Contents

DOWN TO THE WIRE · NICHOLAS PETRELEY

Clarion top development tool to do the job right; an end to Win95 in sight?

TOPSPEED CORP.'S Clarion for Windows 1.5 is just around the corner. This is a must-have upgrade for Clarion developers, and it's terrific news for any developers looking to build applications for any Windows platform. Among the many other improvements and refinements, this version generates either 16-bit or 32-bit Windows applications. And Clarion clones features across the 16-bit/32-bit barrier, so you can generate 16-bit applications with a Windows 95 look and feel and 32-bit applications that use your existing 16-bit Visual Basic controls.

Oddly enough, a year or so ago the market for enterprise-level visual rapid database applications development tools for Windows (and how many buzzwords can you fit in one sentence?) were devoid of products that closely blend database access with an optimizing compiler Clarion for Windows 1.0 was the first to deliver this unique combination, but it didn't escape competition for long. On the heels of its release, Borland International Inc. shipped Delphi, an Object Pascal based visual applications development environment, also for Windows.

It's not without a certain irony that Clarion for Windows should compete head-to-head with Delphi. As it turns out, both Delphi and Clarion share some of their roots in Turbo Pascal. Delphi could be considered the latest version of Turbo Pascal. But did you know that Borland's Niels Jensen and the development team for Turbo Pascal left Borland to form Jensen and Partners International and created the line of TopSpeed compilers? After that, they merged with Clarion Software to form TopSpeed Corp.

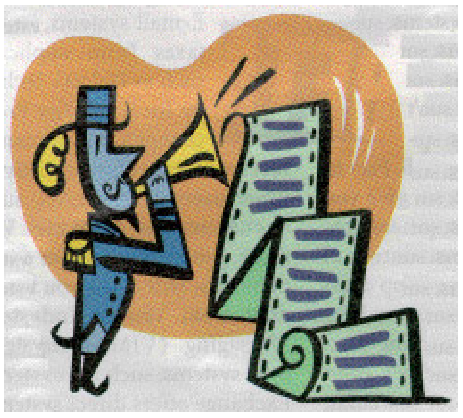
You say you want a revolution

Although Delphi is the logical evolution of Turbo Pascal, Clarion is the revolutionary product. Somewhere along the line, Borland stopped innovating and started imitating. That's not to say there's anything wrong with Delphi, which is, in a sense, Visual Basic done right. And in that respect Delphi is perhaps the most remarkable visual development environment of its type available for Windows. It deftly fills the gaping holes in Visual Basic (i.e., it is based on a language that is neither structured, object oriented, nor compiled).

Clarion, on the other hand, has always represented a complete rethinking of how database applications development should be done. It designed Clarion for Getting The Job Done, and it is unexcelled in this area, even by Delphi. The developers asked themselves what they could do to create a language and set of tools that could grease the whole database application development process as much as possible, and they succeeded admirably in reaching that goal. Then the TopSpeed folks came along and added to these tools the best compiler technology in the business.

So whereas Delphi makes it easy to create applications with complex custom interfaces and a unique look and feel, Clarion makes it far easier to generate robust, tight business database applications with a common look and feel - and do it quickly. It does this through its template-based applications generator and its powerful and supremely flexible COBOL-like, data oriented, event-driven modular language, adeptly fitted to the Windows environment. (You didn't think I could get more buzzwords in here, did you?)

And don't miss this point: Clarion is the best for business applications, and that's business with a capital "B Look at the attention to detail in this respect: In addition to all its other supported database formats, Clarion has native support for Btrieve, still one of the most widely used business database formats. And Clarion uses binary-coded decimal fixed-point arithmetic, which makes it easy to produce financial calculations without the rounding errors that can occur when you use floating-point math in other languages.



With its support for multithreading (something Delphi currently lacks) and capability to generate 32-bit applications for Windows 95 and Windows NT, Clarion for Windows, Version 1.5, will be an irresistible buy for developers wishing to jump into 32-bit Windows programming. Look for it later this month. You can give TopSpeed a call at (305) 785-4555.

Oh joy. Send me E-mail via the Internet to nicholas-petreley@infoworld.com or

HANDS ON

Clarion for Windows v. 1.0

$1,299
No royalties
TopSpeed Corp.
150 East Sample Road
Pompano Beach, FL 33064
(800) 354-5444
(305) 785-4555
(305) 946-1650 (fax)
Circle Number 150

REVIEWED BY K.J. BERNSTEIN

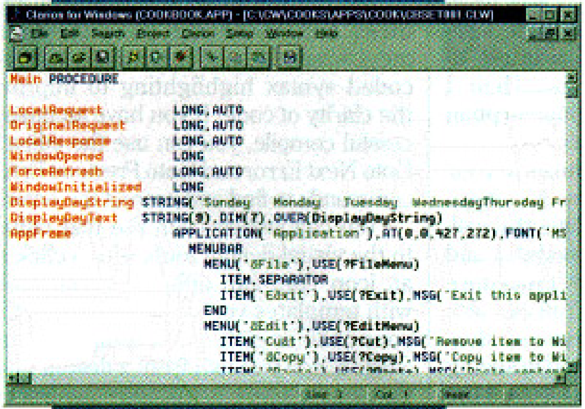
Can You Hear the Clarion Call?

If Not, You'll Miss Out on this New Database Development Environment for Business Apps

A FRIEND WAS INTERESTED IN STARTING a newsletter for his prospering music store. He asked me if I could help him write a program to keep track of his customers, the instruments they played, their achievements, and any other news that might be of interest. Because this was a favor for a friend, I wanted to produce the program as quickly and easily as I could. If there was ever a time I needed the "rapid" in rapid application development, this was it.

I looked around and found Clarion for Windows, a powerful, flexible, 4GL database-development environment. It combines shortcuts and visual design tools with its own custom programming language, allowing you to create applications with as much or as little code as you want -- or no code at all.

Clarion let me create a fast-and-easy program to help my friend with his newsletter.


```
Main PROCEDURE

LocalRequest          LONG,AUTO
OriginalRequest       LONG,AUTO
LocalResponse         LONG,AUTO
WindowOpened          LONG
ForceRefresh          LONG,AUTO
WindowInitialized     LONG
DisplayDayString STRING('Sunday   Monday   Tuesday  WednesdayThursday Fr
DisplayDayText   STRING(9),DIM(7),OVER(DisplayDayString)
AppFrame              APPLICATION('Application'),AT(0,0,427,272),FONT('MS
                        MENUBAR
                          MENU('&File'),USE(?FileMenu)
                            ITEM,SEPARATOR
                            ITEM('E&xit'),USE(?Exit),MSG('Exit this appli
                          END
                          MENU('&Edit'),USE(?EditMenu)
                            ITEM('Cu&t'),USE(?Cut),MSG('Remove item to Wi
                            ITEM('&Copy'),USE(?Copy),MSG('Copy item to Wi
```

CACOPHONOUS COLORS
Color syntax highlighting makes the code easier to read.

THE TRUMPET OF THE TEMPLATES

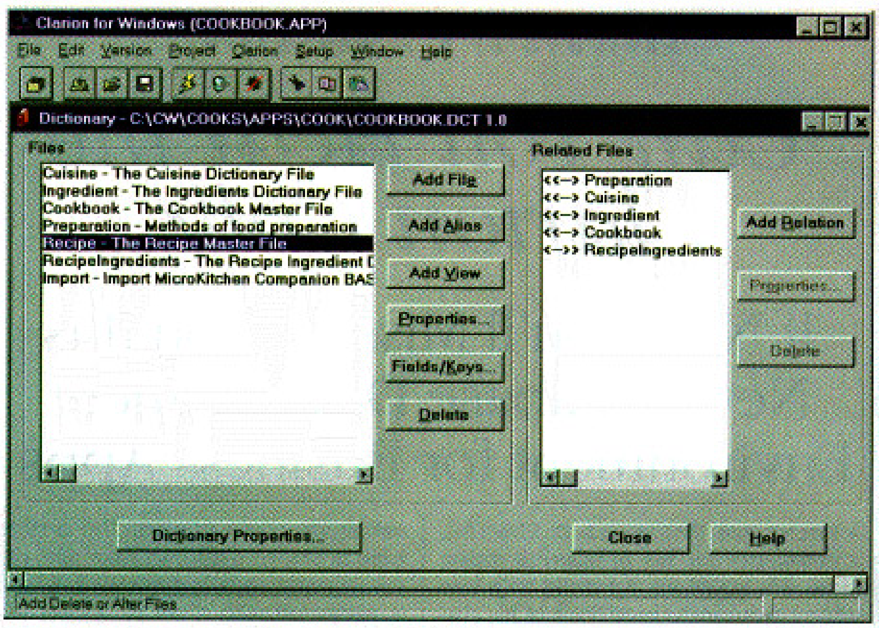
Clarion's installation went smoothly. The heart of Clarion's development environment is the application generator. It contains easy-to-use tools that let you customize the look and behavior of your app's elements, including windows, menus, databases, and reports. The generator uses templates of prewritten generic code and data structures, both which you can customize with visual-design tools and embedded source code. Clarion creates application source code based on the predefined procedures in the template.

'The templates are completely reusable, and one template can have different abilities depending on the customizations you've made. In each use, only the relevant code is generated. Clarion is packaged with a set of default templates-they let you simply plug in many standard features, providing a lot of instant gratification during development. For example, the procedure templates define a procedure type and include a generic window, an MDI parent frame, an SDI window, a source procedure to handle hand-coding, a process procedure to perform an operation on each of a database's records, an external procedure to declare a procedure contained in an external library, a browser, a form, a report procedure, and an ASCII file viewer.

Code templates can contain source code for both embedded source code points and in hand-coded procedures. These templates include source to initiate a thread (needed when opening an MDI window from an application frame), to call a look-up procedure, to control value validation, and to close the current window. The control templates define the appearance and function of the controls placed in a procedure's window. The control templates contain a number of browse controls; save, cancel, close, search, and print buttons; an ASCII listbox; and a DOS file look-up control. Finally, extension templates add specific functions that are not tied to a control. Together, these templates cover most of the facets of an application and can greatly reduce the amount of work a developer has to do.

If the templates that Clarion provides still don't give you everything you need, many third-party templates are available. You can also modify the default templates, or even use Clarion's template language to create your own.

The template language is a flexible script language that contains structures for getting information from the


Clarion for Windows (COOKBOOK.APP)
File Edit Version Project Option Setup Window Help
Dictionary - C:\CW\COOKS\APPS\COOK\COOKBOOK.DCT 1.0
Files
Related Files
Add File
Add Alias
Add View
Properties...
Fields/Keys...
Delete
Add Relation
Properties...
Delete
Dictionary Properties...
Close
Help
Add Delete or Alter Files

Cuisine - The Cuisine Dictionary File
Ingredient - The Ingredients Dictionary File
Cookbook - The Cookbook Master File
Preparation - Methods of food preparation
Recipe - The Recipe Master File
RecipeIngredients - The Recipe Ingredient [
Import - Import MicroKitchen Companion BAS

<<—> Preparation
<<—> Cuisine
<<—> Ingredient
<<—> Cookbook
<—>> RecipeIngredients

ORCHESTRATING THE DATABASE
The data dictionary is where you define fields, tables, and their relationships.

programmer during the design process and also for generating source code. Based on the options selected during program design, the language supports statements that control the exact source code put into an application. The script language also allows you to define points where embedded code can be inserted. This all adds up to provide a great deal of control in defining templates, and that translates into faster, easier application development.

Once you've precompiled and incorporated the template code in a registry file, the template is available in the application generator. Several sets of templates can be in the register at once, and the default templates, third-party templates, modified templates, and created templates can be used together in an application. The ability to mix templates from different sources is helpful when you're building a library of templates for an application.

PERCUSSIVE PROCEDURES

In the application generator, creating an application is a process of adding, defining, and customizing procedures. You add procedures whenever a call is made to a procedure that does not yet exist, such as when designing a menu procedure. A procedure type is defined using the template that most closely matches the function you'd Eke to give it. Once it's defined, you can customize the procedure with the design tools or with source code. The process is very straightforward, and as I set up procedures, I found that Clarion led me naturally through the process.

Procedures aren't set in stone-you can always go back later and redefine one. Any procedures that branch from the original procedure exist as "orphans" until you delete them. I found this especially useful when I was redesigning a part of an application. I removed the procedure I wanted to change and put in a new one. 'Men, I simply attached the original orphan procedures to the new procedure.

You can view the procedures in your application in several ways. A collapsible hierarchical tree displays the calling sequence and lists of modules and sorts them alphabetically by procedure name or by template type. You can also list modules by the procedures they own. Because I was interested in seeing the structure of my program, I used the hierarchical tree the most. Although the views of modules do provide useful information, they didn't seem as useful as the other views during development.

Clarion for Windows also contains a number of tools that make the design process easier. Clicking on a procedure calls the tool appropriate for its defining template. Clarion uses the template to automatically generate source code for items designed with the tool. Changes you make to the generated code are then reflected in future instances of the tool-more instant gratification with no loss of convenience when you start putting in custom code.

The windows structure formatter is a visual-design tool for creating an application's windows and dialog boxes. Starting with the template-defined format of the window, you can manipulate the window's properties and controls. VBX version 1.0 custom controls are supported, but unfortunately there's no support for OLE custom controls. The tool contains a menu editor, listbox formatter, and dialogs that let you specify the controls' properties. Although it's nice to be able to control every last detail of the window and window controls in the formatter, it is somewhat overwhelming at times. The formatter controls can also be cryptic at first.

The report formatter is a visual tool for laying out an application's reports. Report development is simple and fast, and the resulting report data-structure defines page formatting and page overflow management. The print engine takes care of the rest.

The formula editor generates statements that assign expressions to values. You can use it with computed fields, conditional fields, global variables, and local variables. The formulas you create are the equivalent of small functions, complete with conditionals and case structures. After a bit of initial fumbling, I found this tool very useful and powerful.

The programmer's text editor features auto-indenting, multiple-document windows, and customizable color coded syntax highlighting to improve the clarity of code. If you have an unsuccessful compile, you can use the editor's Goto Next Error and Goto Previous Error commands to find and correct the errors.

You can switch from the text editor to the visual-design tools with a click of an icon. You can either use the tools with templates you've already placed in your code or pick a new template from a list. When you exit from a design tool, the source code is instantly generated in the text editor. Here, too, changes in generated source code are reflected in future uses of the design tools. I enjoyed the convenience of being able to jump between coding and visual tools.

JAZZ FUSION

The Clarion language is a proprietary amalgam of languages incorporating elements of Pascal, COBOL, C++, and others. Coding in it gave me a strange feeling of deja vu. The following code snippet is from the Accept loop-the main event-handling loop in a Clarion program. The first event shown here fires when the application starts up and the second fires when the Window gets the focus.

```
ACCEPT
  CASE EVENT()
  OF EVENT:OpenWindow
    Splash
    IF NOT WindowInitialized
      DO InitializeWindow
      WindowInitialized = True
    END
  OF EVENT:GainFocus
    DO FLD2::FilIList
    DO RefreshWindow
    IF NOT WindowInitialized
      WindowInitialized = True
      DO InitializeWindow
    ELSE
      ForceRefresh = True
      DO RefreshWindow
    END
  ELSE...
```

Clarion compiles to native, 16-bit executable code using the TopSpeed optimizing compiler and linker. This compiler gained fame some years back for the speed of its executables. Performance in a database application, though, is more dependent on the database engine and the database schema than on the compiler. The Clarion database engine provided adequate performance in my simple program, but I can't say how well performance would scale up.

TURN THOSE PROPERTIES DOWN!

You can also use the application generator to set global properties that define the app's behavior as a whole, including how and when files are opened and closed. You can also use it to create global variables that can be defined in the same manner-in fact, with the same dialog-as fields.

A project-management system similar to a Make utility visually manages the source code files, external libraries, resources, and other application components. You may not always need the project manager, because the application generator takes care of project maintenance. You’ll need it if you use Clarion without the application generator.

A database dictionary is an essential part of a database application. It holds a description of the database, including its database drivers, fields, keys, relations, field-validation rules, and referential-integrity constraints. Clarion supports the database structure, integrity, and manipulation required by the relational model. Multiple files, including file aliases and views, can belong to a database dictionary. You can add an alias for a file already in the dictionary. Ibis lets you have a second file buffer for a file, as well as multiple relational links between files while obeying the strict theory of relational databases. A view is a virtual file constructed from selected fields taken from multiple files. Because you can choose to use only the fields that are actually required for an operation, this feature provides an advantage in a client-server environment.

Field definitions let you define almost every aspect of a field's appearance and behavior, including the type, field length, display-string format, initial value (which may be a function), default prompt, column heading, controls, validity checks, typing mode, justification, help information, and instant event notification to controls referencing the field. (This is another feature that might overwhelm you with choices.) The types of fields available depend on the file driver you've selected. The Clarion

Clarion for Windows supports many database formats.

database driver lets you select any of the standard data types, including a memo field that you can flag to hold binary data. There is also support for a group (compound) type and a LIKE type that takes the type of another field.

WHAT?

Clarion for Windows supports a number of database formats beyond the proprietary TopSpeed data file format. Drivers are provided for ASCII, Basic, Btrieve, Clarion 2.1/3.0, Clipper, dBase 3 and 4, DOS (binary byte addressable), FoxPro, and Paradox 3. Several other drivers, such as AS/400, Oracle, and Sybase, are available separately. ODBC support is also included. With it, Clarion can handle any file system that has an ODBC driver. This adds client-server support with the selection of an appropriate file driver.

During development, the data dictionary is maintained by the dictionary editor. Through this editor, you can define all of the properties, files, fields, relations, keys, and so on.

You can access data files directly, without an application, using the database manager. It's a programmer's tool, with no validity or referential integrity checks. Use it with care.

Although Clarion for Windows focuses on database development, the application generator lets you create non database applications. Once you have turned off the "Require a dictionary" option, you do not need to include a data dictionary.

Two shortcut tools, Quick Start and Quick Load, make the application generator easier to use by giving you a head start in development. With Quick Start, you fill in a few fields about the application in general and then define the name, display format, and key for each field in its data file. It takes only minutes to enter the information. Once you're done, an entire application is automatically constructed with a data dictionary, a view window and report for each key, and a form procedure for updating all fields in the database file. You get immediate results, which you can use right away or modify.

Quick Load is an easy and fast way to add a new file definition to an existing data dictionary. Like Quick Start, you fill in a few items of general information and then define the name, display format, and key for each field that's in the new file. The full data file is generated automatically.

Because you provide only the most basic information for these applications and database files, the results of Quick Start and Quick Load are generic -- whatever fields you set up are always used in the same manner. For example, in the forms and reports, fields are always placed in a column and set to the default values. Though you can use them as they are, you will probably want to customize. You can do this just as you would with any other Clarion application or database. There's no difference between an application created with the shortcut tools and the same application created manually.

LEARNING TO PLAY

Clarion for Windows is powerful and flexible. But the power and flexibility come with a price-a steep learning curve. Once you familiarize yourself with the basic concepts, however, the Clarion language is not difficult to learn. It may take some time to get acquainted with its full capabilities, but the structure is, for the most part, straightforward, and the language reference presents clear explanations and code examples.

There are different levels on which you can use this product, and the amount of effort you need to put into mastering the development environment depends on how much of the environment you want to use. For example, using just the shortcut tools and application generator, you can complete a database application within hours of removing the shrink-wrap. But going beyond the application generator with custom coding requires that you learn the Clarion language. Naturally, this takes more time than simply figuring out the application generator.

Basic navigation around the development environment and elementary concepts can be a bit confusing. Just figuring out what each icon represents, determining which menu choice you want, and wading through dialogs full of options can be frustrating. However, Clarion provides plenty of assistance-and you'll probably need it, too. The Guided Tour walks you through the development process and shows you actual application-development windows with descriptions of each button and field. Tutorials provide a step-by-step, hands-on introduction to developing an application.

The package contains a full set of manuals and online Help. The online documentation includes a guide to examples, the full text of the language reference, and a master index to the manuals. The documentation seems complete and provides straightforward explanations. Sometimes I needed to search to find the explanation for a particular icon or option, but I generally found the documentation on par with the better software documentation I have encountered.

If you still have trouble, technical support by phone and fax is free for 30 days from the date of your first call. Although it was sometimes difficult to get through to a real live tech-support person, the person I talked to was very helpful. After the first 30 days, pay-per call support is available through a 900 number, and free support is available on CompuServe. Technical and marketing documents are available with a fax retrieval system. Patches and other files are available through both a BBS and CompuServe. And if all that doesn't help, there's a support system that lets you report problems and make suggestions electronically. You can also view and manage a database of previously reported problems and suggestions.

IT TOLLS FOR THEE... IF YOU DON'T BUY IT

Clarion for Windows is thorough in anticipating the needs of programmers. It's a complete application development package whose capabilities cover the spectrum from instant gratification to complete customization.

However, the ultimate test of a development environment is the quality of the applications created with it. Here, too, Clarion for Windows shows its value. In my testing, the application generator produced code that ran smoothly, and database operations proceeded speedily with the TopSpeed driver. So even though Clarion is a bit confusing, even frustrating, at first, its convenience and power make learning the environment and language well worth the effort.

K.J. Bernstein is a software engineer for Duffers Scientific Inc. in Poestenkill, New York.

Pick a Database That Runs on its Records

THE TWO DATABASE experts blinked. I'd asked them how I'd choose among Microsoft's database offerings.

"Well, it one said, "I can't give you a general rule of thumb, but if you give me a concrete example, it'll be obvious."

"Okay here's a concrete example: You're in the hotel business and you want to write a central reservations system. It has to support distributed databases. The tables potentially have millions of records and a record has to come up very quickly.

"If a guest calls the main toll-free number and says, 'Hi, this is John Smith from Omaha, and I'd like a room for next

Illustration: SCOTT BALDWIN, Photo: RICK FRIEDMAN

Monday,' the reservations clerk should answer, 'So you're coming to Dubuque again, Mr. Smith? I can reserve your preferred nonsmoking room with a king-size bed overlooking the pool. Would you like to guarantee that for late arrival with your American Express card?'

"How would you do that?"

"I'd do it with SQL Server on the back end and Visual Basic on the front end."

I smiled broadly. "That's exactly what I told my client at the time. Of course, since then, the client decided to use Delphi and Oracle instead.

"Okay, now try this one: You run a hospice, and your records system needs to track the dying patients, their doctors, their case workers and their families."

Microsoft's database guru started to look uncomfortable.

"It's all right. People do die, and a hospice makes the process more bearable for the patient and the bereaved. I do some pro bono consulting for them and make some bereavement calls. One problem this hospice has is generating all the paperwork required after a death-the sympathy cards, the mailing labels, the forms for volunteers who work with the bereaved. Right now the staff types everything over and over for the different forms. The database schema is pretty simple, and the biggest table runs to thousands of records-tens of thousands, tops."

The database guru gave me a big smile. "That's easy. I'd do that with Access."

"But what about Visual FoxPro?" Visual FoxPro (800-426-9400) was what this meeting was ostensibly about.

Martin
Heller

"Well, sure, Fox could handle it, but Access would be a lot easier, especially if they themselves wanted to modify the forms later. They probably don't have too many technical people (I smiled thinly), but even the clerks should be able to add a field to an Access form using its visual design tools. Fox might be too complicated for them to maintain. If they needed hundreds of thousands of records, Fox would win because it scales better, but for thousands of records Access is just fine."

Eventually, we did get around to looking at Microsoft's Visual FoxPro, which I found impressive and a compelling choice for Xbase developers who want to migrate to a visual, object-oriented environment. But either I'm easily impressed these days or database development products are getting very good indeed. I find Clarion for Windows (TopSpeed Corp., 800-354-5444) and Delphi for Windows (Borland International, 800453-3375, x13O9) equally impressive for various reasons. But I'll have to defer any further discussion of Visual FoxPro until I actually have a copy in my office.

First there was Turbo Pascal

Clarion and Delphi are both spiritual descendants of Turbo Pascal. Those of you up on Borland trivia will recall that a 14-year-old Dane named Nils Jensen wrote the original Turbo Pascal product and Philippe Kahn marketed it successfully for $99. Borland rose from there, diversified, acquired, grew-and faltered.

At one crucial point, as the story goes, Nils and his key people had their own C compiler under development when Philippe bought Wizard C instead and renamed it Turbo C. Nils left, formed Jensen & Partners and came out with the TopSpeed compiler family. Eventually, Jensen & Partners (which had some great compiler and linker technology, but no database technology and not much marketing clout) merged with Clarion Corp. (which had some great database development technology and a decent installed base, but nothing worthwhile as far as compilers). The resulting Topspeed Corp. seemed to go quiet for more than a year and then surfaced with Clarion for Windows.

Meanwhile, back in Scotts Valley, Turbo Pascal went through revision after revision, picking up more depth with each iteration, moving nicely to Windows and adding object orientation. Then suddenly there were mutterings about this new Delphi product that was being called a VB-killer, a PowerBuilder-killer and a visual Pascal with reusable objects and integrated databases.

You've heard about Delphi by now. I've mentioned it before, and most of the magazines and journals for Windows programmers have written it up. It's a fine product, with in-memory compilation, two-way visual tools, a single-user version of InterBase Server, ReportSmith and all the Borland design and debugging tools, classes and samples. It requires a CD-ROM drive and at least 30MB of free disk space-or closer to 90MB if you want to do a full install. People keep writing me to ask how I like Delphi. I like it fine, except for the amount of disk space it hogs.

Nobody writes to ask me how I like Clarion, and I think that's a crying shame. It may not have all the polish and features of Delphi, but it's the most efficient database development package I've ever used-not to mention that it installs from five diskettes. Despite the Microsoft guru's recommendation of Access, I've decided to build the applications the hospice needs with Clarion. I haven't had much time to work on them yet, more's the pity, but when I do I'll let you know how they turn out.

Clarion feels right

I have begun to learn the Clarion environment. It's a bit different, but for a database application it feels right. If you can design a data dictionary and pick the right templates, you can have a working, compiled database application faster than you could ever have imagined. Clarion's Quick Start feature gets you from the data dictionary to a vanilla database application in minutes. And templates let you add standard functionality in hours rather than days.

So, what's a template? If you know what templates are in C++ you already have a pretty good idea. A template is a prewritten procedure skeleton that allows for parameter substitution. In Clarion, the parameters are things like tables, data fields, data validation ranges and referential integrity checks. In other words, if you want to implement a browse procedure in Clarion, you won't have to write any code. just place a few templates and customize their properties, in much the same way as you'd place bound controls on a VB form. The major difference is that the Clarion template does a lot of stuff you'd have to hand-code in a VB application.

I'm a language junkie from way back. Clarion has its own fourth-generation language, which isn't exactly like Pascal or C or Xbase. I haven't bothered to learn it. I haven't even needed to look at the generated source code. I just treat the entire Clarion development system as a black box that turns my design into an application.

I haven't gotten to the point of stressing the system yet. I haven't found out how well Clarion handles reading and writing locks in multi-user applications or how well it imports data in odd formats with entry errors or any of the other things that make or break a database system in real life. Stay tuned.

Martin Heller consults for a variety of businesses, illuminates dark code and writes in Andover, Mass. Contact Martin care of the editor at the addresses on page 18.

TopSpeed development tool yields fast, engaging database applications

REVIEW Clarion for Windows is rich in data-definition skills, but lacks graphical aids for some functions

BY PETER COFFEE

The late-April update of TopSpeed Corp.'s Clarion for Windows 1.0, Release 1002, is an outstanding tool for developers who want to automate access to data in fast, attractive, stand-alone applications. Clarion for Windows demonstrates a fine balance between making simple things obvious and making complex things accessible but not intimidating.

The $1,299 front-end tool greets the developer with an empty data dictionary, instead of a blank GUI "canvas." Developers can easily refine the user interface with Clarion's convenient drag-and-drop tools, but only after defining a data model for their application.

Microsoft Corp.'s Visual Basic 3.0 or Borland International Inc.'s Delphi 1.0 (see comparative review, April 24, Page 71) still provide more direct access to the basics of algorithms and data structures.

In addition, we continue to recommend Delphi for applications where complex code is a major driver of project schedule and cost.

DEVELOPMENT TOOLS

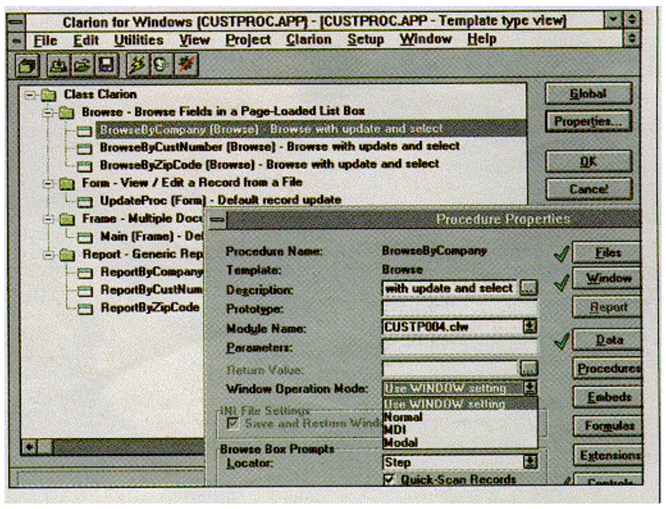
Clarion's tools, both text-based and graphical, follow current programming conventions and offer informative and helpful pointers during development.

For example, when defining the data dictionary for an application, Clarion displayed examples of entries for various fields at the bottom of the screen, freeing us from having to consult documentation.

During our tests, prototype applications took shape quickly, providing menu-driven com-. mands to browse, update, and print reports based on the data-dictionary definition with no further coding.

The integration of database definition and data access in the Clarion development cycle encouraged us to provide sophisticated query and report options even at early stages in its prototyping process.

In addition, prototyping was accelerated by Clarion's excellent error handling, with quick detection and one-click access to the error's location and on-screen diagnosis of possible corrective actions.


Clarion for Windows [CUSTPROC.APP] - [CUSTPROC.APP - Template type view]
File Edit Utilities View Project Clarion Setup Window Help
Global
Properties...
OK
Cancel
Procedure Properties
Files
Window
Report
Data
Procedures
Embeds
Formulas
Extensions
Controls

Class Clarion
Browse - Browse Fields in a Page-Loaded List Box
BrowseByCompany (Browse) - Browse with update and select
BrowseByCustNumber (Browse) - Browse with update and select
BrowseByZipCode (Browse) - Browse with update and select
Form - View / Edit a Record from a File
UpdateProc (Form) - Default record update
Frame - Multiple Docu
Main (Frame) - Def
Report - Generic Rep
ReportByCompany
ReportByCustNum
ReportByZipCode

Procedure Name: BrowseByCompany
Template: Browse
Description: with update and select
Prototype:
Module Name: CUSTP004.clw
Parameters:
Return Value:
Window Operation Mode: Use WINDOW setting (Use WINDOW setting, Normal, MDI, Modal)
INI File Settings: ☑ Save and Restore Wind
Browse Box Prompts
Locator: Step
☑ Quick-Scan Records

CLARION FOR WINDOWS gives a hierarchical view of project components and flexible control of application options.

PERFORMANCE

In a processor-intensive empty-loop benchmark, Clarion ran much faster than Visual Basic, even surpassing Delphi and Blue Sky Software Corp.'s WinMaker Pro 6.0 despite disabled optimizations and the inclusion of debugging code.

In tests involving computation and display, Clarion for Windows provided a more flicker free display during rapid updates than Visual Basic while running only 20 percent slower.

Even the C-based WinMaker Pro and Pascal-, based Delphi were only 20 percent to 30 percent faster than Clarion in a tight data-display loop (see benchmark chart, Page 79)

Although not the performance leader, Clarion for Windows is still one of the most efficient tools we've seen for high-level database access. However, Clarion's EXE files, though fast and small, rely on a 577K-byte DLL (dynamic link library) that is almost 50 percent larger than Visual Basic's run-time DLL.

UNDERLYING ARCHITECTURE

The Clarion programming language has overtones of both XBASE and COBOL, and we found Clarion code quite readable. The product includes a variety of sample applications that helped us become quickly productive with the language.

With years of refinement under its belt, the Clarion language has evolved to combine strong support for basic data processing with well-conceived extensions for Windows.

Though not as elegant as Delphi's language, Clarion will lend itself to production coding by those who do it as a job rather than an art form.

With its use of text files for storing program components, Clarion will work smoothly with any of the more common version managers, such as Intersolv Inc.'s PVCS. However, Version 1.0

AN EVOLVING LANGUAGE

Though not as elegant as Delphi's language, the Clarion language has evolved to combine strong support for basic data processing with well-conceived extensions for Windows.

FOR YOUR INFORMATION

Product	Company	Phone
Clarion for Windows 1.0	TopSpeed Corp. Pompano Beach, Fla.	(800) 354-4444 (305) 785-4555
WinMaker Pro 6.0	Blue Sky Software Corp. La Jolla, Calif.	(800) 677-4946 (619) 459-6365
Delphi 1.0	Borland International Inc. Scotts Valley, Calif.	(800) 223-2444 (408) 431-1000
Visual Basic 3.0	Microsoft Corp. Redmond, Wash.	(800) 426-9400 (206) 882-8080

PROGRAMMABLE FRONT-END TOOLS

PCWEEK LABS CORPORATE Buyers' Advisory

	Clarion for Windows 1.0 TOPSPEED CORP.	WinMaker Pro 6.0 Reviewed April 24 BLUE SKY SOFTWARE CORP.	ANALYST'S CHOICE Delphi 1.0 Reviewed April 24 BORLAND INTERNATIONAL INC.	Visual Basic 3.0 Reviewed April 24 MICROSOFT CORP.
PROS	Development cycle integrates database definition; excellent integration of tools; fast development and execution of code; language easy for COBOL, dBASE developers to learn.	C/C++ tool sets already familiar to developers; produces smallest executable files.	Modern software-engineering foundation; fast, well-integrated development cycle; fast applications with good database support.	Easiest of the packages to learn for non-programmers; support for OLE automation and ODBC data access; broadly supported by third-party tools.
CONS	No Object Linking and Embedding; large run-time dynamic link library required; many functions require coding without graphical aids.	Most difficult to learn for inexperienced developers; generated code was sometimes slower than Delphi and Visual Basic; database access requires separate product.	Pascal foundation less familiar than BASIC or C/C++; client/server add-on required for some tasks; executables not as compact as WinMaker Pro's.	CPU-intensive tasks notably slow; language vulnerable to many common errors; abstract data typing limited.
RECOMMENDATIONS	Where application backlogs are driven by database design rather than interface bells and whistles, Clarion for Windows is a preferred migration path for corporate developers who are ready to move to a GUI.	Organizations that are using C/C++ but are looking for a better tool with which to build user interfaces should take a good look at WinMaker Pro.	If Visual Basic is too slow or too undisciplined, and if C and C++ intimidate your middle-weight developers, then Delphi is the right choice for your organization.	Organizations that are happily using Visual Basic should stick with it and wait to see Version 4.0, which is due later this year.

Clarion

from page 78

provides no built-in support for using these tools as part of the development cycle.

As with most competing tools, Clarion for Windows provides no group development features and no repository for non-sequential operations using multiple, multi vendor tools.

OBJECT-PROGRAMMING SUPPORT

Clarion's environment clarifies relationships among application components with clear graphical views. In addition, the data dictionary's discipline makes a clear distinction between data and application logic—clearer than that of products using C, Pascal, or BASIC, which treat data as a mere external resource.

We found it both practical and productive to build applications on the foundation of templates included with the product, and extendable by the developer.

These templates represent such basic application components as windows, menus, data browsers, reports, forms, and viewers.

Clarion's approach offers the reusability of objects without the conceptual leap required to move from COBOL or dBASE to Smalltalk or C++. Although Visual Basic and Delphi make definition of reusable elements more intuitive and scalable, Clarion developers will still achieve significant productivity gains by reusing common application components.

TEST METHODOLOGY

We performed our development tests of TopSpeed Corp.'s Clarion for Windows 1.0 on a 90MHz Pentium-based Micron Electronics Inc. machine with 16M bytes of RAM. For performance benchmarks, we used a 66MHz 486DX2-based Digital Equipment Corp. 466D2LP workstation with 16M bytes of RAM. We tested Clarion for Windows running under Microsoft Corp.'s Windows 3.1, with IBM's PC DOS 7 during development and Microsoft's MS-DOS 6.2 during benchmarking.

PCWEEK LABS Scoreboard

	TOPSPEED CORP. Clarion for Windows 1.0	BLUE SKY SOFTWARE CORP. Reviewed April 24 WinMaker Pro 6.0	BORLAND INTERNATIONAL INC. Reviewed April 24 Delphi 1.0	MICROSOFT CORP. Reviewed April 24 Visual Basic 3.0
DEVELOPMENT TOOLS				
Initial prototyping	EXCELLENT	FAIR	GOOD	EXCELLENT
Query and report creation	GOOD	NA	GOOD	GOOD
Adding and refining user-interface elements	GOOD	GOOD	EXCELLENT	EXCELLENT
PERFORMANCE				
Screen I/O	EXCELLENT	GOOD	EXCELLENT	EXCELLENT
Processing speed	EXCELLENT	EXCELLENT	EXCELLENT	FAIR
Interactive operations	GOOD	GOOD	GOOD	GOOD
Resource usage and requirements	FAIR	EXCELLENT	GOOD	FAIR
UNDERLYING ARCHITECTURE				
Strength of programming language	EXCELLENT	GOOD	EXCELLENT	FAIR
Support for version control	FAIR	FAIR	FAIR	FAIR
Group-development support	NA	NA	NA	NA
Strength of repository	NA	NA	NA	NA
OBJECT-PROGRAMMING SUPPORT				
Clarity of application structure	EXCELLENT	FAIR	GOOD	GOOD
Encapsulation of data and logic	EXCELLENT	GOOD	EXCELLENT	FAIR
Ease of identifying and reusing components	GOOD	GOOD	EXCELLENT	EXCELLENT
DATABASE SUPPORT				
Database driver support	EXCELLENT	NA	FAIR	GOOD
Efficiency of using database-server functionality	GOOD	NA	GOOD	GOOD
Ease of defining database-server connections	FAIR	NA	GOOD	GOOD
DESKTOP INTEGRATION				
Support for interapplication APIs on native operating system	FAIR	EXCELLENT	GOOD	EXCELLENT
Import/export	GOOD	GOOD	GOOD	GOOD
PLATFORM SUPPORT				
Windows	EXCELLENT	EXCELLENT	GOOD	EXCELLENT
Macintosh	NA	NA	NA	NA
OS/2	NA	NA	NA	NA
Unix	NA	NA	NA	NA
DOCUMENTATION AND SUPPORT				
Ease of installation and learning	GOOD	GOOD	EXCELLENT	EXCELLENT
Reference materials	EXCELLENT	POOR	GOOD	EXCELLENT
Vendor assistance	GOOD	GOOD	GOOD	GOOD

DATABASE SUPPORT

Clarion provides efficient, format-specific data file access drivers for ASCII, Btrieve, Clipper, dBASE III/IV, Foxbase/FoxPro, and Paradox 3, as well as the older DOS-based Clarion 2.1/3.0 and a new proprietary TopSpeed format.

Access to different data formats and database servers, including those used via Open Database Connectivity, is accomplished with code, not graphical tools—a process that is not obvious. A product with Clarion's emphasis on database access should provide more straightforward tools, like those of Trinzic Corp.'s Object Pro.

DESKTOP INTEGRATION

Like other aspects of Windows, DDE (Dynamic Data Exchange) is accessible from Clarion to developers willing to write the required code.

By calling the DDESERVER function, for example, we obtained a DDE server channel number that we could then use with such functions as DDEWRITE to exchange information between applications. The resulting code is clear, not cryptic, but it does require some effort to absorb.

Clarion applications can also interact with other Windows applications and data sources via programmed access to the Clipboard, with strings both readable and writable from and to that shared resource.

OLE (Object Linking and Embedding), however, is not included in the Clarion developer's tool kit which is a shame and places it in the shadow o@ such products as Visual Basic.

Clarion's excellence in defining data sets and data interactions would make a Clarion-based browser a useful OLE server to embed in other applications.

PLATFORM SUPPORT

Clarion, a Windows-specific front-end tool, is as good a package for developing 16-bit applications as either Visual Basic or Delphi.

When TopSpeed ships Version 1.5 late in the summer, Clarion will provide an upgrade path to the 32-bit Windows environment. The 1.5 release will include both 16and 32-bit compilers and add new user-interface, controls for compatibility with Windows 95.

DOCUMENTATION AND SUPPORT

Clarion for Windows installed easily and automatically made all the required settings. We found the Getting Started manual both accessible and thorough in helping us understand the product's unique approach to developing applications. We strongly recommend taking the time to follow the hardcopy tutorials.

We also found the Language Reference manual clear, well-organized, and usefully indexed. Unlike most other manuals, Clarion's seemed to anticipate our questions. In particular, the Language Reference manual offers an interesting and informative description of the language's rationale.

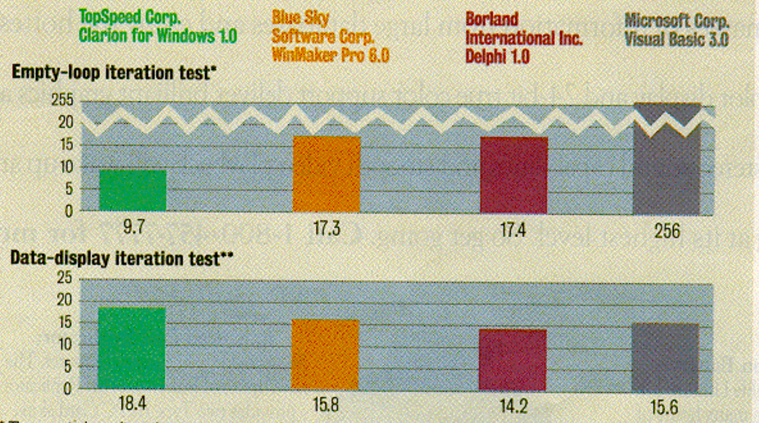
Clarion developers find both vendor support and mutual assistance in a CompuServe support forum. We found one discussion that involved a possible bug in one of Clarion's numeric data types that may lead to unpredictable changing of the arithmetic sign of a value. TopSpeed's technical support responded promptly and courteously to this report.

CLARION FOR WINDOWS RUNS AHEAD OF PACK IN EMPTY-LOOP TEST

Programmable front-end tool a bit slower than competitors in data-display iteration test

PCWEEK LABS

Measured in seconds. Shorter bar indicates better performance.



* The empty-loop iteration test measures how long it takes to cycle through an empty loop 125 million times.

** The data-display iteration test measures how long it takes to display 5,000 data values.

Review

Will Fastie

Clarion for Windows

Clarion for Windows (CW) is a rapid application development tool. The integrated develop ment environment (IDE) includes a compiler, an application generator (AppGen), a text editor, a screen/window editor, a data dictionary editor, a project management system, a report designer, and many other elements to support visual application design. CW also includes a "template" language facility, which drives AppGen, and a system of interchangeable database drivers that enable native (direct) access to a long list of database types, including TopSpeed's own Clarion and TopSpeed formats as well as dBASE, Paradox, ASCII, DOS binary, Btrieve, ODBC, and SQL. Underlying all this is the Clarion language itself, an elegant but practical programming language ideally suited for business applications but perfectly acceptable for any kind of programming.

There are three ways to use CW as a tool. An application can be built entirely withAppGen, without writing a single line of code. This is an ideal approach for larger organizations wishing to build front-end programs to browse and maintain legacy mainframe data or newer client/server data. Or, an application can be entirely hand-coded using the Clarion language. Because of the language's design and its inherent abstractions, hand-coding is remarkably efficient and typically faster than using any other language. Finally, an application can be built in a hybrid fashion, with the developer using AppGen to the extent desired while embedding Clarion code freely throughout the project.

AppGen is perhaps the most appealing characteristic of CW AppGen is centered around the display of a tree diagram of the application under construction. Each leaf (node) on this tree represents a type of procedure whose properties are enumerated in a series of dialogs (see Figure 1) that appear when the developer clicks on the procedure of interest. Properties include such things as a list of local data, a list of files needed by the procedure, and the layout of the Window if the procedure has one. Properties also include any custom code the developer cares to write, code that is automatically embedded at the appropriate point when AppGen generates code. A procedure can also include nothing but hand-written code, if desired.

Once the application structure has been defined, the user builds the program. AppGen uses the property lists to generate Clarion source code. This is automatically merged with any hand-written code the developer may have produced. The combined source code is then compiled and linked. The project system is incremental, so only new code or changes are recompiled. The resulting program is native x86 code in EXE format, a module that can be run or debugged from within the CW environment.

Such a brief description of AppGen hardly conveys its power. AppGen is controlled by templates that define 'available procedures. Templates are a combination of Clarion code, template language (macro) code, and pre-defined properties. A user entering information in the property window of a procedure is actually providing the variable information needed by the template. Once the variable information has been supplied, AppGen can generate native code customized to the developer's specifications.

A brief example will help to explain this. CW comes with a "browser" template-a browser is nothing more than a window that contains a scrolling region in which records (or views) meeting certain criteria are displayed. The browser template doesn't know anything about the file it might be browsing, the fields from those records it might be displaying, or the criteria by which it might select records. The browser template is completely general, able to display any set of fields from any file supported by the database driver system. Let's say the user wished to browse a typical customer file, showing the customer's name and phone number. AppGen takes its generic template, combines it with procedure properties and any embedded code, and generates specific Clarion code to handle the particular case. The user writes no code!

Snapshot

TopSpeed Corporation
150 East Sample Road
Pompano Beach, FL 33064
800-354-5444 or 305-785-4555
305-946-1650 FAX

Summary: A rapid application development tool that includes an integrated development environment (IDE); a powerful, template-driven application generator; replaceable database drivers for standalone, transactional, or client/server development; a data dictionary editor; and a native-code compiler. Applications can be developed using AppGen, hand-written code, or both. Front-end browsers for client/server or legacy data can be built very quickly.

List Price: Clarion for Windows 1.001, $1,299.

Documentation: Over 1,000 pages in four documents. Extensive online help, plus additional information on database drivers.

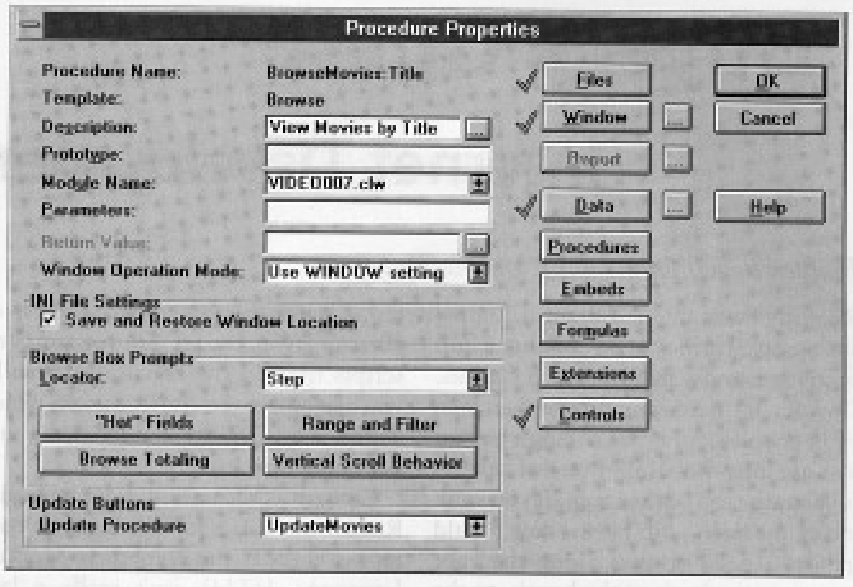
Circle 3 on Reader Service card

The templates, being source code themselves, can be modified or completely replaced by the developer. For example, if a developer preferred different browsing behavior, it would only be necessary to revise the browser template accordingly. Several companies, including both long-standing third-party Clarion vendors and some new entrants, have already written add-in templates for a variety of uses. C3 (708-3 8 5-9 844) and Boxsoft (416-944-2358) have updated their suites of templates (previously available for Clarion for DOS), while newcomer Toolcraft (408-7324300) has released an enhanced browser. Others will surely follow.

Visual design with CW is similar to most other products in this category. The CW window editor has a basic set of controls that can be enhanced with VB custom controls (VBX). Using these controls is a simple drag and drop affair, almost identical to using VB. The property system of the controls is visible in CW, so the handling of controls is also similar to that of VB. CW can handle only level I VBXS, and not all level 1 controls work with CW (although this is often the fault of the VBX itself). There is no list of controls that do work with CW, so trial and error (or begging and bribing people on the CLARION forum on CompuServe) is necessary to learn the ropes. TopSpeed does not plan to significantly enhance VBX support, opting instead to support OLE custom controls (OCXs)-a sensible direction. An aside: If TopSpeed can maintain its schedule, it could have OCX support in CW before Visual Basic 4.0 ships.

On all fronts, CW is fast. In one test case, AppGen generated 5,700 lines of Clarion code, then compiled it all and linked it in 90 seconds on a 486DX2/ 66. 1 wanted to get a better idea of how fast Clarion is, so I took a number of snippets of production code, wrote I them in CW, Visual Basic, Visual C++ 1.5, and several PC database prod ucts, and then compared the results. I VB is the fastest interpretive system, beating most of the other PC database tools by a factor of five. However, CW is dramatically faster than VB in these tests-from six to eighteen times faster! ' CW stood up well against C also, t averaging 60 percent of the speed of C. I Visual performance was measured by designing a window with a reasonable number of controls, then dis-

FIGURE 1—AppGen properties dialog box.


Procedure Properties

Procedure Name: BrowseMovies:Title
Template: Browse
Description: View Movies by Title
Prototype:
Module Name: VIDEO007.clw
Parameters:
Return Value:
Window Operation Mode: Use WINDOW setting
INI File Settings
☑ Save and Restore Window Location
Browse Box Prompts
Locator: Step
Update Buttons
Update Procedure UpdateMovies

playing and hiding it several thousand times. CW is about 60 percent faster than VB at this task-a visibly noticeable difference.

TopSpeed recently released an interim upgrade to CW, making the current version 1.001. The interim releases generally fix as many bugs as the company has been able to tackle and also add requested features if possible. TopSpeed makes these interims available fairly frequently, which many in the Clarion community like because bugs are getting fixed-but many others dislike because of the time required to test and validate the new tool.

In CW version 1.001, and in the initial release, the back end (primarily the compiler) has shown great stability and few bugs. In short, the compiler is solid. The front end (the IDE, AppGen, and all the visual tools) is still a bit creaky, rather typical of a vendor's first Windows product. The biggest problem with the CW front end is its user interface (Ul), which lacks the refinement of other Windows products that have been around longer. The interface problem extends to the window editor, which does its job but which could be much smoother, more intuitive, and more powerful. One Ul quirk: Topspeed's developers must have done their development work on high resolution machines, because a number of dialog boxes don't fit on the screen in 4Ox480 VGA resolution! TopSpeed told me that its upcoming version 1.5 will have major improvements in all aspects of the Ul (no release date for 1.5 has been set).

Despite interface difficulties and the somewhat stodgy window editor, Clarion for Windows is capable of building an application whose Ul can go toe-to-toe with the best Windows apps. The first commercial application written in CW has been released. It is called Time Matters for Windows, a time, event, contact, and case management program for attorneys from Data.TXT Corporation in Miami. A slide show demo disk is available that shows clearly what can be done with CW; if interested, ask TopSpeed to get you the demo. Time Matters is also an exemplar of all that is good about Clarion. The program works standalone or in the networked, workgroup environment. On the network, the Windows version coexists with the older OS version, sharing the same data from the same files at the same time. These capabilities spring, in large measure, from Clarion itself.

In spite of all this, Clarion for Windows remains the only Windows development tool that delivers high-performance code; generalized database access; a potent, customizable application generator; a complete set of visual tools; and a strong IDE/ project system all in a single product. It is arguably the best Windows development tool on the market today.

HANDS ON

Clarion for Windows 1.001

■ TopSpeed Corp., 150 East Sample Rd., Pompano Beach, FL 33064-9990; 800-354-4444, 305-785-4555, or fax 305-946-1650.
■ Pricing: $1299; AS/400 Connect: $3999 per server. Additional SQL drivers (DRDA, Oracle, and Sybase) will be released throughout 1995. The 32-bit version of Clarion for Windows is scheduled for release in the fourth quarter of 1995.
■ Minimum requirements: Windows-capable computer, 4MB of RAM, 80386, 6MB of free disk space, and a mouse.

Clarion for Windows is an application development system for Windows, with a rich development environment, a native-code compiler, and a built-in linker. Its stated purpose is to let developers create "blazingly fast Windows applications quickly. While Clarion has traditionally been used for database application development, designer Bruce Barrington has always aspired to make Clarion a general-purpose language. With Clarion for Windows, he is most of the way there. While Clarion might appear to compete with Borland's dBASE for Windows, Microsoft's FoxPro, and Computer Associate's CA-Visual Objects, you can consider Clarion for applications that you would normally write in Visual Basic, C++, or Borland Delphi.

Clarion for Windows introduces a new database format called TopSpeed, and also supports the CA-Clipper, dBASE, and FoxPro flavors of .dbf, Paradox, Btrieve, ASCII, and other formats. All these formats are supported directly, and they are buttressed by standard ODBC access to other databases. A driver for AS/400 is available from TopSpeed ($3999 per server), while drivers for DRDA, Oracle, and Sybase will be released later this year.

First Impressions

While performing the installation, I read through Bruce Barrington's notes on language design. A few years ago I read a previous version of this document and found it illuminating to read about the author's transition to Windows. I am a believer in playing into a language rather than against it, and I wish that all language designers would take the time to write a similar document. It illuminates the designer's origins, intentions, perspective, and goals in a way no "Getting Started" pamphlet can.

The 12 sample programs are supplied as executables, so I ran them even before I ran Clarion itself. Their quick load times impressed me, so I popped into File Manager to browse the directories. 'Me largest sample program, RELATION.EXE, which shows off Clarion's multitable data windows, was a mere 9OKB. This program creates a handful of tables and then populates them with artificial data, displaying them in a series of related scrolling lists. (See Figure 2, page 32.) Besides the executable, at least one other file (CWRUN.DLL, 562KB) is required, but, even so, this is far smaller than the equivalent program in CA-Visual Objects or Microsoft FoxPro , both of which demand approximately 4MB.

Development Environment

Clarion for Windows thinks in terms of applications and components, presenting the application object as an indented outline that reflects the calling structure of the various components. From the basic shell, you can reach any of the tools with a mouse click or two.

The Clarion environment consists of seven components housed in a multiple document interface (MDI) shell window: a dictionary editor, window formatter, report formatter, text editor, formula editor, application generator, and source-level d bugger. The dictionary editor maintains the application's data dictionary, which houses all details concerning tables, indexes, relationships, fields, and views.

You use the window formatter to create menus and reports as well as windows the entire visual dimension of your application. (It surprises me that more software does not work this way. After all, the differences between forms and reports are so trivial, they could be encapsulated in a template an approach TopSpeed takes.)The window formatter provides the standard range of control objects, and also supports VBX and custom controls. However, it lacks some of the cosmetic capabilities of other window painters, such as definable field borders (3D, recessed, or shadowed). It also doesn't let you tag a group of fields to load simultaneously.

Clarion's text editor performs color coded syntax highlighting, but it lacks programmer-oriented facilities such as regular-expression searching. The formula editor vaguely resembles tools such as FoxPro's Query Builder, except that it is generalized to create any statement resulting in a value, so you can use it to create filter expressions, calculated fields, and other expressions. The application generator automatically writes a project file for the application, containing compile and link options. and other properties.

Beneath the IDE (interactive development environment) and its various editors, an application consists of two files: a file containing the application's components (called the APP file), and a data dictionary (called the DCT file) containing all the information about tables, indexes, keys, fields, relationships, views, and validations.

To debug an application, you click the Debug button, set breakpoints near the problematic code, and inspect the values at those points in the program. You can open multiple source files as you investigate the problems. Given the large proportion of generated code and the formula editor, I have so far had little occasion to work with the debugger. You can run your application from within the development environment, and, because it opens an application window, you are still free to browse the APP file as you test the program.

You don't have to give any consideration to managing the windows in an MDI application. Starting with the application window and its menu, you attach dialogs and data windows to menu options, and don't have to give it another thought. Clarion's application framework handles the complexities automatically.

For the most part, I found the development environment intuitive and responsive. As I began to develop my test application, however, it became obvious that the environment is not quite as modeless as it appears. Although you can open several components at once, you can only open one instance of each. Ibis prevents you from editing two windows at once, for example. More serious (and more puzzling, because it's undocumented), you cannot open the data dictionary while the APP file is open.

Language, Documentation, and Performance

Beneath the Clarion environment are the high-level. Clarion language itself and a template language that you can use to tailor the generated source code. Most Clarion users will probably never get around to writing their own templates, but for advanced programmers, the gateway is there.

Because my language roots are in Xbase and C, the Clarion syntax is new to me. I was pleased to discover that Clarion allows C-style in-line assignments (x += y) and supports a wide range of data types for precise control. Clarion does not support pointer variables; instead, it provides C++-style reference variables. (A reference variable contains the data type and the identity of a variable.) The language also supports function and procedure prototyping, so you can eliminate many common errors at compile time rather than at runtime. Given the enormous percentage of generated code in a given application, I found myself quite willing to learn the new syntax.

One well-conceived aspect of Clarion is threaded table access, which lets you open n independent instances of a table. Enabling threaded access is exceedingly trivial in Clarion: You click a checkbox with the mouse. (See Figure 3.)

The printed documentation consists of three well-written volumes that are much improved over the earlier DOS versions. With these and the 12 sample applications, programmers new to Clarion will have little trouble getting up to speed.

Thanks to a native-code compiler backed by dedicated database drivers, Clarion for Windows applications are blazingly fast far outpacing interpreters such as dBASE for Windows and PowerBuilder. In terms of table access speeds, the TopSpeed data format seems fastest. The dedicated drivers for. dbf files are considerably faster than going through ODBC. Clarion for Windows applications are particularly fast in their displays, with none of the lurching and control-by-control display characteristics of, for example, FoxPro.

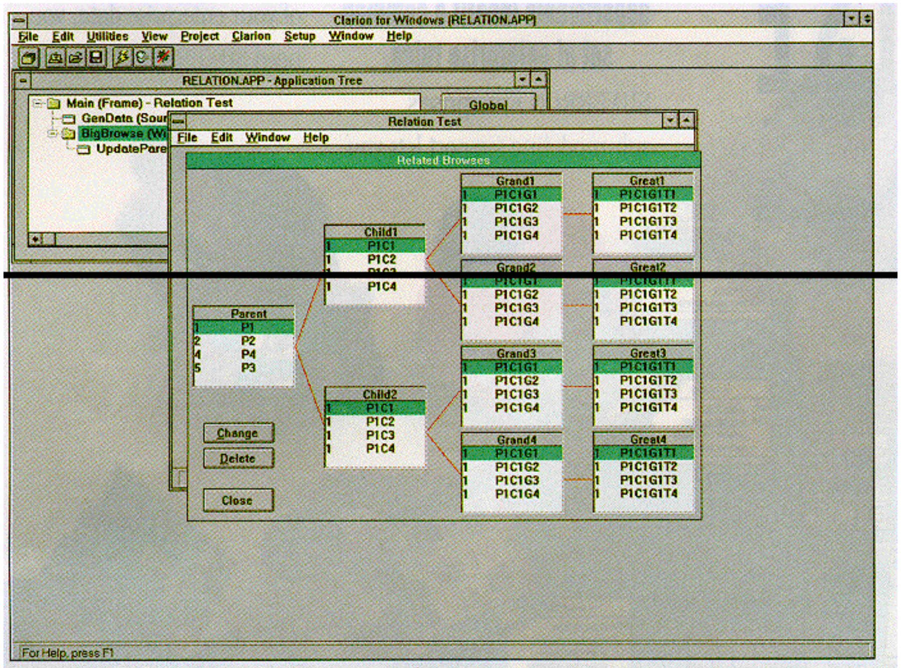
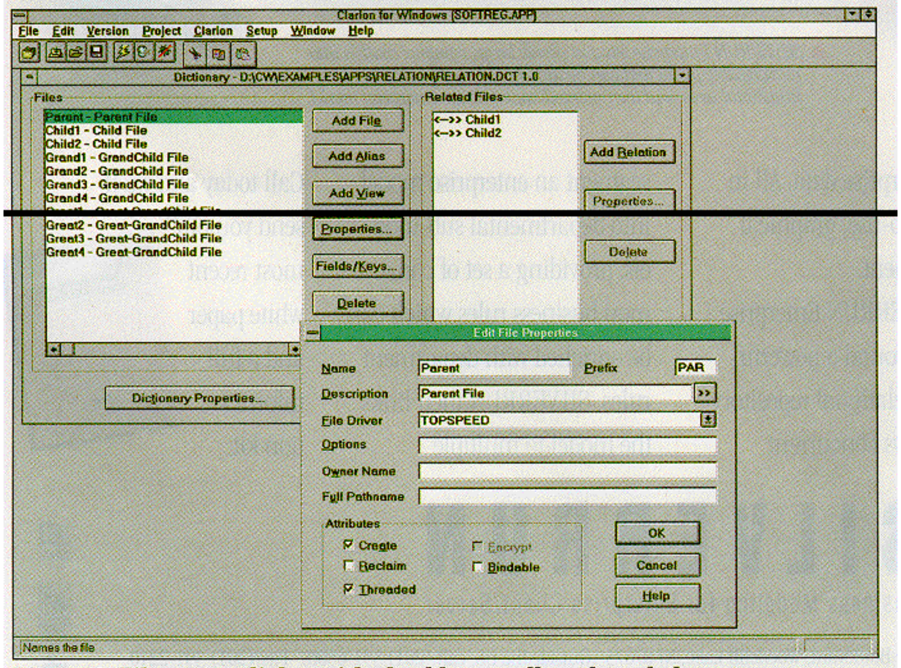


Figure 2. Relation.exe in browse mode.



Files:
Parent - Parent File
Child1 - Child File
Child2 - Child File
Grand1 - GrandChild File
Grand2 - GrandChild File
Grand3 - GrandChild File
Grand4 - GrandChild File
Great1 - Great-GrandChild File
Great2 - Great-GrandChild File
Great3 - Great-GrandChild File
Great4 - Great-GrandChild File

Related Files:
<—>> Child1
<—>> Child2

Name: Parent
Prefix: PAR
Description: Parent File
File Driver: TOPSPEED
Options:
Owner Name:
Full Pathname:
Attributes: [✓ Create] Reclaim [✓ Threaded] Encrypt Bindable

Figure 3. File access dialog with checkbox to allow threaded access.

Criticisms

The window formatter is too spare, lacking the basic abilities I've already mentioned. The Quick Start, application wizard insists on creating the table that it's going to use, thus precluding the use of existing databases. Furthermore, it'. restricted to string and numeric fields, disallowing dates, memos, or logicals. After some experimentation, I discovered ways to work around this limitation, such as first creating only the supported fields, and then modifying the table structure and regenerating it afterward, but this process is silly. Without these basic abilities, Quick Start is terrific fun for the first day or two, after which you'll never bother to use it again.

Aside from color-coded syntax, the source code editor offers only the most basic search and replace facilities, with a few adjustable settings such as smart indentation. In lieu of a genuine programmer's editor, it would very nice if you could substitute your tool of choice. Although in my limited tests I did not hit the ceiling in designing reports, I would like to know that I can substitute the reporter of my choice. You could certainly write an application that calls an external report executor such as Crystal Reports, but you would have to create those reports externally, and they would not be able to draw from the data dictionary.

Although Clarion for Windows supports DDE as both client and server, it does not support OLE 2.0., A TopSpeed Corp. representative informed me that OLE support would be included in the 32bit version, due for release later this year.

A Winner

With Clarion for Windows, TopSpeed Corp. has delivered a rapid application development environment that lets developers work at a very high level, yet compiles applications to native code. Despite the shortcomings I've noted, this is an impressive release. judging by the traffic in the Clarion forum on CompuServe, the bugs are few and minor. With only a quick look, its impossible to tell precisely where the outer limits are, but at this point I feel that I could comfortably use Clarion for Windows for anything I might have written in CA-Clipper.

Although it costs more than dBASE for Windows, Paradox, Access, or FoxPro, Clarion for Windows offers Windows developers two strong benefits: rapid application development and native-code performance. If you're looking for a pure object-oriented product, look elsewhere. n the other hand, I suspect that many developers are craving a Windows application development tool that does not demand total immersion in OOP. In the short time I've had Clarion for Windows, I've already moved two of my CA-Clipper applications to Windows. TopSpeed Corp. as a winner here.

Arthur Fuller teaches and writes about database and object technology. He is author of the Dynamics of Clipper books (Business One Books, 1989 and 1992). You can reach Arthur via CompuServe at 76506,1301, or via the Internet at artful@passport.ca.

Clarion's Windows Version Rings Clear

TopSpeed promised a solid Clarion for Windows, and they came through. Here's a look at this versatile C-like Windows application development language.

By Craig M. Bobchin

Clarion has never had presence in the database market like dBASE or Paradox, but has always had a small, dedicated, vocal group of adherents. Still, it's considered among one of the more powerful and flexible application development environments for DOS. Users have been waiting for the Windows version since TopSpeed Corp. announced it at the 1994 Clarion User Conference. (In October 1994, Clarion Software changed their name to TopSpeed Corp.)

TopSpeed Corp. tested Clarion for Windows for almost five months and has released a solid and impressive program. Clarion for Windows still holds to the Clarion way of performing tasks, so those jumping from DOS should feel comfortable, while new users who are familiar with Windows applications development environments may have to make slight adjustments until they get used to Clarion's method of application development.

Clarion for Windows 1.0, US$1,299
TopSpeed Corp.
150 E. Sample Road
Pompano Beach, FL 33064-9990
800-354-4444, (305)785-4555
Fax (305)946-1650

Focus on language

While Clarion for Windows ostensibly competes with Microsoft Access, Borland Paradox for Windows, and others, its true competition is Microsoft Visual Basic and Visual C++, and the upcoming Borland Delphi. Clarion for Windows is a business-oriented programming language that compiles to an executable (.EXE) file. You distribute the EXE and dynamic link library (.DLL) files to users. This doesn't mean you develop in a text editor. On the contrary, Clarion for Windows ships with a full-featured integrated development environment (IDE), which includes, along with a text editor, screen and report generators, and menu and toolbar builders.

You can get from the text editor to the screen and report generators at the touch of a hot key. If you're editing the source code for a data entry screen and you want to paint the screen, press Ctrl+F – you're now in the screen generator. Any changes you make are updated in the source code when you save the screen. The same holds true for reports.

Extensible Foundations

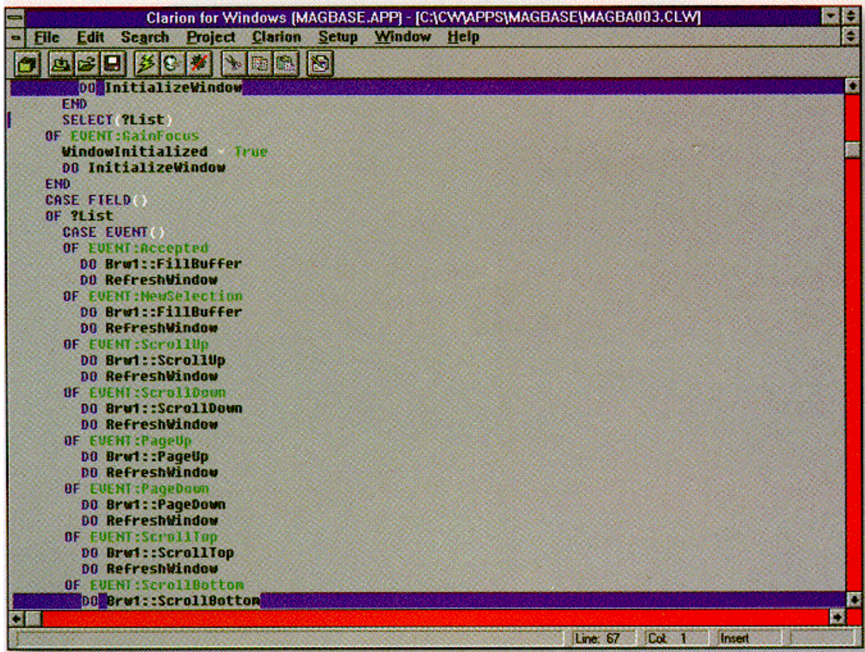
Clarion for Windows actually contains two languages. The first is the Clarion language you see most of the time. Clarion's second language is the Template language. Clarion for Windows uses templates to create screens, reports, and menus. The templates, which are source code modules with boilerplate code, tell the application generator how to generate code for the screens, reports, and menus. You can modify the templates easily. In fact, the ease with which templates can be modified have resulted in the creation of a cottage industry of third-party template providers (Boxsoft Corp. and Power Craft, for instance). These companies, with more to follow, provide templates with functionality that isn't present in the default Clarion for Windows templates.

Data, data everywhere

Clarion for Windows handles data for practically any database format. It has drivers for Xbase .DBF, Paradox .DB, Btrieve, Clarion for DOS, BASIC, ASCII, and TopSpeed's new proprietary file format. You can also hook up to other databases, like Microsoft Access, using an ODBC driver. TopSpeed also sells drivers for AS/ 400, Oracle, and SYBASE SQL Server.

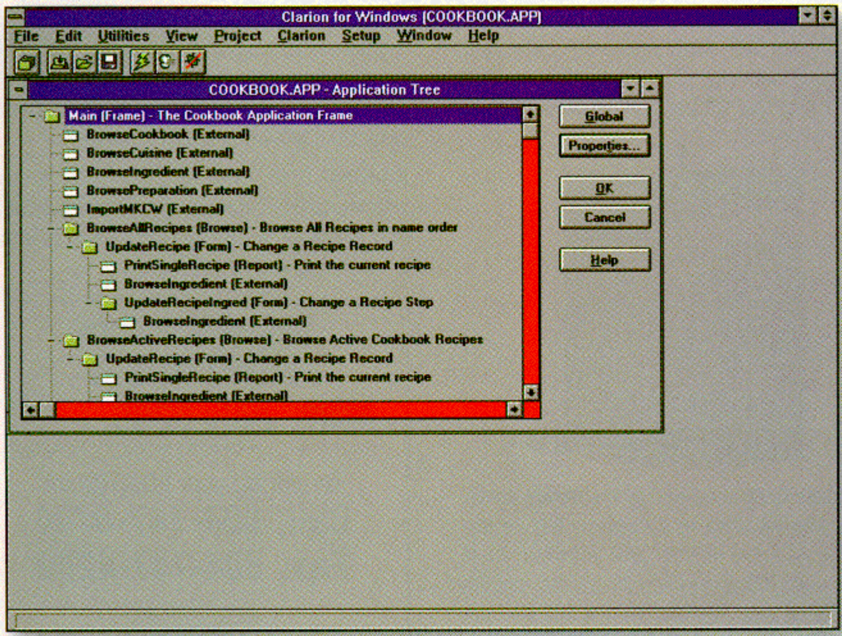
Craig Bobchin is owner of CMB Systems Design, a microcomputer consulting firm specializing in database development. He has a B.S. in Information Systems from the University of Redlands. CompuServe 76040,500.

Figure 1—Clarion for Windows menu editor.


Clarion for Windows [MAGBASE.APP] - [C:\CW\APPS\MAGBASE\MAGBA003.CLW]
File Edit Search Project Clarion Setup Window Help
Line: 67 Col: 1 Insert

```
        DO InitializeWindow
     END
     SELECT(?List)
   OF EVENT:GainFocus
     WindowInitialized = True
     DO InitializeWindow
   END
   CASE FIELD()
   OF ?List
     CASE EVENT()
     OF EVENT:Accepted
       DO Brw1::FillBuffer
       DO RefreshWindow
     OF EVENT:NewSelection
       DO Brw1::FillBuffer
       DO RefreshWindow
     OF EVENT:ScrollUp
       DO Brw1::ScrollUp
       DO RefreshWindow
     OF EVENT:ScrollDown
       DO Brw1::ScrollDown
       DO RefreshWindow
     OF EVENT:PageUp
       DO Brw1::PageUp
       DO RefreshWindow
     OF EVENT:PageDown
       DO Brw1::PageDown
       DO RefreshWindow
     OF EVENT:ScrollTop
       DO Brw1::ScrollTop
       DO RefreshWindow
     OF EVENT:ScrollBottom
       DO Brw1::ScrollBottom
```

Figure 2—Clarion for Windows Application Tree.


Clarion for Windows [COOKBOOK.APP]
File Edit Utilities View Project Clarion Setup Window Help
COOKBOOK.APP - Application Tree
Global
Properties...
OK
Cancel
Help

```
- Main (Frame) - The Cookbook Application Frame
    BrowseCookbook (External)
    BrowseCuisine (External)
    BrowseIngredient (External)
    BrowsePreparation (External)
    ImportMKCW (External)
  - BrowseAllRecipes (Browse) - Browse All Recipes in name order
    - UpdateRecipe (Form) - Change a Recipe Record
        PrintSingleRecipe (Report) - Print the current recipe
        BrowseIngredient (External)
      - UpdateRecipeIngred (Form) - Change a Recipe Step
          BrowseIngredient (External)
  - BrowseActiveRecipes (Browse) - Browse Active Cookbook Recipes
    - UpdateRecipe (Form) - Change a Recipe Record
        PrintSingleRecipe (Report) - Print the current recipe
        BrowseIngredient (External)
```

You manage data through the Clarion for Windows data dictionary, where you specify file keys and indexes. Keys are updated automatically during record addition, deletion, or modification. Indexes have to be built each time they're accessed.

The data dictionary lets you specify one-to-many, one-to-one, and many-to-one relationships between files. While the creation of relationships isn't done graphically, as in Paradox for Windows or Access, it's still easy to do, and you can specify referential integrity at the same time. For referential integrity you have several choices: cascaded updates and deletes, to roll out changes to any affected child records; restricted, to prevent deletion of child records; and clear, to clear the value of a foreign key if changes to the child record with the primary key are made (clear generates no referential integrity code).

Clarion is RAD

Clarion for Windows excels at rapid application development. It has a QuickStart feature that lets you create an application with one table and as many keys as you want with almost no coding. The application generator creates a browse window and report for each key and a data entry screen for the table. You can then compile this into an application.

QuickStart is fine for new applications, but it doesn't let you use an existing file as a starting point for a new application. Allowing this would make converting applications from other database systems easier. QuickStart is also limited only to String and Numeric data fields. I'd like to see support for more field types such as Memo.

If you do your development without QuickStart, you can set up files and screens any way you want. When you develop an application, you specify an MDI (multiple document interface) frame-essentially, your main menu using the menu editor (figure 1). The menu editor is easy to use and intuitive enough that you won't use the manuals too often.

After you specify menu choices and procedures to run, you can pull up a tree structure to see how your application fits together (figure 2). Double-clicking on a procedure lets you define it. You can create Windows, Reports, Browse Boxes, call an external program, or hand code your procedures. This is where templates come in. All procedures, except for hand-coded ones and external programs, are contained in templates.

These templates are boilerplate source code modules that tell the application generator how to create the code for them. As you develop the procedures, you can jump between the source code version and the graphical version. Changes made in one are immediately made in the other.

In creating your windows, you have the option of embedding source code in virtually any location on any control. This is useful for pre and post-field processing, or for making any control change its properties or actions based on conditions. Also, Clarion for Windows thoughtfully gives you Control Templates, templates for writing often-used controls, like buttons.

Since most applications rely heavily on a pleasing user interface, it's a great advantage if a development system has a good set of tools to create interface elements, such as data entry screens, browse boxes, etc. Clarion for Windows as what's called the Window Formatter (figure 3), a screen painter that is used as the starting point for creating your own browse boxes, data entry screens, popup lists, and reports. That's right; the Window Formatter also creates reports. Only the underlying template is different.

The Window Formatter is good, if not spectacular. It has a few minor flaws that make it less than perfect. One annoyance is apparent when you try to add more than one field at a time to a window or report-you can only choose one field at a time. Although there is a multiple populate choice, it only allows you to put down one field. Then you have to choose another to put on the screen. This becomes tiresome after a few minutes and is contrary to the way most DOS and Windows database management systems work.

Another flaw is the inability to jump from the Window Formatter to the data dictionary to add or modify a field. To add fields to a table, you have to close what you are doing, then open the data dictionary. This is both time consuming and cumbersome.

Once you've developed all of your procedures, the next step is to compile and link the app into an EXE file. It's simple; you just click on the Make toolbar button. You can also use the Clarion for Windows debugger to help you keep your program clean. The debugger is one of Clarion's best features. It lets you debug as you run the program or as a separate application. The debugger also has a unique feature; if your program breaks the screen that caused the problem, the screen is still fully redrawn. Other Windows debuggers leave the window only partially complete, thereby making it more difficult to trace where the problem lies in the code.

During the compile and link phase, a text file is produced that tells you exactly what files you need to give to users in order for them to run your program. This eliminates any guesswork on your part. In most cases you'll distribute the EXE, a database driver DLL, and the Clarion for Windows runtime DLL. The two DLL files are typically less than 1M, making them very efficient for disk space. It's fair to say most applications you create can be contained on one or two 1.44M disks.

Documentation and help

TopSpeed Corp. came under fire for the poor quality manuals for Clarion Database Developer for DOS; this time, they appear to have learned from their mistakes. The documentation consists of three manuals: a Language Reference, a User Guide, and a slim Getting Started guide hat doubles as a tutorial. There is no printed template language manual, but since this is one most users won't use, help for the template language is provided in an online Windows help file.

All of the manuals are easy to read with helpful hints scattered throughout the text on interface and database design tips. There are also plenty of screen shots to let you tie the text with what you see on the screen.
There are five help files that cover everything from a guide to the example programs to the template language, to ore than a dozen example programs showing the versatility of Clarion for Windows. Included are a calculator, strategy game, and a couple of sample database applications.

Wish list

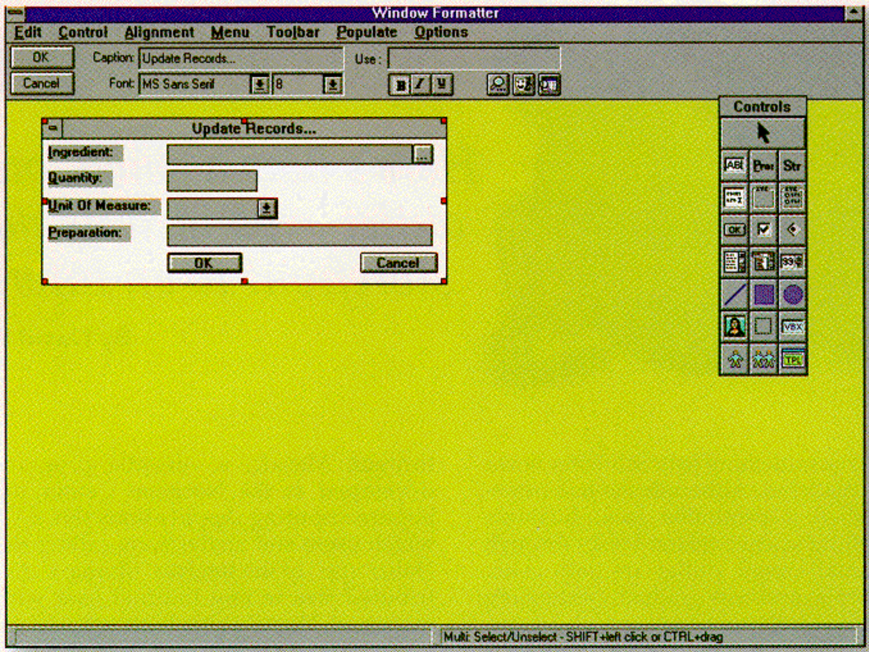
TopSpeed created a very developer friendly environment with Clarion for Windows. You'll encounter a few tumbling blocks as you create larger applications, but none of them are large enough to prevent you from creating mall, fast applications. One stumbling block is the text editor; it isn't as friendly as ones for Access or Paradox. You can only open one file at a time, so cutting and pasting from other modules or programs can be an exercise in frustration. On the flip side, most developers won't spend much time in the editor and will opt instead for Clarion's IDE.

I'd also like to see TopSpeed add a user-friendly report generator. As any developer knows, users want more reports or the ability to create ad hoc reports after you've designed an application. The ad hoc reporting capability will likely be provided by third-party tool providers, some of which may include Query By Example (QBE) templates. Clarion for Windows also needs direct access to more file formats, such as Access. You can access Access via ODBC, but this is slow and has limits. As a plus, TopSpeed Corp. sells database drivers for Oracle, Sybase SQL server, and AS/400, so client/ server data access isn't an issue.

Quick assessment

Perhaps the highest praise I can give Clarion for Windows is this: If I were stranded on a desert island and could only take one Windows development tool with me, it would be Clarion for Windows. Clarion for Windows reminds me of an advertisement for the game, Othello: "A minute to learn, a lifetime to master."

Figure 3—Clarion for Windows Window Formatter.


Window Formatter
Edit Control Alignment Menu Toolbar Populate Options
OK
Cancel
Caption: Update Records...
Font: MS Sans Serif 8
Use :
Controls
Update Records...
Ingredient:
Quantity:
Unit Of Measure:
Preparation:
OK
Cancel
Multi: Select/Unselect - SHIFT+left click or CTRL+drag

First Look / Nicholas Petreley

Clarion for Windows a must-have for developers

With TopSpeed's database development tool you can build fast apps — and do it fast

Clarion for Windows is TopSpeed Corp.'s new rapid database application development environ ment for Windows. This is no namby-pamby visual programming environment where your application winds up using interpreted, tokenized, or semicompiled code along with a run-time module. This is the real thing. Clarion is a hybrid set of development tools consisting of visual application development utilities, an event-driven programming language, and a compiler that creates optimized ex-ecutables with execution speeds that rival the output of the best C compilers.

The DOS version of Clarion already has a loyal following of database application developers. If you're one of them, you can skip straight to the phone number at the end. Clarion for Windows is a must-have addition to your development library.

A LANGUAGE LOUD AND CLEAR. For the rest of you, Clarion for Windows is, like the DOS product, a superb set of development tools with which you can quickly whip up the essential pieces of an application. TopSpeed had to make some changes in the language to make Clarion suitable for Windows development, but the original language has proved to be surprisingly adaptable to its new digs. All the core strengths of the language and utilities that won converts to the DOS product are still there.

Granted, the Clarion language and tools are difficult to get used to, but once you begin to master then, you can start churning out professional-quality applications in short order. That's what Clarion is about. I used to be a Clarion developer several years ago, but I'd forgotten most of what I knew since then. It took a few days of struggling with this product to get my lip back. But there was a point after which I hit that Clarion stride, where I could start cranking out the code with unbelievable ease. It will take new users longer to hit that stride, but it's a goal worth attaining.

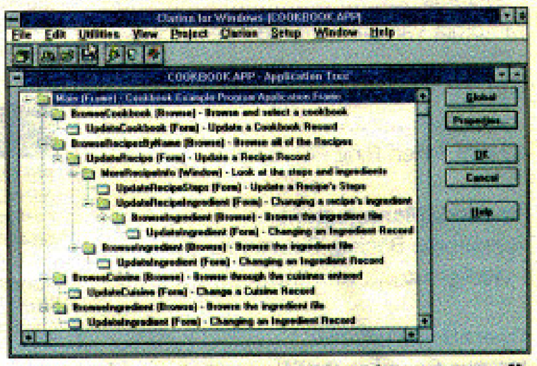
A UTILITARIAN PRODUCT. You can jump right into Clarion and start coding, compiling, and debugging with the interactive debugger. Or you can use some combination of code-writing and development tools to build your application.

The database dictionary utility lets you define fields, tables, default screen format-ting for fields, and table relationships that you expect to use in your application. The tables you create can be several formats or combinations of formats, including the different flavors of dBase, Clipper, FoxPro, Clarion native, TopSpeed database format, and Btrieve, to name a few. You can also use ODBC drivers to connect your Clarion applications to any SQL database for which you can find ODBC drivers.

The hefty Application Generator is a template-driven code generator for building a more sophisticated program. Your application starts with a Windows Multiple Document Interface frame, and then you add modules to it as necessary. The application takes shape as a hierarchical tree of interconnected code modules. Studying the code that the Application Generator creates for each module is a great way to learn your way around the Clarion language.

Each time you define a module, you use the appropriate visual development tool to paint data entry screens and reports. (You can use these visual tools even if you hand-code your application.) Then the Application Generator uses the appropriate template to take your screen and build part of the application around it. The screen formatter is something like a dialog box editor. You drop controls onto the canvas and then define the properties for them, such as which field they will display. (Yes, you can add your own custom VBX controls.) You can also define an action to be taken when the control is used.

Clarion for Windows is, like the DOS product, a superb set of development tools.


```
Clarion for Windows [COOKBOOK.APP]
File  Edit  Utilities  View  Project  Clarion  Setup  Window  Help

COOKBOOK.APP - Application Tree
Main (Frame) - Cookbook Example Program Application Frame
  BrowseCookbook (Browse) - Browse and select a cookbook
    UpdateCookbook (Form) - Update a Cookbook Record
  BrowseRecipesByName (Browse) - Browse all of the Recipes
    UpdateRecipe (Form) - Update a Recipe Record
      MoreRecipeInfo (Window) - Look at the steps and ingredients
        UpdateRecipeSteps (Form) - Update a Recipe's Steps
        UpdateRecipeIngredient (Form) - Changing a recipe's ingredient
          BrowseIngredient (Browse) - Browse the ingredient file
            UpdateIngredient (Form) - Changing an Ingredient Record
      BrowseIngredient (Browse) - Browse the ingredient file
        UpdateIngredient (Form) - Changing an Ingredient Record
  BrowseCuisine (Browse) - Browse through the cuisines entered
    UpdateCuisine (Form) - Change a Cuisine Record
  BrowseIngredient (Browse) - Browse the ingredient file
    UpdateIngredient (Form) - Changing an Ingredient Record
```

The Application Generator treats your application like a hierarchical structure of modules. The names in parentheses indicate what template the program used to create the procedure.

There are several places in the Application Generator where you can add snippets of your own code, so that the code will be inserted into your application every time you make modifica-tions using the Application Generator. You'll want to work that way until you're satisfied with the core of the application, because once you start modifying the source modules directly, you won't be able to go back to the Application Generator without losing your changes.

The templates used by the Application Generator are much like the templates used in the DOS product. You have a handful of screen types and control types to choose from, including browse windows and data entry windows. There is one catch. The applications you generate will always have a typical Clarion look and feel, because the templates force you to think like the template designer. It's not a bad look and feel, but it's always basically the same.

But as the templates are simply text files, third-party vendors have the opportunity to follow up the release of Clarion for Windows with after-market sales of custom templates. Or, you could take a month off work, study the template language yourself, and come up with your own designs.

When you're done using the Application Generator, you have Clarion generate the code for you. You go into the code and tidy up, and then use Clarion to compile and link the code into an optimized application. And not just any application - one that can run hundreds of times faster than what you'll get out of Paradox for Windows, Dbase for Windows, MicrosoftAccess, and all the rest.

Take careful note that we're not talking about data access speed. That will depend entirely on the data driver you choose. What we're talking about here is the sheer speed of the compiled language. An application built with Paradox for Windows can have so much overhead executing instructions that you can watch your application load a list box with entries one by one. The same operation in a Clarion program is instantaneous.

That's the beauty of Clarion for Windows. You get the benefits of a visual environment with plug-and-play database access without having to sacrifice the execution speed you'd get if you wrote your application in C or C++. I recommend it for anyone who takes database applications programming seriously enough to want to use a real language.

Clarion for Windows will sell for an introductory price of $599 for the first 90 days; list price to cost $1,299. It is due to ship this month. TopSpeed Corp., in Pompano Beach, Fla., can be reached at (800) 354-5444 or (305) 785-4555; fax: (305) 946-1650.

Nicholas Petreley is Executive Editor of Reviews & Testing.