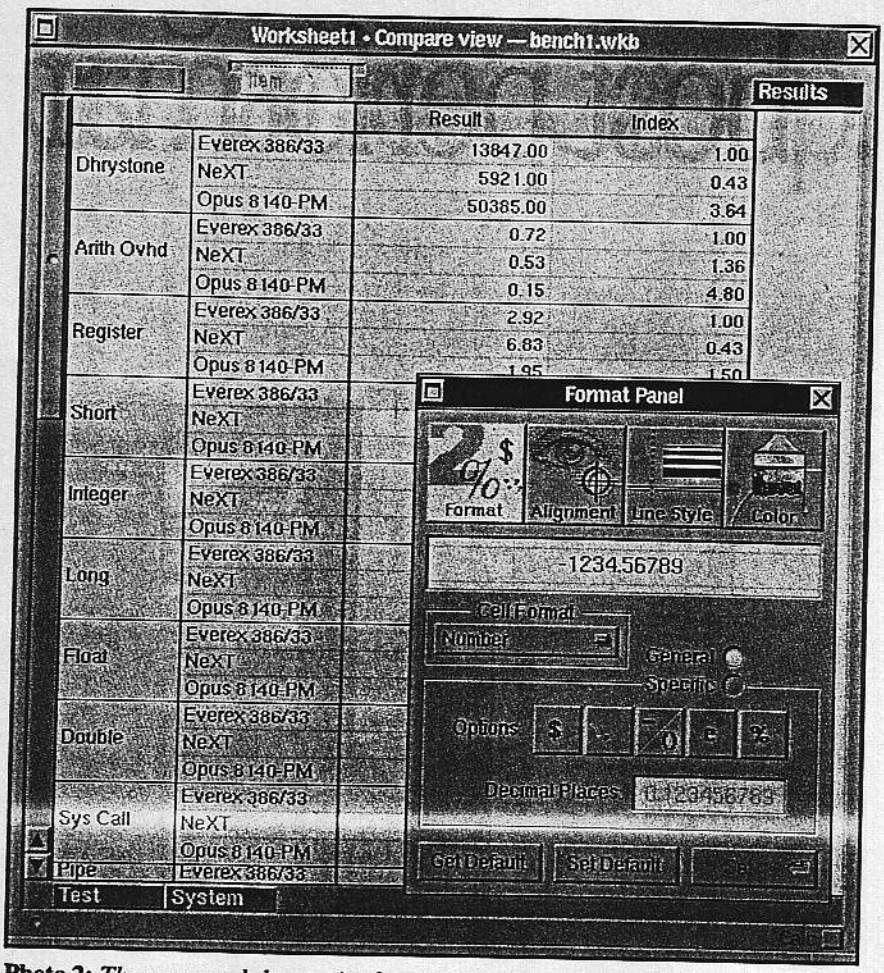

Photo 2: *The same worksheet as in photo 1, after having its data rearranged with flexible views. The format box is typical of Improv's interface: stylish and functional.*

The Hatchet Falls

Improv is *much* easier to use than to describe. There is great depth to its abilities, but the interface causes it all to make sense. To discuss *everything* that makes Improv special would fill several articles this size. Instead, here are a few points of interest that shouldn't be overlooked.

Presentation Builder is an Improv tool through which you create graphs from worksheet data. You can add freehand drawings, paste in worksheet contents, and bring in PostScript and TIFF graphics files. Help is rampant, and it can be brought up in context-sensitive, click-to-describe, and other modes. If you ask to open an Improv file that is currently in use by somebody else on the network, a warning panel is presented (which you can override).

In all, Improv knocked me out. It is the first *new* program that I've seen in months, and Lotus's design ideas are truly innovative. Lotus claims that the primary market for Improv will be experienced spreadsheet users who have outgrown the limited, traditional programs. That may be, but I'm convinced that new users, even those who have never used a computer before, will find Improv easier to learn and use. Running under Unix, Improv benefits from virtual memory, transparent networking, and multitasking. Running on the NeXT, it inherits the stunning NextStep graphical interface.

Lotus could have played it safe, producing (as with the Sun version of 1-2-3) a text-based spreadsheet that runs in a graphical window. Instead, the company decided to take a chance, and the results are astounding. You may never own a NeXT machine, but if you sit down to use a spreadsheet five years from now, chances are it will have a lot in common with Improv. ■

Tom Yager is a technical editor for the BYTE Lab. He can be reached on BIX as "tyager."