Adventures in Scientific Computing
An introduction to extensible programming

Jeff Martin
Developer Support Engineer
Joshua Doenias
Software Engineer
NeXT Computer, Inc

We showed our new prototype paint program to a friend recently. He made the usual comment, "It would be really great if only it had ±±±±±" (fill in the blank with your feature of choice, you've heard it before). One way to avoid having to solve everybody's requests with one program is to make your program extensibleÐthat is, write the program so you can add features without rewriting the source code.

For example, a CD player is an extensible environment. You don't buy a stereo with ten albums built in; you buy a CD player and some CDs to go with it. When a new CD is released, you don't buy a new CD player to listen to it. And one type of CD player suffices whether you listen to classical, country, or rock `n' roll. It's extensible!

Loadable objectsÐCDs for your software
Extensible applications are not new, but with the NeXTstep object-oriented development environment, creating them has never been easier. NeXTstep applications are collections of independent objects that know how to carry out functions and communicate with other objects. Window objects, for example, know how to draw themselves and move and resize themselves in response to mouse clicks. Extensible NeXTstep applications can load additional objects while they are running. Developers can write new objects that can be loaded into completed applications written by other developers. You can load these objects into your applications to give them new capabilitiesÐjust as you would add new CDs to your collection to hear new music.

Loadable objects add flexibility to NeXT applications
Applications that can load objects have unlimited flexibility. Several NeXT applications use loadable objects to increase their functionality. Interface Builder lets users load custom palettes into its palettes window. The custom palettes are then used to create advanced applications. For example, Objective Technologies, Inc. sells MathPalette™,which can be used to create custom front ends to Mathematica, a powerful mathematical symbolic manipulation program.

Lotus, Improv™,a revolutionary spreadsheet and data analysis application, also has the ability to load objects. OTProvide™, a collection of loadable objects from Objective Technologies, lets users extract data from databases and information services from within Improv. OTProvide then creates Improv worksheets from the retrieved data. Recently, the Los Angeles County Sheriff's Office used loadable objects to turn Improv into a donation tracking, tallying, and display system for its annual S.A.N.E. Kids Say No Telethon.

Applications and environments that never go out of style
Applications that take advantage of loadable objects are more than just applicationsÐthey become application environments. Just as the Workspace Manager is an environment for running applications and managing and accessing files, Interface Builder is an environment for building applications, and Improv is an environment for analyzing and manipulating data.

Applications that use loadable objects provide users with unlimited flexibility and a customizable environment, and because new loadable objects can replace old ones, they never become obsolete. Take a circuit design tool where each chip is an object. When new chips become available, users simply load them into their application. In summary, programming with extensibility in mind lets your application satisfy the needs of different users and turns an application into an environment for research and experimentation, thus allowing it to grow long after the original development ends.

A practical introduction
A list of programs that could take advantage of a flexible, extensible framework is long and varied. To show how this framework can be used, we'll discuss our paint program project.

As part of our development endeavors for NeXT computers, we are working on "Project DynaPaint," an extensible paint program now in the research phase of its life. It is presented here to show how an

extensible program can be structured.

DynaPaint is designed to load two kinds of objects: simple painting tools and filter objects. The standard painting tools that come with DynaPaint are loadable objects. These tools include a pencil, paintbrush, rectangle, line, among others.

To create a tool, programmers write an object that responds to the following methods:

```
- mouseDown:(NXEvent *)event;
- mouseDragged:(NXEvent *)event;
- mouseUp:(NXEvent *)event;
```

The application calls these methods in response to user actions such as a mouse click, drag, and release. The `mouseDown` method of a simple "rubberband" line tool, for example, saves the point where the user first clicks. The `mouseDragged:` method then draws a line from the original point to wherever the user drags the mouse.

Because the main program handles tasks such as redrawing, erasing, and undoing, the tools only need to know how to draw themselves. As a result, writing tools is simple and straightforward. A typical drawing tool takes up no more than one page of code.

Filter objects, which alter the selected region on the drawing canvas, are even more straightforward. The filter object must respond to only one method:

```
- filter:(NXBitmapImageRep *) thebits
```

`thebits` is a bitmap of the selected region on the canvas. When this method is called, the filter can examine and alter the contents of the bitmap to perform image manipulation operations such as blur and contrast-adjust.

All these objects can take advantage of Interface Builder to create user interfaces. For example, when

users select the Line tool, they may want to adjust the line thickness or color. Using Interface Builder, developers can create a Line Inspector window. The Inspector will get loaded into the program when needed.

On disk, tools are stored as file packages (folders that appear as a single file) and contain the following items:

- An object file that contains the code for the particular tool object
- A .nib file, created with Interface Builder, containing the Inspector window
- A text file for use in the help system

When DynaPaint loads a tool, it checks the file package for the tool object file and then uses a simple Objective C library call to load the tool. Then it loads the inspector window, and finally, it adds an icon to its tool palette to represent the newly loaded tool.

The NeXT Application Kit and the Objective C run time environment provide all you need to create extensible applications. All the tough work is done by an Objective C library function: `objc_loadModules()`; and an Application object method: - `loadNibFile:owner:withNames:`

The objc_loadModules() function dynamically loads a set of Objective C objects. The program can then use the loaded objects just as if they had been linked into the application at compile time. No special code is needed.

The power to load Objective C objects at run time should not be overlooked. NeXTstep makes it easy to customize and improve applications. As more developers release programs that can take advantage of this feature, users will not only customize programs to their liking, but also trade and perhaps even sell the modules they have created.