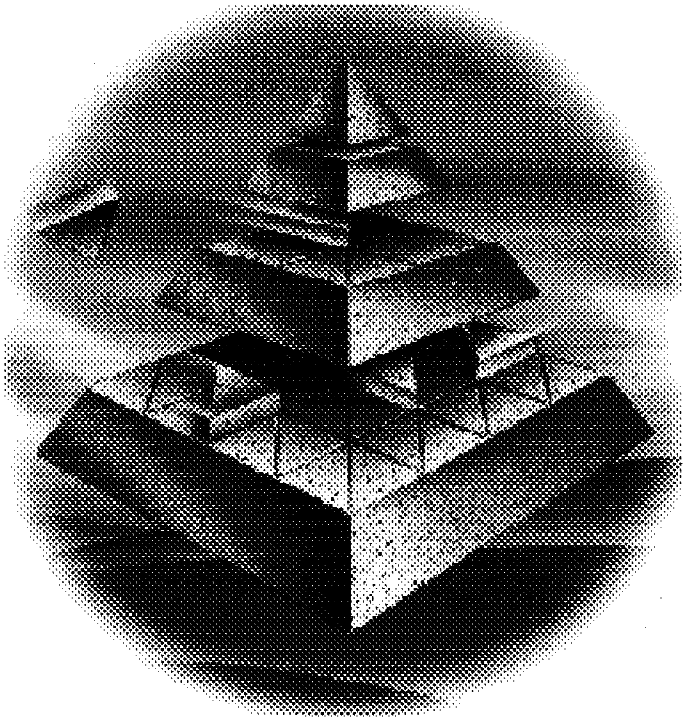


IBM C Set ++ News No.5 May 93

In this issue

- New logo, new title, new product, new prices!
- C Set ++ ™ is announced, page 2
- An update on WorkFrame/2 ™ V2, page 23
- Education news, page 21
- Talk to the world....electronically, page 28
- ColoradOS/2, page 30
- Usability, page 31
- Tips when using ICLCC, page 36

and our usual assortment of reviews, comments, hints, and snippets from the world of OS/2™ and C and C++ compilers.

To celebrate the announcement of C Set ++ we have changed our logo from the C made of miniature OS/2s to the pyramid that represents our product family. We have also changed the title to C Set ++ News.

It's here!

Dateline May 4th, 1993 ....IBM Programming Systems is pleased to announce C Set + + . All the C+ + tools you've been waiting for are now officially orderable, with what we term GA (general availability) May 31th, except Europe, the Middle East, and Africa (June 30th) and Japan (a few months later).

But first a word from our sponsor....

It's certainly with great excitement and pride that we announce C Set + + for OS/2. Our driving force has been an overwhelming desire to satisfy you, the customer, with the best C/C+ + offering available on the OS/2 platform. I have had the pleasure of working with a highly motivated, energized development team organized into several geographical locations. The team includes Toronto, Ontario (C, C+ + , WorkFrame/2, Browser), Lexington, Kentucky (PMD, EXTRA, Browser), Cary, North Carolina (User Interface Class Library), Hawthorn, New York (PMD, EXTRA, Browser), and Boeblingen, West Germany (Collection Classes). We in development also owe a great deal of thanks to our network support and Developer Assistance Program people worldwide for their support of our beta program. In my 11 years of experience with IBM, I have never worked with a more dedicated, enthusiastic team of professionals. This project has involved a lot of hard work, but it has also been great fun!

The extensive beta program for C Set + + has provided us with an excellent opportunity to hear your thoughts on the product as development has progressed. As we deliver C Set + + on May 31 1993, know that we intend to continue providing the first class service and support we began with C Set/2.

The tools we have announced enable you to produce powerful, high performing applications. As we proceed, we will continue to solicit your feedback on how effectively we are satisfying your application development needs. As Bogey once said: "I think this is the beginning of a beautiful friendship"...

Tom Doucher, C Set + + Product Manager

Overview

More technical details appear further on in this newsletter, but first a look at the product from a high level. It's called **C Set** + + . That's the name for the entire package which contains all the tools you need to develop C and C+ + applications to run on OS/2 2.0 and higher. Developed in the PRGS Toronto Laboratory, C Set + + builds on the success of C Set/2 and includes:

A 32-bit C and C+ + compiler

IBM's C Set + + for OS/2 runs under OS/2 and allows professional developers to create 32-bit applications for OS/2 2.0 and higher in both C and C+ + programming languages. This allows programmers to incorporate existing C code into new C+ + programs to create OS/2 executables, libraries, and DLLs, as well as Presentation Manager™ applications. The IBM C+ + compiler produces 32-bit object code, supports development of single and multi-threaded applications and mixed-mode programming in OS/2 (16-bit calls from within 32-bit executables and 32-bit calls from within 16-bit executables) and supports the OS/2 calling conventions.

Optimization

The algorithms implemented in the C Set + + compiler ensure the fastest application speed of any 32-bit OS/2 compiler. The C Set + + compiler provides cross-module optimization, function in-lining and instruction scheduling in addition to the more common global optimization algorithms implemented by other OS/2 compiler vendors. C Set + + for OS/2 performs safe optimization only, optimizing only those portions of code which will lead to increased application performance without side effects. The instruction scheduler allows advanced professionals to target their application for best performance on either i486 or Pentium [1] processors at the flick of a switch.

Standards conformance

With the introduction of C Set + + IBM extends its leadership in delivering standards-conforming C and C Set + + compilers, consistently implemented across all major platforms, from S/370 through AS/400 to AIX and OS/2. The C portion of C Set + + conforms not only to ANSI standards but is NIST certified.

The C+ + part surpasses other compilers by providing a full ARM 3.0 implementation, conforming to the ANSI C+ + draft, and by delivering exception handling and supporting templates. By conforming

[1] Trademark of Intel Corporation

to industry standards, the professional developer can be assured that the compiler will behave in a predictable fashion. In summary:

- ANSI C X3.159-1989 and ISO 9899:1990(1992) conformance (full Plumhall suite)
- NIST validation
- SAA C CPI Level 2 conformance (excluding Record I/O)
- Japanese MIA standards conformance
- C++ Draft Standard X3J16 (AT&T 3.0; full ARM implementation)

C++ class libraries

C Set++ contains the most extensive suite of C++ class libraries or data objects of any C++ offering on the market today. Classes unique to IBM include the PM User Interface class library (which allows developers to create PM applications without needing to know the myriad details of PM) and the Collection classes (a class library which provides a basic set of data structures). Industry standard classes such as the AT&T classes (iostream, task, complex) are available as well. This extensive set of C++ class libraries provides a consistent base from which all user objects can be derived. Objects derived from these base classes can be reused across applications.

Productivity-enhancing tools

C Set++ contains a complete set of OS/2 development tools, including unique features such as the visual Execution Trace Analyzer (EXTRA), the graphical C Set++ class browser and the visual PM debugger.

- The C Set++ class browser, the first full-function browser for OS/2, allows developers to understand the complex semantic and structural relationships between program components in their applications. Through the browser, a programmer can quickly search an entire application for variables, class definitions, and other application objects.
- The Execution Trace Analyzer (EXTRA) allows the developer to locate "hot spots" in code, identify portions of code which take the longest to execute, and to determine how much time various portions of the code take to execute. EXTRA allows professional programmers to observe the behavior of their large and complex applications in response to varying conditions. Through the analyzer, developers can time and tune their code to deliver maximum performance.

- The visual PM debugger helps the professional developer detect and diagnose errors in code developed using C Set + + . Professionals can use the debugger to monitor the behavior of both PM and non-PM applications, single or multithread. Monitoring the PM message queue is also straightforward using the PM debugger.

Open integration through the WorkFrame/2

The compiler, compiler-related tools, and associated utilities which form the C Set + + Tools portion of C Set + + , are knitted together and communicate with each other, through the PM-based WorkFrame/2. Together, the graphical IDE toolset allows developers to organize and manage their complex projects, edit, debug, and then run their applications -- all from within a single environment.

Support & Service

With C Set + + we continue the service you have come to know and appreciate, the standard of service you usually see on the mainframe. All product defects will be fixed free of charge. Customers who identify product defects will know at the time they report the defect when a fix will be in their hands. In some cases, fixes for critical defects will be available as soon as overnight.

As many of you know already, we're accessible through many of the major bulletin board services. By dealing with us this way, you are assured of assistance from knowledgeable support personnel without the need to phone a Support Center. Programmers can correspond directly with us, the development community, and can discuss potential problems directly with the people who wrote the code.

How you order

C Set + + is available on diskette and on CD-ROM. In addition, there are several options for purchasing documentation. The order numbers that follow are ***worldwide*** - the same numbers apply whatever country you are in. However, the prices we quote in the costs section are promotional (and therefore for a limited time) and are US only....and what prices they are! Customers in other countries should contact their local IBM office for pricing.

	3.5 inch dskts	*5.25 inch dskts*	*CD-ROM*
Complete C Set + + package, containing C/C+ + Tools, WorkFrame/2 and Developer's Toolkit for OS/2 2.0	***61G1175***	***61G1425***	***61G1412***
US price	***$175***	***$175***	***$149***

Naturally we offer upgrades from our original C Set/2 product to C Set + + . Here's exactly what to order:

The product you're upgrading from	*The medium you prefer for C Set + +*	*Order this number*	*Price*
WorkSet/2, any medium	***CD-ROM***	***61G1407***	***$129***
WorkSet/2, any medium	***3.5 diskettes***	***61G1402***	***$149***
WorkSet/2, any medium	***5.25 disk-ettes***	***61G1435***	***$149***
C Set/2 (compiler/debugger)	***CD-ROM***	***61G1406***	***$129***
C Set/2 (compiler/debugger)	***3.5 diskettes***	***61G1405***	***$149***
C Set/2 (compiler/debugger)	***5.25 disk-ettes***	***71G1588***	***$149***

Note: Please note that the upgrades are to the complete C Set + + package. You will also notice also that the order number for the option going from C Set/2 to C Set + + on 5.25 diskettes really does start with 71G, not 61G like the others.

WorkFrame/2

If you have Workframe/2 V1.0 and you would like to upgrade to V1.1, here are the two numbers you need:

From	*To*	*Order this number*
WorkFrame/2 V1.0 any medium	*WorkFrame/2 V1.1 3.5 diskettes*	*71G1598*
WorkFrame/2 V1.0 any medium	*WorkFrame/2 V1.1 5.25 diskettes*	*71G1597*

Note: There is no CD-ROM option available for WorkFrame/2 to WorkFrame/2 upgrades.

Documentation

Of course, documentation comes with the product. You can also buy it separately as the C Set + + Library, as well as buy each manual individually. The C Set + + Library is divided into two groups, called, not surprisingly, Group 1 and Group 2. Group 1 manuals are shipped hardcopy with the diskette versions of the product, Group 2 manuals are not. Group 2 manuals are shipped softcopy with the diskette versions and the CD-ROM, i.e. no hardcopy is shipped at all with the CD-ROM.

We'll come to what manuals are in which group in a moment. First, a table showing what was just said, so it's quite clear what manuals you get with each version of C Set + + .

	Diskette version	*CD-ROM version*
C Set + + Library Group 1 61G1439	*Hardcopy*	*Softcopy in Post-Script format*
C Set + + Library Group 2 61G1440	*Softcopy .INF*	*Softcopy .INF*

Ordered separately, Group 1 (61G1439) costs $35 US, Group 2 (61G1440) costs $58 US.

Next, as promised, what manuals come in each group. Group 1 contains:

Number	**Title**
10G6199	Developer's Toolkit for OS/2 2.0 documentation
61G1181	C/C+ + Tools: Programming Guide

61G1184	C/C++ Tools: Debugger Introduction
61G1186	C/C++ Tools: Class Libraries Reference Summary
61G1397	C/C++ Tools: Browser Introduction
61G1398	C/C++ Tools: Execution Trace Analyzer Introduction
61G1428	WorkFrame/2 Introduction
61G1441	C/C++ Tools: Reference Summary
71G2230	C/C++ Tools: IBM Class Library: User Interface - User's Guide

Group 2 contains the following:

Number	**Title**
61G1178	C/C++ Tools: IBM Class Library: Collection Classes -
61G1179	C/C++ Tools: IBM Class Library: User Interface - Reference
61G1180	C/C++ Tools: Standard Classes Reference
61G1183	C/C++ Tools: C Library Reference
61G1185	C/C++ Tools: C++ Language Reference
61G1399	C/C++ Tools: C Language Reference

You'll notice there's no reference to .BOO files. Nor to LIST3820 format. Neither are available with C Set++. We understand the OS/2 folks are working to provide BookManager format of their documentation, likely on a CD-ROM, and we expect to join them on what we affectionately refer to as a doc-ROM because it contains documentation. If you need .BOO files, this would be the CD for you. No LIST3820 versions of the documentation are available at all.

What you pay

Now you know what to order, and what it will cost you. Be aware these prices are for a limited time only. List price is $595. Any customer can acquire C Set++ at the special, introductory price of $175 on 3.5" and 5.25" diskettes, and $149 on CD-ROM, or upgrade from IBM's C Set/2 or Workset/2 for just $149 for 3.5" and 5.25" diskettes, and $129 for CD-ROM. However, if you order from IBM, orders must be placed on or before August 31st, 1993, and can only

be made in the US through 1-800-3IBM-OS2 (credit card) or 1-800-IBM-CALL (PO). In other countries, please contact your local IBM office.

C Set + + is also available from the following dealers, who may sell for less:

- Programmer's Paradise
- Programmer's Connection
- Programmer's Warehouse
- Programmer's Shop

The Technical Details

The C/C+ + Compiler

IBM's C Set + + runs on OS/2 workstations and allows you to create 32-bit applications for OS/2 2.0 and higher in both C and C+ + . Existing C code can be incorporated into new C+ + programs to create new OS/2 executables, libraries, and DLLs, as well as PM applications. The C Set + + compiler produces 32-bit object code, supports development of single- and multi-threaded applications and mixed-mode programming in OS/2 (16-bit calls from within 32-bit executables), and supports the OS/2 calling convention.

- Calling Conventions
  - Register linkage conventions (OPTLINK)
  - System linkage conventions
  - 32-16 bit run-time coexistence and linkage conventions (Pascal, cdecl, fastcall)
- Extensive run-time library support
  - Static and dynamic C run-time libraries provided
  - Fully reentrant multitasking and single-tasking C and C+ + run-time libraries provided
  - Supports the building of Dynamic Link Libraries (DLLs)
  - Subsystem C and C+ + development libraries
  - 80-bit long double formatted I/O support
  - High Performance File System (HPFS) support
  - Performance-enhancing features

    - — Memory File I/O support (C only)
    - — Virtual Device Driver (VDD) support (C only)
    - — Inlining of selected library functions
    - — Fast and accurate floating point math libraries
    - — Low serialization overhead in multi-threaded libraries
  - Robust exception handling and debugging support
    - — Memory management debug support
    - — Diagnostics provided on abend and other exceptions
    - — Localized exception handling and I/O support for DLLs
- Template support (C++)
- C++ exception handling support
- C++ Class Libraries:
  - IBM C/C++ Tools: Standard Class Library
  - IBM C/C++ Tools: Collection Class Library
  - IBM C/C++ Tools: User Interface Class Library
- Ease of use
  - Hypertext documentation presented through IPF and Book-Manager
  - Different run-time libraries selected via compile options
  - Extensive compile-time messages
    - — Three levels of informational and error messages
    - — LINT-like warning messages, with subsets that can be selectively turned on or off
    - — Individual warning message control

C Set++ Class Libraries

We offer an extensive suite of libraries:

- *IBM C/C++ Tools: Standard Class Library*
  - complex arithmetic library
  - task library
  - iostreams library
- *IBM C/C++ Tools: User Interface Class Library*
  - CUA '91 look and feel support
  - application window framework

- robust SBCS/DBCS string manipulation
- encapsulated DBCS/NLS enabling
- control layout management ("Canvas")
- object-oriented drag/drop
- object-oriented dynamic data exchange (DDE)
- DLL support
- exception handling

- *IBM C/C++ Tools: Collection Class Library*
  - data structures
  - flat, recursive, restricted access
  - ordering
  - key equality access
  - element equality access
  - uniqueness
  - templates
  - exception handling

PM Debugger

The PM Debugger quickly identifies defects in code. You can monitor application behaviour, step through complex, multi-threaded applications, and monitor the PM message queue.

- Debug Session Control Window

  The Debug Session Control window controls the debugging session and provides the following features:

  - Displays the hierarchical relationship between executables (EXEs and DLLs), object files (OBJs), and functions
  - Enables and disables application threads
  - Searches for functions, including overloaded functions, in the application
  - Allows the user to customize the debugging session
  - Allows the user to access all debugger windows

- Source Windows

  The Source window displays a view of the user's application. The window's attributes include:

  - Support of Source, Disassembly, and Mixed views
  - Control of execution with the following commands:
    - Run
    - Halt
    - Step over
    - Step into
    - Step debug
    - Step return

    - — Run to location
    - — Jump to location
  - Setting breakpoints with "point-and-shoot" or menu driven interface. The types of breakpoints supported include:
    - — Line
    - — Function
    - — Address
    - — Change Address - hit when specified memory changes
    - — Load Occurrence - hit when specified DLL is loaded

    Where appropriate, each of these breakpoint types may be made more complex with the addition of any of the following:
    - — Thread specificity
    - — Conditional Expressions
    - — From, To, Every clause
  - Searches for strings contained within the source file
  - Searches for functions, including overloaded functions, in the application
  - Customization of the debugging session

- Stack Windows

  Each thread in an application may have a Stack window associated with it. The Stack window:

  - Displays all functions that are currently on the stack
  - Allows the customization of the display of stack related data
  - Provides a mechanism to display the source window for any function on the stack.

- Register Windows

  Each thread in an application may have a Register window associated with it. The Register window supports:

  - The display and modification of the processor and math co-processor registers
  - The display and modification of individual processor flags

- Storage Windows

  The Storage window allows the user to examine the storage used by the application. In this window, the user may

  - Display and modify storage
  - Customize the way addresses are displayed
    - — 32-bit
    - — 16:16
  - Customize the way the contents are displayed
    - — Character

— Hex
— Float
— Etc.

- Monitor Windows

  There are four different monitor windows which may be used to display variables or expressions. These differ in their relationship with other windows as well as in the length of time that expressions remain in them. Expressions supported, include those using typecasting. From any of these window a user may:

  – Observe the current value of an expression
  – Change the value of an expression
  – Change an expression's representation
  – Expand or contract classes, structures, and arrays

- Inheritance Windows

  The Inheritance window displays a graphical and textual hierarchical representation of all classes contained in an object file (OBJ). To aid in this examination, this window supports:

  – Searching for specific classes
  – Scrolling, zooming, and panning of the graphical representation
  – User customization of the display of the hierarchical relationship

- Class Details Windows

  The Class Details window shows the attributes of a given class including member data, member functions, inheritance relationships, and friend relationships.

- Message Queue Monitor Window

  The Message Queue Monitor window allows the user to watch PM messages as they are processed in the application. This window allows the user to specify:

  – Types of messages presented, including user defined messages
  – Quantity and format of the information presented with each message
  – Window or queue to be monitored

- Window Analysis Window

  The Window Analysis window allows the user to analyze the PM windows contained within the application. This window pro-

vides graphical and textual information on the window's characteristics. These characteristics include but are not limited to:

  - Size
  - Location
  - Window handle
  - Parent
  - Owner

- General Features

  Other features contained within the debugger include:

  - Where - Locates the current execution point in a thread
  - Exception handling control
  - Default representation for data types
  - Multiple font support
  - Contextual help

Class Browser

The C++ class browser [2] allows you to understand the complex relationships between C++ objects. You can also edit the original source of the component you are browsing.

The browser features an easy-to-use graphical user interface. Using individually sizable "windows" with customizable colors and fonts, the browser screen lets the user simultaneously display any combination of:

- List Windows:
  - display lists of program elements (files, functions, variables or classes)
  - Query on:
    - directory
    - file
    - class
    - template
    - function
    - variable
    - enum

[2] See the April 93 issue of this newsletter for a detailed technical article on the browser

— enumerator

— typedef

— macro

- filter (List Kind Options) on specific Query elements to be studied
- filter (Attributes Options) on specific Query and Kind elements to be studied

- Graph Windows:
  - display program relationships in a graphical format:
    - — specify level of detail to be shown on graph
    - — scroll over graph
    - — Options:
      - Graph Kind
      - Graph Filters
      - Class Inheritance-Graph
      - Zooming (in/out)
  - Text Windows:
    - — display and edit source code text of the focused element.

Other browser features include the following:

- Control Window:
  - The Control Window is the starting point for examining program components using the browser. Tasks that apply to the entire browser session, such as browser file loading and unloading and opening new browser windows, are performed in this window. The client area of the Control window displays an icon for each browser window. Icons, displayed in a variety of formats, can be used to manipulate individual windows.
- Browse Actions:
  - *Search Database* - the capability of searching the entire database from any Graph, List or Text View. Performing such a database search is the first step in examining program components code using the browser.
  - *Query Object* - the capability of querying a particular program component in the database, from any Graph, List

or Text View. Performing an object query allows the user to obtain specific information on a particular component.

- *Copy to Object List* - lets the user copy multiple program components to a clipboard which is shared by all Graph, List and Text Views. Program components copied to the Object List can be used in subsequent queries.

- Browse Filters:
  - The user formulates database searches and object queries by specifying a filter. Available filter values depend on the view type and whether the action is Search Database or Query Object. The number of search and query filters for each database search and object query for each view type, is too great to list in this document.
  - The following is a brief description of each filter type:
    - *Scope* - The Scope filter determines the portion of the data in the browser database to which a Search Database or Query Object action applies.
    - *Category* - The Category filter further restricts the data specified by the Scope filter by limiting the types of data to be retrieved by a Search Database or Query Object action.
    - *Attribute* - The Attribute filter imposes further restrictions on the category of data specified by the Scope and Category filters, that are to be applied to a Search Database or Query Object action.

The Execution Trace Analyzer

The analyser is a tool designed to help you tune and understand your programs by monitoring program execution and generating a procedure trace of the run. Subsequently, this trace can be examined by utility programs that graphically display the execution trace.

It offers:

- Statistics (Statistical Summary)
  - Textual report of execution time for each routine:
    - total execution and active times
    - minimum, maximum and average execution times , and variance
    - total number of calls

- Time Line (Timeline Diagram)
  - use to determine which groups of routines or which sequences of events take up most time in the application
  - displays sequence of nested function calls and returns
  - uses time stamp information to place each event along a time dimension
- Procs Plot (Execution Density Diagram)
  - divides entire execution time into slices for each function which logs events
  - time slices colored to indicate percentage time spent in each routine
- Call Trace (Call Nesting Diagram)
  - displays sequence of nested function calls and returns performed by target program
- Call Tree (Dynamic Call Graph)
  - graphical view of target program execution:
    - each called routine shown as a node
    - calls shown as arcs between nodes
    - sizes and colors of nodes depict amount of time spent in the node
    - sizes and colors of arcs depict numbers of calls between nodes

Other analyser features are as follows:

- Correlation
  - correlation of Call Nesting, Timeline and Execution Density diagrams based on event time or time interval
- Data Reduction Techniques
  - techniques to reduce the amount of information displayed:
    - filtering - select a subset of traced events for presentation
    - scaling - change granularity of detail by magnification/reduction
      li.scrolling - scroll over all diagrams to focus on sub-sections of interest

— multiple views - view the traced program's execution from several points-of-view simultaneously

— summarizing

— color coding (as described above)

— pattern recognition - allow coalescing of repetitive patterns

— interaction - information on demand only, via interactive diagrams to reduce amount of information on diagram. For example:

- column labels in the Execution Density diagram
- execution time rulers
- call stack pop-up windows
- execution times in the Dynamic Call graphs
- node expansion/collapsing in the Dynamic Call graph.

WorkFrame/2

IBM WorkFrame/2 V1.0 has now been revised to V1.1, in order to serve as the integration point for the new C/C++ tools we've just described. It is both a tools integrator and a tools starter, and features an SAA/CUA™ conforming user interface.

IBM WorkFrame/2 organizes the programmer's working environment by grouping files into logical units, or projects, each consisting of source files and object files, as well as one target .EXE or .DLL, etc. Base projects can be combined to form composite projects. Each project can be associated with a compiler / debugger / maker / linker.

As a tools integrator, the IBM WorkFrame/2 is designed to allow different, as well as multiple Edit-Compile-Debug components to be plugged in. Interfaces between WorkFrame and each of the edit, compile and debug components, are defined to allow seamless integration. Included is an interface to the 32-bit Resource Compiler which facilitates the building of projects for PM applications.

As a tools starter, the IBM WorkFrame/2 is designed to allow for user tools, as well as the IBM-supplied tools, to be plugged in.

IBM WorkFrame/2 Graphical Build Tools:

MAKEDEP: Make File Creation Utility - creates the MAKE file with which NMAKE (see below) works. The utility consists of a graphical interface and a language-independent DLL which interfaces to a DLL provided with the compiler.

LIB: LIB is a utility with both a *graphical and a command-line* interface, which allows a developer to maintain object libraries. It can be used to:

- Create a library
- Add objects to a library
- Unload objects to any directory
- Change the characteristics of a library
- Generate listing files of different levels of detail
- Display the details of object modules
- Delete objects from a library

IBM Developer's Toolkit for OS/2 2.0

The IBM Developer's Toolkit for OS/2 2.0 includes headers and .LIB files for defining and resolving OS/2 V2 API calls, tools for building programs, API reference information, sample programs to demonstrate the coding of API's and a Kernel Debugger.

Build Tools

NMAKE: NMAKE is a utility that can save time by automating the process of updating project files. NMAKE compares the modification dates for one set of files, the target files, to those of another set of files, the dependent files. If any of the dependent files have changed more recently than the target files, NMAKE executes a specified series of commands.

NMAKE is typically used by specifying a project's executable files as target files and the project's source files as the dependent files. If any of the source files have changed since the executable file was created, NMAKE can issue a command to assemble or compile the changed source files and link them into the executable file.

MKMSGF: MKMSGF converts an error, help, prompt or general text information file to a binary format for display at runtime.

MSGBIND: MSGBIND binds the message file as a data segment to the application's executable file. The MKMSGF and MSGBIND utilities help isolate message text to support, for example, translation of text.

EXEHDR: EXEHDR is a utility which allows a developer to display or change fields in the header of an EXE file.

MARKEXE: MARKEXE marks .EXE files to indicate whether (a) they can handle long file names and (b) they are window-compatible.

IMPLIB: IMPLIB is a utility that creates an import library for C and C+ + . Import libraries are used to resolve an application's external references to subroutines in dynamic link libraries.

PACK: PACK compresses a file or group of files into a single file optimizing disk space and I/O performance on install.

LINK386: LINK386 produces 32-bit .EXE files for C and C+ + .

32-bit PM Resource Tools

DLGEDIT: DLGEDIT is a PM resource editor which allows the creation and modification of dialog boxes for use with other PM programs.

FONTEDIT: FONTEDIT is a PM resource editor which allows the creation and modification of fonts for use with other PM programs.

ICONEDIT: ICONEDIT is a PM resource editor which allows the creation and modification of icons, bit maps and pointers for use with other PM programs.

Information Presentation Facility Compiler (IPFC): Creates .HLP files for panels or viewable .INF files for documents from tagged text files.

32-bit Resource Compiler (RC): Creates a binary file from Resource Editor (DLGBOX, FONTEDIT, ICONEDIT) output, making the resources available to the application.

System Object Model Precompiler (SOMC): Creates input to source programs that make available existing classes, and creates new classes, of objects. Bindings for C source programs are in the Toolkit.

Libraries:

OS2286.LIB: OS2286.LIB - Library against which system calls for 16-bit programs are resolved.

OS2386.LIB: OS2386.LIB - Library against which system calls for 32-bit programs are resolved.

OS2STUB: OS2STUB - Module included with an OS/2 program. At runtime, alerts a DOS user that program is an OS/2 program (and therefore won't run on DOS).

Headers

OS2H: OS2H - directory containing ANSI conforming header files that define the OS/2 API's for C and C++ .

OS2INC: OS2INC - directory containing include files that define the OS/2 API's for ASSEMBLER.

Information and Samples:

On-line (IPF) Information and Samples: On-line information and sample programs provide assistance in exploiting the OS/2 V2 API's in application programs. Additional guidance in the use of API's with references to the sample programs, is provided in the Programming Guide, the Application Design Guide and the SOM Reference; all available in the Technical Library. Printed versions of the on-line API reference information are also in the Technical Library.

KwikINF (KWIKINF): Provides a convenient method of accessing information in the OS/2 online documents by implementing an on-demand text search function.

Debug Support

Kernel Debugger: Included in the Toolkit to assist with Ring 0 debugging. The Kernel Debugger is a set of files and utilities that are installed separately from the tools and samples in the Toolkit.

Note: All Toolkit C++ functionality is provided only on the Toolkit Update Diskette.

Back to school

Having an announcement of a product is all very well, but what about education? You can learn this kind of product through self study, but hands-on is quite possibly the best way to become proficient. How are you going to learn it all?

Skills Dynamics in the US is right up to date with their C and C++ education. They already offer two SRA self-study Object-Oriented Programming courses (these are OO courses, not C++ courses):

- Object-Oriented Programming Fundamentals (32231)
- Designing Object-Oriented Applications (32232)

  Course 32232 prereqs 32231

Building on these courses, Skills Dynamics has 2 new offerings based on C Set++:

Name Introduction to C++ on OS/2 (K3621)

Duration 4.5 days

Audience Experienced application and system programmers.

Abstract After completing this course, you should be able to:

- Use and apply the concept of data abstraction.
- Use and apply the features of C++ to implement an object-oriented program design.
- Design C++ programs which are understandable, portable and maintainable.
- Write a complete application in C++.

Prerequisites Before taking this course, you should be able to write programs in C language using structures, pointers, and dynamic storage.

These skills can be developed by taking C Language Fundamentals (K3604) and C Language Data Structures (K3605).

And for the IBM Class Library: User Interface

Name OS/2 V2 PM Programming Using C Set++ and ICLUI (P1067)

Duration 4.5 days

Audience Experienced C++ application programmers

Abstract This course provides the basic concepts that C++ programmers and application developers need to program OS/2 V2 PM applications using the C++ Language. This course uses the IBM Class

Library: User Interface (ICL:UI) to create the PM windows and controls needed for your applications.

Prerequisites Before taking this course, you should be able to write programs in the C++ language. These skills can be developed by taking Introduction to C++ on OS/2 (K3621) or equivalent job experience. You should understand the basic functions of OS/2 V2. A basic understanding of how PM works is beneficial, but not required.

Also planned is an advanced course for the IBM Class Library: User Interface. This will include advanced topics such as DDE, which is not covered in P1067.

If you're interested in these offerings, contact Skill Dynamics at 1-800-IBM-TEACh in the USA (which translates into 1-800-426-8322) or in Canada call 1-800-661-2131.

But I'm in Europe, you may say. Or Asia. Non-US customers may attend US classes, but you need to contact your local IBM office for assistance with course identification, booking, and costs. Alternatively, they may be able to help with information on local country arrangements for C and C++ education offerings.

So what about WorkFrame/2?

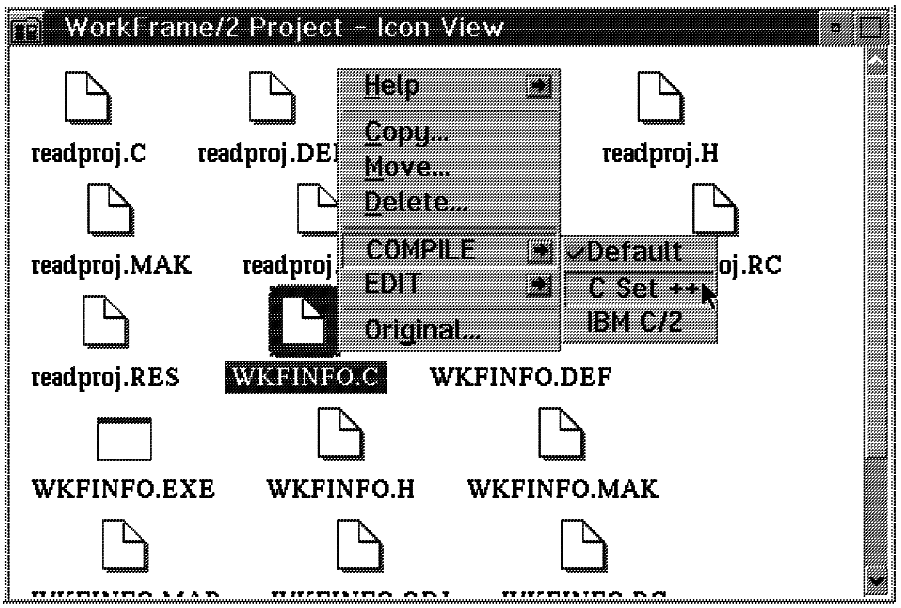
So far this newsletter has talked about C Set++, mentioning WorkFrame/2 just a little. WorkFrame/2 is the open, language independent development environment included with C Set/2 and C Set++. WorkFrame/2 V1 has been very well received, and people are now asking what we're planning to do next .... why is the beta on the CD-ROM still version 1? Why isn't WorkFrame/2 more workplace shell aware? Are any new versions of WorkFrame/2 planned?

We're pleased to say that we are planning a new version of WorkFrame/2 and it will be fully integrated as a workplace shell application. In this article we'll review some of the new features that WorkFrame/2 will provide, and although it's not available yet, we think this will whet your appetite.

Interface

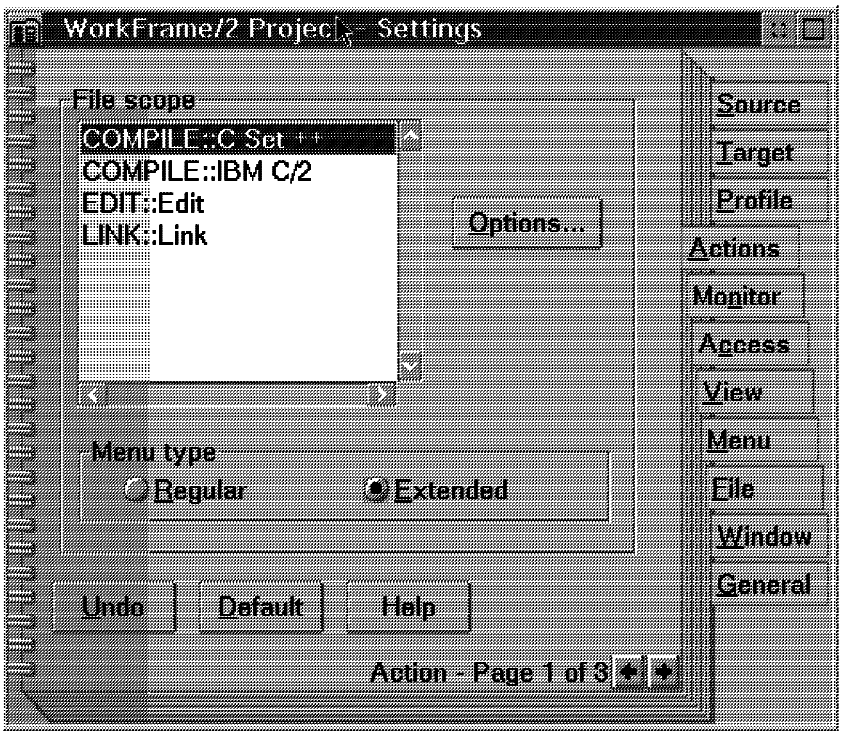
The most obvious change to the new WorkFrame/2 is the interface. Gone is the standard application look, with its title bar and menu. WorkFrame/2 V2 is completely integrated with the workplace shell (WPS) < see below> . A base project is now represented by a WPS folder, and can be placed anywhere on your desktop. A composite project is comprised of multiple base projects, and also appears as a folder. Actions are now started from a context sensitive pop-up menu, or by a user selectable accelerator key.

Although we want the new WorkFrame/2 to be as integrated with the WPS as possible, we know that performance is key for developers. The result is an application that is consistent and easy to use, but does not suffer any significant performance loss over version 1. The complete interface has undergone extensive usability tests, and both new and experienced users are full of praise.



Base Projects

Project options are controlled from a settings notebook (see below). One of the most asked-for features was multiple directory support, and this capability has been added. Multiple file masks are also supported, giving you incredible flexibility in defining the contents of your projects.



Whereas WorkFrame/2 V1 required all your project information to be placed in one directory, you can now locate your projects anywhere. You can have all projects stored in one directory if you wish, or you can share some (or all) projects with others by placing them on the LAN. The choice is up to you - and when you've finished setting up, shadow the projects to your desktop, and access to them is as easy as double clicking on an icon.

As mentioned above, a base project now appears as a WPS folder. This means you can select the font simply by dragging from the font palette. Selecting new colours is just as easy, as is changing the style of your view from "flowed" to "non-flowed", or changing the icons from normal size to small or invisible. You can open a project as an icon view, or a details view, customize the menus that appear, and associate comments, key phrases, and history information with any project. WPS integration doesn't just mean consistency -- it means lots of new features as well!

Composite Projects

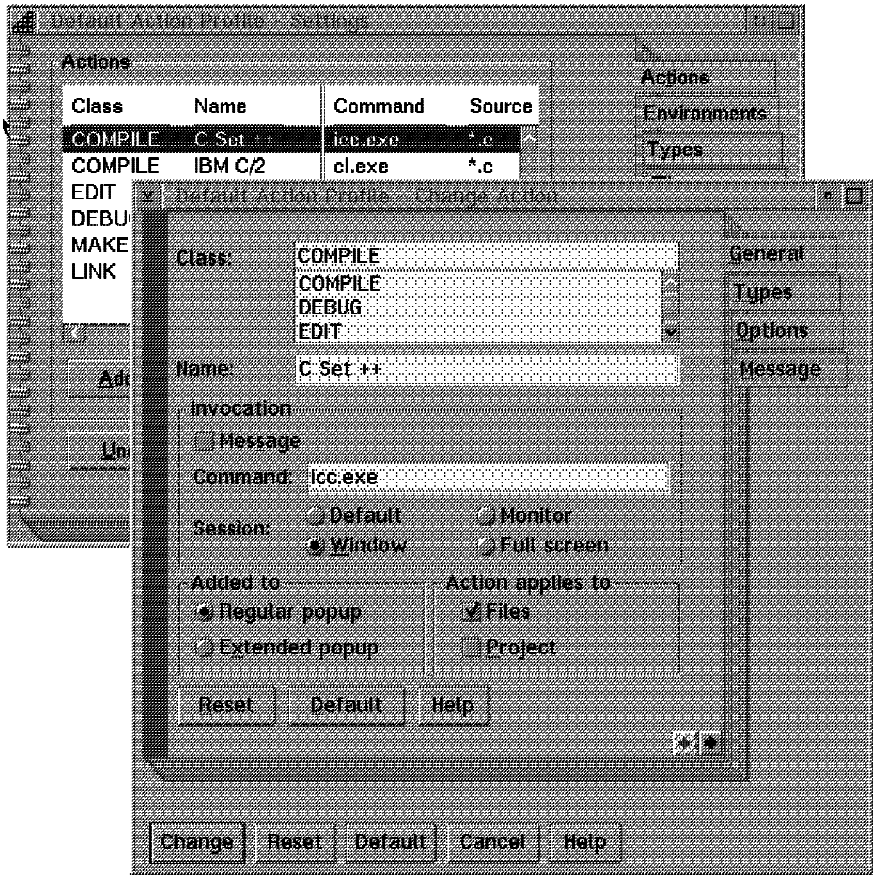
A composite project provides the ability to group folders containing projects, or other composite projects. One obvious benefit of composite projects is to improve the organization of your projects. You can, however, sequence the contents of a composite project, and execute actions against the projects contained within in a specified order. This will allow you to make a complete hierarchy of projects in one step, for example.

Actions Profiles

Another feature that was often requested of WorkFrame/2 was the ability to install new actions. WorkFrame/2 V1 allowed multiple "language profiles", where each language profile defined a compiler, a linker, a debugger, and a make facility. Although you could create new language profiles, each profile was limited to one of each of the above tools. A common problem with this method was the inability to efficiently support language pre-processors.

WorkFrame/2 V2 now supports "actions profiles", with the change of name indicating the increased flexibility. You can install any action in an action profile, and you can have as many actions profiles as you wish. WorkFrame/2 provides a default actions profile, the contents of which apply to every project -- you would probably add your editor to the default actions profile, for example. Each project can also list a specific actions profile, which will supplement the default actions.

Each actions profile (including the default) can contain an unlimited number of actions < see below> . Each action consists of a class name, a descriptive name, the actual command name, a list of source and target filemasks, information about how to run the action, and what options DLL to use for the action.



Executing actions

The action source filemasks are used by the base project, when you initiate a pop-up menu. The base project will examine all selected files, and include the class name of every applicable action in the pop-up menu. The class name is a cascade menu item (similar to the open item on pop-up menus), and the cascaded menu lists the descriptive name of every applicable action in that class.

As complicated as this sounds, it results in a very intuitive method of executing actions. If, for example, you selected some .C, .CPP, and .FOR files, the pop-up menu would contain the class COMPILE. Simply selecting that menu item would result in every selected file being automatically compiled by the correct compiler. Alternatively, if you have multiple C compilers installed, you could invoke the cascade menu to select exactly which one to use.

Project scope actions are also supported. These are actions that require no files to be selected - an example of which could be the make action. Project scope actions are also available from the pop-up menu of the project icon - the project itself does not even have to be open.

Project Access Methods (PAMs)

Another key requirement for WorkFrame/2 was the ability to better support source code control systems. This has been achieved by abstracting almost all WorkFrame/2 file-system access through a replaceable "Project Access Method", or PAM. File listings, copying, moving or deleting files, and executing actions, are all achieved via the PAM. By providing a new PAM, WorkFrame/2 can support projects located in source code control systems and still appear to you exactly the same as all other projects -- a WPS folder. PAM selection is made on a project basis, so you may have multiple projects active, all accessing their files via a different method, from different locations.

Tools provided with WorkFrame/2

As with version 1, the WorkFrame/2 simply provides the ability to integrate tools - it does not include those tools. The exceptions to this are the two tools that were included with WorkFrame/2 V1.

A Lib facility is included with WorkFrame/2, with both a command line a PM interface. Both versions have been updated for version 2, and the new PM Lib provides a much more usable, CUA compliant interface than version 1.

The makefile creation utility has also been completely revamped, to support the new installable actions. As explained above, every action includes a list of source and target filemasks -- the source masks list all possible source for the action, while the target masks describe every possible file that action may create. The makefile creation utility uses this information to determine the correct order that actions should be executed in. The action options API has also been extended, to allow the vendor provided options DLL to communicate more information with WorkFrame/2, such as exactly which source files were used, and what target files will be created. Of course, all version 1 style option DLLs will continue to be fully supported.

These changes mean that any action with a valid options DLL can be added to WorkFrame/2, and can automatically be included in the makefile. Support is now possible for any number of pre-processing steps, and almost any other action you may wish to add. Another new feature is that all selections made in the utility are preserved, so you do not need to reselect all the files for subsequent invocations.

Conclusion

Now that we have you excited at the prospect of the new WorkFrame/2, let us remind you that it's not available yet. However, those people who purchase C Set + + ***and*** return their registration cards to us (so we know who you are!) will be notified when it is announced.

LINK up with us

We all know about e-mail, and most of you reading this newsletter have IDs on at least one network - many of you, like us, have ids on several. It's fast, easy, and you can talk to people around the globe about pretty much anything. You may even have received this newsletter softcopy.

But what does it mean to you when you think of IBM and talking electronically? Do you realize you could be at the other end of a conversation with IBM? Are you aware you could be initiating discussions with IBMers such as the people behind this newsletter? Some of you already know what we're talking about, and the names Hank, Dave, Stephen, Max, Hiroshi, IBMers all, are very familiar to you.

TalkLink is IBMLink's newest family member, and with this service you have access to countless IBM people, not to mention thousands of your industry peers throughout the United States, Canada, Germany, and Sweden (that's right now .... expansion is planned to

Latin America, the rest of Europe, and Asia/Pacific). It doesn't matter where you are, or when - TalkLink is at your service.

So what's in it for you?

"During 1992, TalkLink was the fastest growing application on IBMLink," says Dennis Johnson, IBM's owner of the TalkLink service. "It's mushroomed from a few key initial forums to over 76, covering a wide range of subjects."

You'll find a host of relevant topics dealing with purchasing, installing, supporting, and using products and services marketed by IBM and related business affiliates. Current hot topics include OS/2 and LAN with short-term plans for DOS, AIX™ and many networking system products to be added. And of course, there's C Set/2 and C Set + + , hotly debated in the C-SET2 forum.

Don't think that TalkLink is just about products, either. Dennis states that the plan is to broaden TalkLink far beyond just product subjects. Future topics may include industry marketing, planning, and services, as well as other forums on all IBM software platforms. "And our expansion plans don't stop there," Johnson continues. "We would like to expand TalkLink to include many other non-IBM business partners and affiliates who will own their own forums. We plan to make TalkLink accessible from several other paths in addition to the Advantis network. We believe this vastly broadens the potential for valuable new products and services."

Now that we have your attention, let's take a closer look at what you can do on TalkLink.

- Conferences and forums

  these question and answer bulletin boards contain various product-related topics. Items posted here are shared between BBS users and IBM personnel worldwide. You can pose your technical questions and ideas on these forums to an audience of product and application developers, both within IBM and within your peer community.

- User-to-user messaging

  send private messages to any other TalkLink user.

- Support

  submit or view problem reports that receive the prompt attention of that conference or forum owner. You can also place orders for product-related materials.

- Software library

  download our application programs and upload your contributions.

- News and announcements

  view recent information regarding products and services.

And did we mention **our** support? C-SET2 Forum on TalkLink is an official support channel for our products. As well as being on CompuServe [3] the C Set Team is active on TalkLink, talking to customers, answering questions, and generally getting to know you. This newsletter is now available too, in LIST3820 and PostScript formats, for all issues we have written to date.

And how do you do all of this?

You can operate TalkLink in two modes. With the interactive mode, you are connected on-line to TalkLink while using its facilities. However, you also have the option to quickly connect to TalkLink, download pertinent forums to your personal computer, and read at your convenience. Your off-line responses are saved and sent to the host the next time you connect in this batch mode.

So there you have it: search IBM databases of questions and answers, System Center Flashes, and problem management libraries, all in an environment you can customize for your personal display choices.

To find out how you can benefit from TalkLink's umbrella of support and services, send a note on IBMLink to AM00190 on HONE92. For a TalkLink information package, call 1-800-547-1283 in the USA. Outside of the USA, please contact your local IBM office.

He's doing it again!

The only negative thing said about last January's ColoradOS/2 conference was that it was a one-off, and many of the attendees spent time trying to persuade Wayne Kovsky, the organizer, to run another. If you were one of the many who thought this was one of the best conferences you'd ever attended, then this article is definitely for you: Wayne is running another.

---

[3] ™ Trademark or registered trademark of CompuServe Inc

The second international ColoradOS/2 Developer's Conference will be held at the Cheyenne Mountain Conference Resort in Colorado Springs, Colorado, the week of October 31st through November 5th, 1993. Marketing folks, system or LAN administrators, beginner programmers, programming managers (unless very technical), and Windows programmers, etc, are likely to be out of place at this get-together because, like the January one, the November conference will have a strong technical orientation, being aimed squarely at the very experienced or professional OS/2 developer. The reviews of the last conference showed that it was intense but fun - like being on a semester-long graduate course with all your best friends but in a mountain resort! And you don't have to dress up.....it's not a business meeting or a fashion show; it's an informal conference focussing on developing for OS/2. Business dress (for either sex) is definitely not encouraged - in January, even the keynote speakers wore blue jeans and sweatshirts.

Topics and speakers for November have not yet been made public, but will be similar in scope to January, where sessions included : IBM C Set + + ; C Set/2 optimizations; object programming and distributed object programming in OS/2; SOM; REXX, object-oriented REXX, and REXX interfaces to programs; programming the WPS; SmallTalk V/PM [4]; using IPMD and EXTRA; programming Containers, Notebooks and Sliders and subclassing controls; writing device drivers; writing multi-threaded programs and migrating to 32-bit OS/2; client-server programming in an AD/Cycle™ environment; OS/2 printing; and many, many, more.

Of the 28 speakers at the January conference, seven had OS/2 books already in the stores, and three more have published (or will do very shortly) since then. Many of the January speakers plan to return in November, and there will be new speakers too.

Want to know more?

More information will be available in early July, when brochures will be mailed out by Wayne. This newsletter will also carry details. If you haven't received a brochure by mid-July, you can call 719-481-3389 for one, or fax your request to 719-481-8069. Of course, if you know for sure you're attending, you can register direct with the Cheyenne Mountain Resort by calling 719-576-4600 (in the USA there's a toll free number 1-800-648-5717).

---

[4] SmallTalk is a trademark of the Digitalk Corporation

Making life easier

It is no coincidence that Usability begins with U and ends with Y ! The main focus of the Toronto User Centred Design Department (UCDD) in supporting C Set/2 (and now C Set + +) is to focus attention upon users who actually work with the products. In doing so we can determine how our product is actually being used to solve real programming problems. More importantly we can also understand why it does not work the way that users may want it to, and we can determine how to improve the product.

Toronto UCDD takes seriously the principle of "Excellence through Quality": we consider user satisfaction to be the key measure of quality products. We define user satisfaction in several ways, including:

Functionally Effective.

Software must provide users with the features they need. It must cater for both expert and novice users in supporting the full range of tasks necessary for users to meet their job requirements, i.e. programming.

Easy to Learn.

Users should be able to learn the software with minimum effort. They should be able to apply previous experience with related software, and relate product concepts to their domain of expertise. Manuals should be well written, and well organized, and on-line help tutorials should be available.

Walk Up and Use.

Users should be immediately productive with the software. Although they may not realize the software's full potential, it should be usable without reference to manuals, and the necessity for lengthy training.

Operationally Efficient.

Users should be able to accomplish their tasks with sufficient speed, and a minimum number of tasks. Errors should be easy to recover from, and unnecessary time kept to a minimum. In essence, the software should work in the same way as the user.

Supported Well.

When things sometimes go wrong, users should be able to determine problems as quickly as possible. On-line contextual help, clear documentation and action-oriented messages should assist

users in their error recovery. Finally, direct support services should be provided (Bulletin Board, telephone etc).

Configurable.

Software should recognize that users are inevitably unique. Differences range in both expertise with computers and with the software domain, i.e. programming. Software should provide appropriate levels of configuration which help users personalize their own systems.

Transparent.

Last but not least, software should be transparent to the user. It should be so well designed that it does not get in the way of users accomplishing their tasks. Users should not have to repeatedly stop and work out how things must be done in order to do their job.

So how do we ensure that these goals are met ? Toronto UCDD has extensive human and physical resources available to support IBM software products. Staffed by professional Human Factors Engineers, Toronto UCDD applies many proven usability processes at various stages of the software development process. Our physical resources includes four usability testing suites equipped with 'state-of-the-art' video / audio recording equipment, and screen capture hardware. It also includes extensive video / audio editing facilities, and specialized logging and analysis software to help us understand the users responses during software evaluations.

Utilizing these facilities, both Internal users, and External customers regularly participate in software evaluations. During these evaluations, participants are expected to solve real problems while we record the various events that take place. The test suite environment can also be adapted to various scenarios, in order to increase the participants comfort, and reproduce various test situations that users may experience in their real work-place. The results from these evaluations are then analysed from different perspectives such as 'time on task', 'number of errors', 'number of documentation references', 'user problems', and general user satisfaction.

Other UCDD activities to help us improve software usability include questionnaires, customer visits, audience analysis, prototyping, user task analysis, customer requirement validation, professional heuristic reviews based on detailed usability objectives and generic guidelines such as Common User Architecture (CUA), and interface standards compliance. Recently we have also added competitive evaluations, and more focus upon inter-disciplinary user centred design teams

with responsibility for determining future product / release objectives, requirements and directions.

Once again, the focus of all these activities is you, the user. The earlier we can collect and analyse validated user information the better....for you, and for the product.

One thing to remember, though, is that in today's changing environment, Usability is a moving target. We endeavor to satisfy current user requirements, along with technology, but users are themselves changing. Users become more sophisticated, expect more from technology, more features in their software, and greater productivity gains from their programming tools. As user interface technology is constantly evolving, software must also change to incorporate more graphical interaction techniques such as the OS/2 Work place shell. In reality, usability is also a fine balance of project development resources. Function may sometimes take precedence over complexity, consistency between tools may be made difficult through technical problems, and user input may sometimes be too late to influence the current release. Needless to say no problem is left untackled, and as these constraints are resolved, new and improved solutions will be implemented in future releases.

To this extent as you read this article you too can be a part of this usability quality process. Your personal experiences with C Set + + will help us guarantee that future versions continue to put the user first. Toronto UCDD appreciates hearing both criticisms and compliments from users. Whether you are having problems, need a feature, are finding something too difficult to use, want to praise a specific feature, have a comparison to make with a competitor, or simply want to 'let off steam', we'd like to hear from you. Although we can't guarantee that all problems will be addressed, we do promise all communications are be carefully considered. Please write, email, or FAX your comments to the address below:

Dr. Roger Spall,
Human Factors Engineer,
Toronto User Centred Design Laboratory,
41/672,
895 Don Mills Road, Tower 1,
North York,
Ontario, M3C 1W3,
Canada.

Electronic Mail: RogerS at vnet.IBM.Com
FAX (416) 448-2114.

Read all about it

Two more books for your bookshelf, both very new. These reviews are based upon input from the authors.

OS/2 2.1 Unleashed

Moskowitz and Kerr (with contributing authors). SAMS Publishing, 1993.

So you have the code. You have the documentation. You're in business, right? Certainly. But there's a new book out that will help you get where you're going with OS/2. Whether you already have OS/2, or you're thinking about coming on board, this book is one you shouldn't miss.

Billed as the only reference book you need about OS/2 2.1, **OS/2 Unleashed** was written by the people who designed or tested OS/2 both inside and out of IBM. Included are many tips and tricks about 2.0 and 2.1 which are not documented anywhere else. And the help isn't limited to OS/2 applications: if you use Microsoft Windows [5] or DOS applications, ***Unleashed*** will show you how to accomplish more, quicker, with Windows and DOS running in the OS/2 environment.

With ***Unleashed*** by your computer, you'll be able to optimize your system in ways you may not have thought possible. Also included is extensive information about troubleshooting to help you find and fix problems faster and easier, not to mention avoid them altogether.

1134 pages crammed with tips, techniques, and how-to's are divided into 18 chapters and 3 appendices, including:

- Getting the best performance from DOS and Windows 3.1 applications
- Undocumented tips about the WorkPlace Shell
- REXX programming: the macro language of OS/2
- Printing techniques
- Changing and optimizing video drivers
- Networking OS/2
- Multimedia and OS/2

[5] ™ Microsoft Windows is a trademark of Microsoft Corp

Included with the book is a disk containing almost 3 Mb of the best OS/2 shareware utilities, such as TE/2, Galleria, LH2, DeskMan/2.

Price? Direct from Productivity Solutions, in the USA $34.95 plus S&H (3 or more copies earn a discount price of $29.95). In Canada $43.95 plus S&H. To order, contact Productivity Solutions at 215-631-5685 (fax 215-631-0414). Otherwise order from the publisher. Prices outside of North America not available at press time.

Designing OS/2 Applications

Reich, Wiley and Sons, 1993, ISBN 0 471-58889 X. $34.95 US. Also available from IBM as order number SC28-2701.

This book is written for application designers and programmers interested in writing applications (in text mode or PM) for 32-bit OS/2. Thus it focusses on the concepts and functions common to the 32-bit OS/2 platform, omitting references to specific versions of OS/2.

An understanding of the basic OS/2 concepts of multitasking, multithreading, virtual memory and device independence is helpful but not required. Designed to be language-neutral, the book discusses the pros and cons of various programming languages and tools. However, a C language orientation is assumed.

Designing OS/2 Applications guides readers through the complete design of an application, from understanding why they would want to write applications for OS/2, to setting the objectives for the application through the design, coding and testing, and finally to performance tuning and designing the installation program and international language support. Throughout the book, emphasis is on efficient program design and structure.

Readers will gain an understanding of the functions and features available as well as which ones are most appropriate for providing a specific feature to users, along with the programming and compatibility considerations of each.

The book is divided into seven sections, covering everything from the reasons for writing OS/2 applications, through OS/2 architecture, PM, and Workplace Shell, to testing, performance and packaging.

We go hi-tech!

In response to the many requests we have received, this newsletter is now available softcopy. We're now available on the OS2BBS of IBMLink, and in Section 5 of CompuServe. In addition, it has been posted, with permission, on several other nets. Please contact us if you have any questions about posting electronically.

ICLCC Debugging - some handy hints

This article is intended to help you debug the most typical problems that might occur when using the collection classes. The following table enumerates the problems, gives a short summary of each, and points to the page of the newsletter where you'll find hints for the problem solution.

Problem Area	Problem Effect	Pg
Cursor Usage	unexpected results when using cursors	38
Element Functions and Key Functions	error messages indicating a problem in *istdops.h*	38
Key Access Function - How to Return the Key (1)	error messages indicating a problem in *istdops.h*: a local variable or compiler temporary is being used in a return expression	39
Key Access Function - How to Return the Key (2)	unexpected results when adding an element to a unique key collection	40
Exception Tracing	unexpected exception tracing output on standard error	41
Declaration of Template Arguments and Element Functions (1)	compiler messages (at pre-link) indicating that an element type or one of its required element functions is not declared	41
Declaration of Template Arguments and Element Functions (2)	compile errors from symbols being multiply defined	41
Declaration of Template Arguments and Element Functions (3)	link errors from symbols being multiply defined	42
Default Constructor	compiler error messages indicating a problem with constructors	42
Considerations when Linking with Templates	unresolved external references during linking	43

Cursor Usage

Effect:

You get unexpected results when using cursors. For example, the *elementAt* function fails for the given cursor or returns an unexpected element.

Reason:

You used an undefined cursor. Cursors become undefined when an element is added or removed from the collection.

Solution:

Cursors that became undefined must be rebuilt with an appropriate operation (for example, *locateElement*) before they are used again. This is especially important for removing all elements with a given property from a collection. This cannot be achieved by coding a cursor iteration. Instead you should use the *removeAll* function which takes a predicate function as its argument for this purpose.

For more information about cursors please refer to the *Collection Class Library User's Guide*, Chapter "Instantiating and Using the Collection Classes" in sections "Cursors" and "Removing Elements".

Element Functions and Key Functions

Effect:

You get compiler errors indicating a problem in *istdops.h* like the following examples:

OS/2 message if key is missing:

```
j:\...\ibmclass\istdops.h(166:1) : error EDC3013                :
   "key" is undefined.
j:\...\ibmclass\istdops.h(160:1) : informational EDC3207:
   The previous message applies to the definition of template
   "IStdKeyOps< Parcel,ToyString> ::key(const Parcel&) const".
```

OS/2 message if hash is missing:

```
j:\...\ibmclass\istdops.h(152:1) : error EDC3070:
   Call does not match any argument list for "::hash".
j:\...\ibmclass\istdops.h(146:1) : informational EDC3207:
   The previous message applies to the definition of template
   "IStdHshOps< ToyString> ::hash(const ToyString&,unsigned long) const".
```

OS/2 message if = = is missing:

```
j:\...\ibmclass\istdops.h(81:1) : error EDC3054:
   The "= = " operator is not allowed between "const ToyString" and
   "const ToyString".
j:\...\ibmclass\istdops.h(80:1) : informational EDC3207:
   The previous message applies to the definition of template
   "equal(const ToyString&,const ToyString&)".
```

OS/2 message if < is missing:

```
j:\...\ibmclass\istdops.h(105:1) : error EDC3054:
  The "< " operator is not allowed between "const ToyString"
  and "const ToString".
j:\...\ibmclass\istdops.h(103:1) : informational EDC3206:
  The previous 2 messages apply to the definition of template
  "compare(const ToyString&,const ToyString&)".
```

Reason:

Compiler error messages indicating a problem in *istdops.h* are always related to the element and key functions that you must define for your elements depending on the collection and implementation variant you are using. The compile errors listed above occur when the key function and the hash function or the operators `==` and `<` are required for your elements and are not defined or defined with the wrong interface. Note that the "constness" of arguments (keyword *const*) is significant and that the generated compile messages cannot always point directly to the incorrect function. For example, a compare function with non-const arguments results in the compile error *The "< " operator is not allowed between "const ..".*

Solution:

Verify which element and key functions are required for the collection's implementation variant you are using. You find this information for each collection in the reference manual in section "Template Arguments and Required Functions".

For more information about element and key functions refer to the *Collection Class Library User's Guide*, Chapter "Element Functions and Key Functions".

Note that the same effect may also be produced if function declarations and definitions are not properly separated between *.h* files and *.C* files, as described in detail in “Declaration of Template Arguments and Element Functions (1)” on page 41.

Key Access Function - How to Return the Key (1)

Effect:

You get a compiler warning like for example:

OS/2 message if key is passed by value:

```
j:\...\ibmclass\istdops.h(166:1) : warning EDC3285:
  The address of a local variable or compiler temporary is being
  used n a return expression.
j:\...\ibmclass\istdops.h(160:1) : informational EDC3207:
  The previous message applies to the definition of template
  "IStdKeyOps< Word,int> ::key(const Word&) const".
```

Reason:

Compiler error messages indicating a problem in *istdops.h* are always related to the element and key functions that you must define for your elements depending on the collection and implementation variant you are using. In this case your global name space function key returns the key by value instead of by reference. This results in the creation of a temporary variable for the key within the operator class function key which returns the key by reference. Returning a reference to a temporary variable naturally causes unpredictable results.

Solution:

Please verify that the global name space function key correctly returns a "key const&" instead of "key".

For more information about element and key functions please refer to the *Collection Class Library User's Guide*, Chapter "Element Functions and Key Functions".

Key Access Function - How to Return the Key (2)

Effect:

You are adding an element into a unique key collection like *key set* or *map* and you are sure that the collection does not yet contain an element with the same key. Nevertheless, you get unexpected results like the *IKeyAlreadyExistsException* or, instead of adding the element, the cursor is just positioned to a different element.

Reason:

This problem is caused by the same reason as the one described in “Key Access Function - How to Return the Key (1)” on page 39. However, you did not get the warning message described above, because you compiled with a lower warning level.

Solution:

The solution for this problem is the same as described in “Key Access Function - How to Return the Key (1)” on page 39.

Exception Tracing

Effect:

You get unexpected exception tracing output on standard error even though the related exception causing the output is caught.

Reason:

For each exception raised the trace function *write* of class *IException::TraceFn* is called and writes information about the raised exception to standard error. This trace function *write* is called whether or not the related exception is caught.

Solution:

To suppress the trace output please provide your own *IException::TraceFn write* tracing function by subclassing *IException::TraceFn* and register the subclass with *setTraceFunction*.

For more information about exception tracing please refer to the User Interface Library Guide.

Declaration of Template Arguments and Element Functions (1)

Effect:

You get compiler messages (at pre-link) indicating that an element type or one of its required element functions is not declared.

Reason:

The element type or element function is defined locally to the *.C* file that contains the template instantiation with the element type as argument. The pre-link phase is executed only by using the header files. Therefore your declaration local to a *.C* file are not recognized which causes these compile errors.

Solution:

Move the corresponding declarations to a (separate) header file and include the header file from the *.C* file.

Declaration of Template Arguments and Element Functions (2)

Effect:

You get compile errors from symbols being multiply defined.

Reason:

The template instantiation automatism needs to include the declaration of the types it received as arguments. Therefore your header files containing declarations of types used in template classes might automatically be inluced several times.

Solution:

Protect your header files against multiple inclusion by using include guards (`#ifndef _MYHEADER_H_ ...`).

Declaration of Template Arguments and Element Functions (3)

Effect:

You get link errors from symbols being multiply defined.

Reason:

The template instantiation automatism needs to include the declaration of the types it received as arguments. Therefore your header files containing declarations of types used in template classes might automatically be included several times.

Solution:

Did you define functions in the header files that are declaring types used in templates? If you did, you must move them from the header file into a separate *.C* file or make them 'inline'

Default Constructor

Effect:

You get a compiler error like for example:

OS/2 message for missing default constructor:

```
itbseq.h(25:1) : error EDC3222:
"IGTabularSequence< ToyString,IStdOps< ToyString>  > ::Node" needs a
constructor because class member "ivElement" needs a constructor
initializer.
Names namesOfExtinct(animals.numberOfDifferentKeys());
ANIMALS.C(55:57) : informational EDC3207:
The previous message applies to the definition of template
 "ITabularSequence< ToyString> ".
```

Reason:

Compiler error messages indicating a problem with constructors for a collection are typically related to the constructors defined for your element. In this case the default constructor for the element is missing.

Solution:

Define the default constructor for the element class.

For more information about element and key functions refer to the *Collection Class Library User's Guide*, Chapter "Element Functions and Key Functions". The element and key functions required for each collection are listed in the reference manual in sections "Template Arguments and Required Functions".

Considerations when Linking with Templates

Effect:

You get unresolved external references during linking that refer to symbols you can not explain.

Reason:

A possible reason for unresolved external references during linking is that template code can not be correctly resolved.

Solution:

1. Did you use *ICC* for linking? *ICC* knows it has to call the template prelinker magic, *LINK386* does not.
2. Did you also use option *-Tdp* for linking? This tells ICC it is processing C++ code that might have templates, so it might have to call the template prelinker magic.

Meet Another C Family Member

As you probably know, C Set/2 is part of the IBM C language family of products. In this issue, we'd like to introduce another family member, C/370. The IBM C/370 Compiler and Library, like C Set/2, conforms to the ANSI C and IBM SAA™ C Level 2 Standards which ensures portability of applications within the IBM C family of compilers. C/370 is designed to run in Multiple Virtual Storage (MVS), Virtual Machine (VM) and VSE (Virtual Storage) environments. C/370 is the oldest member of the IBM C Family as it was first available in 1988. Since then, new functions have been continually added to improve the capabilities, interfaces and performance.

Today, some of the C/370 compiler features on MVS, VM and VSE are:

- An INLINE option to replace function calls with actual function code at the point of call
- Dynamic linking of required routines at program execution
- Database support for SQL/DS™ and DB2™
- VSAM and extended I/O support
- Full program reentrancy allows multiple users to share a single copy of an application
- Checkout compile-time option generates extended informational messages to help the programmer diagnose C/370 problems.

Features unique to the MVS and VM environments are:

- A state-of-the-art Global Optimizer that provides significant performance benefits
- Systems Programming Capability as an Assembler alternative for writing system exit routines
- Inter-language calls to OS PL/1, VS FORTRAN, VS COBOL II, Assember H and CSP
- Strong interfaces to other IBM products such as QMF™, ESSL, GDDM, CICS/ESA™, ISPF, TPF, DL/1, CSL, DWS
- Exploitation of the 31-bit addressing capabilities of the XA environment
- Multi-tasking Facility to exploit an MVS multiprocessor configuration.

Currently, there are 2 Versions of the C/370 Compiler. The IBM SAA C/370 Compiler (5688-187) and Library (5688-188) provide all the functions above. C/370 supports INSPECT, a full-screen interactive source level debugger for C/370 and PL/1 on MVS and VM Operating Systems.

IBM SAA AD/Cycle™ C/370 Compiler (5688-216) provides the capability and power of the C/370 Compiler coupled with the productivity and usability benefits of the AD/Cycle environment. AD/Cycle C/370 runs on the MVS/ESA™ and VM/ESA™ operating systems. The run-time features and functions are provided by IBM SAA AD/Cycle Language Environment/370 (5688-198). LE/370 also provides the library environment for AD/Cycle COBOL/370 (5688-197). Programmer productivity is enhanced with the cohesive edit-compile-debug facilities offered by CoOperative Development Environment/370 (5688-194). CODE/370 is a cooperative application development tool that provides a language-sensitive editor, advanced debugging capability, cooperative processing and comprehensive online information.

Like C Set + + on OS/2, C/370 has its own newsletter. You can subscribe to it by writing or faxing your request to the address below:

Programming Systems Toronto Laboratory
IBM Canada Ltd
Workstation Languages Planning
Dept 884
22/884/844/TOR
844 Don Mills Road
North York
Ontario, M3C 1V7
Canada

Attention of Gordon Sinclair.

For more information on the products above, please contact your IBM Representative or Business Partner or via the reader's comment form at the back of their newsletter.

You ask, we answer

...will not be seen at this time. Tune in next issue.

Normal programming will be resumed

This has been a special edition of our newsletter, issued to support the announcement. As a result, some of our regular features are absent. Those readers who wrote to us wondering why part two of our 16-32 bit calls series wasn't in issue number 4 (April 93), are probably even more concerned that it's not in this issue either. You will be relieved to know that we'll be back to normal in the next issue and that article will be there.

And don't forget.....return your survey from the last issue!

Did you know this?

Have you ever wondered how to relate the EIP value in a trap back to your code? For an EXE, it's usually quite simple:

When you compile your code, specify the /Ti option, even if you are optimizing the code (the /O+ option). When you link, specify the /M and /L linker flags. This will generate a map file, and put the segment and offset of all your source lines into it. You don't have to link with the /DE flag for this to work.

Since most .EXE files place the 32 bit code into segment 1, which is loaded at 0x10000, you can convert the EIP to an address in segment 1 by subtracting 0x10000 from it. Then look up the line with the next lowest address in your MAP file, and you have the line at which you trapped.

A word from your editor

If you didn't receive this newsletter mailed direct from IBM, and you would like regular hardcopy, then let us know. Mail in the reply form at the back with your request to join us, plus your full mailing address, and we'll add you to our mailing list, wherever you are.

Yes, wherever you are...from Argentina to Zaire, if you have a mail service, we'll mail you our newsletter.

For the many of you sending in the reply forms with your comments, we've often had the need to call you to discuss your comments further. It's a great help if you include your phone number. Thanks!

This newsletter is available softcopy on several networks. If you obtained your copy electronically, you may not be on the C Set + + mailing list. If you're not, you're missing out! You're not receiving the product information, information updates, and other general mailouts that others are. You can be on our list for these goodies, with or without hardcopy newsletter distribution. Just mail/fax your full address (phone number is handy too) using the Reader's Reply Form at the back of this copy, and circle whether you want the NL hardcopy, or just want to be added to our mailing list.

This newsletter was produced by the OS/2 C/C+ + Planning department of the PRGS Toronto Lab. For further information on any of the products mentioned, please contact your local IBM office, or an authorized IBM Business Partner.

Numerous product references in this publication are registered trademarks or trademarks of International Business Machines Corporation. IBM Canada Ltd., a related company, is a registered user.

Sample code is provided for information purposes only, and is used by readers at their risk. IBM makes an effort to provide accurate and safe code examples but does not warrant their correctness.

This newsletter was created and marked for processing using IBM BookMaster (Program Number 5688-015) and IBM Document Composition Facility (DCF)™ (Program Number 5748-XX9).

The final copy was printed on an IBM 3825 Page Printer, an Advanced Function Printer.

This newsletter is ê IBM Corporation 1993.

IBM

IBM C Set ++ News No.5 May 93 issue

Reader's Reply Form

1. *Did you find this newsletter useful?*
2. *Is there any way you think we could improve this newsletter?*
3. *Is there any compiler-related subject you would like to see addressed in this newsletter?*
4. *If you received your copy electronically, would you like this NL hardcopy? Or just be on the mailing list? (circle one).*

Please note:

- *IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you, and all such information will be considered non-confidential.*
- *Do not use this form to report compiler problems or to request copies of publications. Instead, contact your IBM representative or an authorised IBM Business Partner.*
- *If you wish, you may include your name, address, and company name if applicable, and your phone number.*

_____

_____

_____

_____

_____

Thank you for your cooperation and help. You can either mail this form to us, or hand it into an IBM office for forwarding.

You can also fax the form to us. Our fax is 416-448-6057. Please mark your fax for the attention of MJ Houghton.

IBM C Set ++ News No.5 May 93 issue

Reader's Reply Form

Fold here and tape.....fold here and tape.....fold here and tape.....fold here and tape.....fold here and tape..... fold

IBM

PRGS Toronto Lab
IBM Canada Ltd
Workstation Languages Planning D394
22/394/844/TOR
844 Don Mills Road
North York
Ontario, M3C 1V7
Canada

Fold here and tape.....fold here and tape.....fold here and tape.....fold here and tape.....fold here and tape.....fold