The Front End Module

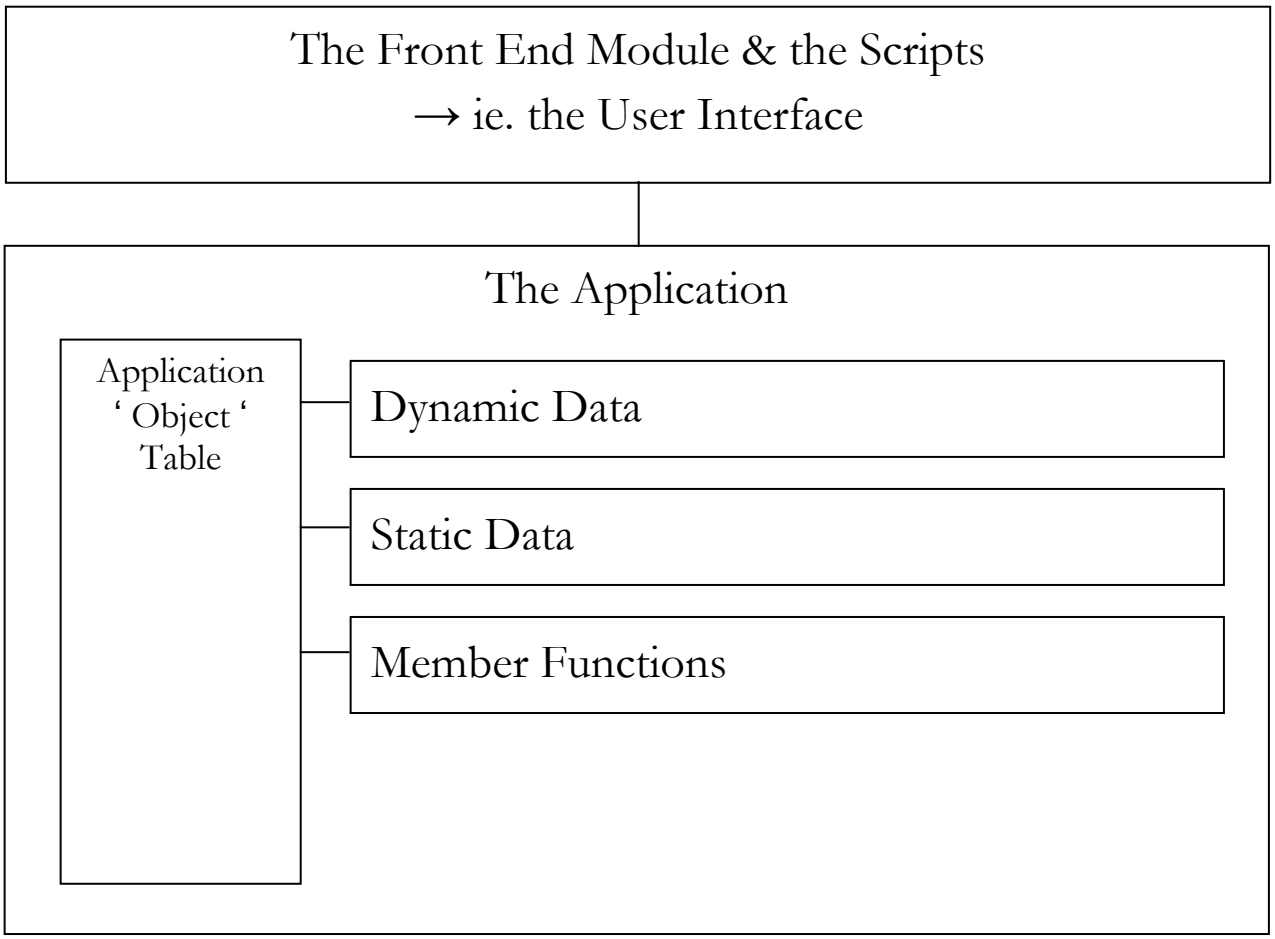
a highly sophisticated and extremely cost effective method of designing and managing User Interfaces

Description

The Front End Module is a middleware product that provides the means to design and manage full featured User Interfaces linking up the external user environment directly with the Application program . It sits between the Application and the GDI functions . The Module acts as a User Interface layout and input manager for the management of the display and the input of data . Full data editing facilities are provided . The screen layouts are specified using highly compact binary scripts . The Module provides for the complete separation of the User Interface design from the Application functionality . This allows for common and fixed Application components to be used thus maximising reusability of the Application code . The User Interface design being fully changeable and upgradeable . Further because the User Interface design is scripted it is fully portable . It does not need recompiling whenever the application code is changed or when it is installed on different hardware .

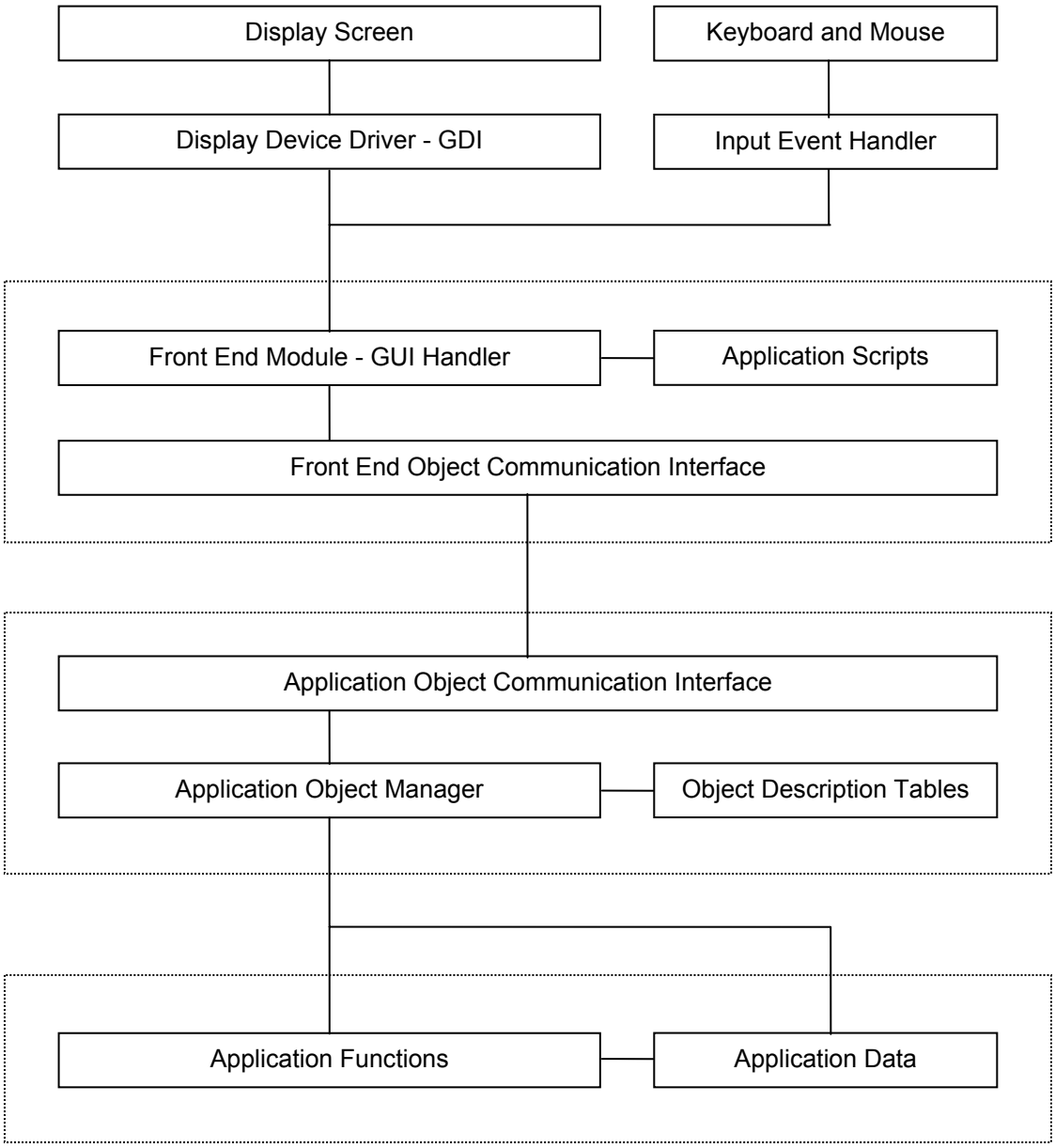
The Module packs in a lot of intelligence into the User Interface operation . It is an extremely useful piece of software that can take your User Interface design and operation up to a very high level of operation .

Operational Architecture



The Module is designed on an ' Object Oriented ' principle where objects in the Application with data and function element pairs are linked up to corresponding data on the screen . This provides for a simple and seamless connection from the user to the Application data and operation .

Module Architecture



The Module has an extensive set of commands providing all the facilities required for User Interface construction . These provide for full screen layout control , data input and text and data display . Further it has extensive operation management facilities thus ensuring that all operations can be handled and that the application code size is minimised . It can be easily and quickly set up and configured for almost any application .

Using the script commands is like constructing a C program except the commands are directly concerned with user interface design . As they are interpreted commands the Module handles the commands and their parameters in a sophisticated and dynamic manner . This means that user interfaces can be designed to be dynamic and responsive through simple command specifications . The commands are written using macros . These allow the scripts to be simply constructed and to be easily compiled into binary code . As the scripts are binary coded they are very quick to execute .

The Application Scripts are within the domain of the Front End Module and the Application Objects are within the domain of the Application . As such they are separated . Further the Front End Module and the Application can be fully separated and be connected by a communication link . As such the Application acts as a purely functional module and the Front End Module acts as a purely display and input management module . This allows the User Interface to be customised to specific market and customer requirements without alterations being required to the Application .

Because the Front End Module can be separated from the Application the User Interface can , for example , be located on a separate computer - eg. a lap top or a palm top (PDA) - from the device - eg. an embedded system - that is being monitored and controlled thus greatly reducing the size of code on the device . The Module is also ideal for Single Chip Solutions .

Further the ability to separate the User Interface from the Application means that the Front End Module can also be placed within a separate user interface chip within a multiprocessing environment .

The ‘ Objects ‘ in the Application consist of data and function element pairs . If the data is specified it is updated in response to it’s corresponding element data in the User Interface being updated . If the associated function is specified the function is called . This provides a mechanism for checking data values and acting on data changes . Further if the data is not specified but the function is specified only the function will be called . This allows , for example , buttons on the User Interface screen to be linked up to functions in the Application .

All actions , such as button presses , result in script executions . Data manipulation and Application function call commands can be placed within the scripts . Thus internal User Interface operations can be set up to result in external Application function executions . Everything is fully automatic . All situations have been catered for . Interfacing to the Module and using the Module is extremely easy .

When the Application wants to update data in the User Interface it informs the Front End Module that the data is to be transferred to the User Interface . On reception of the Object Element Event Message the Front End Module will update the data and will execute any script objects and associated scripts that use that data . As such the display will dynamically react to the new data . A full set of conditional script execution commands are provided to display scripts according to specified data conditions . As such full control can be provided to the Application as to which scripts are executed .

The Module is asynchronous in nature and is supplied with a basic Co-operative Operating System . As such there are no lock up conditions within the module .

The Module is written in ANSI C and is supplied with a MSVC++ harness for evaluation and Application User Interface prototyping purposes . A full set of support files are provided along with a basic Display Device Driver (GDI) , an Input Event Handler and a Communication Device Driver . A separate MSVS project is provided to develop the script files . Coding standards have been used .

Because the Module has already been developed the unknowns have been taken out of your project development . You can quickly , easily and cheaply get your User Interface up and running .

The Module can be easily and quickly connected to your platform's GDI functions and operating system .

The Front End Module is extremely easy to learn to use . It’s just a matter of trying out the script commands and getting to know their capabilities . Simple screen designs can be developed initially . The screen designs can then be evolved to more sophisticated designs . Your can be on stream in a couple of hours . The Module is very easy to port . It only requires a basic GDI and can operate without an Operating System .

Further the use of The Front End Module allows a formalised approach to User Interface design to be used .

Features

1) The description of the screen layout is specified by compact binary coded scripts . Separate bit ranges are used for commands , data prefixes , data and text .
2) The User Interfaces - via the scripts - can be developed very quickly and very easily . Not only can products be released quicker but also more time is available for the User Interface design – eg. for maximising the quality and features of the design .
3) Through the use of different script files for the same Application - different User Interfaces (skins) can be designed for different market areas thus maximising the market reach of your product . As such both large general and small niche markets can be easily and cheaply catered for .
4) Scripts can be accessed as files - this allows the User Interface to be remotely changed and updated . Further because the Application acts only as a functional module the application code can be fixed - as it often would not need any alterations when the features are updated - where these are provided by the User Interface . Scripts can also be compiled in as part of the code .
5) Multiple script files (including library files) can be simultaneously accessed . Script files can be closed .
6) Script ‘ objects ‘ and associated ‘ display objects ‘ are created on script execution . Scripts can be ‘ instantiated ‘ - re-executed as separate entities . Full script , branch and overlay management and controls are provided .
   i) Screens .
   ii) Overlaid Screens .
   iii) Branches (parent and child script objects) .
   iv) Segments - branch sections that can be removed from the branch .
   v) General and specific instantiation of branches .
   vi) Command based removal of scripts (script objects) and branches (parent and child script objects) .
   vii) Automatic displacement (removal) of scripts and branches displayed at the same position .
   viii) Automatic detection and handling of conditions such as looped branches and implied script instantiation .
   ix) Re-display of area associated scripts - when actions such as text field movement and re-sizing occurs .
   x) Automatic display object re-display (on associated data change) and automatic complete screen re-display .
7) Inline (within the script) , System , Local (to the scripts) and Global (Application) Data . Indexed (script and data array) , Percentage , Script Table and Parameter based data . Numerical , string , document and picture data and graphics characters . Separate character ranges for commands , data description prefixes and data and screen text . Bi-directional parsing of scripts . Interchangeable data sources .
8) Data Objects - all data types - including Unsigned and Signed Decimal , Binary , Hex , String , Document (eg. .txt files) and Picture . Initial value setting . Maximum value setting and checking . Inline and external data message setting . Array based data . Fixed and variable size string and text document data . Instantiated data .
9) Direct connection to Application Data and Functions - easily specified via Data ' Object ' Tables . Application data is automatically updated and application functions automatically called . Alteration of data in the Front End - User Interface - causes automatic updating of application data and calling of associated functions .
10) Linkage to the Application is via simple tables - macros are supplied to encode the table entries . The tables specify the ‘ object ‘ names and the data and associated and separate functions .
11) Linkage to the Application uses an ‘ object ‘ based format and provides for Gain Focus (constructor) and Lose Focus (destructor) functions . Objects are named within the Application and within the scripts . Linkage is on the basis of the ‘ object ‘ name and the ‘ object ‘ field number - within ANSI C format .
12) Linkage from the Application to the Front End is via a simple set of macros .
13) Application Data can be static or dynamic . If dynamic the application side Object Manager will automatically create the data on the heap .
14) The Application can remotely change the system data to cause , for example , the display of a specified script .

15) Linkage to Scripts and Data Based Selection of Scripts by :-
    i) Directly specified data within the script - out of a command parameter list - within the text area .
    ii) Link command - displays the specified script .
    iii) Multi Link command - displays the list of scripts for the number of scripts specified in the data . Has overlay controls .
    iv) Call command - allows parameters to be passed from script to script - provides for the design of complex and portable screen objects - such as Menus and Combo Boxes .
    v) Call External command - allows parameters to be passed from a script to a script in an separate script file .
    vi) Script Macros – sections of script code and data can be defined as macros and be referred to as code or data .
    vii) Widgets – external scripts executed via numerical (enumerated) references . Widgets be referred to as code or data .
    viii) Repeat command - repeats the specified script for the number of times specified in the data .
    ix) Page command - repeats the specified script for the number of times specified in the data . Provides page controls - allows buttons (eg. directory entries) to be created and managed as a page .
    x) While command - repeats the specified script while the specified data is FALSE .
    xi) Choice command - displays the script corresponding to the index specified in the data .
    xii) Masked Link command - displays the list of scripts corresponding to the selection bits specified in the data .

    This allows scripts to be linked to (executed at the current script position) and allows the screen display to be altered in response to data changes . Altering the data causes an automatic update of the display .

    The Multi Link , Repeat , Page , While , Choice & Masked Link commands can also handle inline script code sections .
16) Dependence of scripts on data - when the data is altered the script is re-executed (re-displayed) or removed .
17) Internal Scripts - referred to by Script Number - and External Scripts - referred to by Script File Name . Scripts handled as files - can be externally loaded . Script files can also be placed directly in the code space .
18) Association of Scripts with data - when the data is altered the script is removed or is re-executed (re-displayed) .
19) All styles are specified within Style Tables . Individual Frame , Area , Table , Field , Graphics and Font styles can be selected . Styles can also be overridden by using the Set Style command .
20) Text Styles :-
    i) Type Face .
    ii) Style - eg. italic or bold .
    iii) Size .
    iv) Background and Foreground Colours .
21) Variable Font - full screen and data text display and cursor movement handling . Script lines are parsed - text and objects - to detect the overall line height and ascent . Control over the line height and ascent is provided .
22) Frame Styles :-
    i) Flat , Raised or Lowered Edges .
    ii) Edge Width setting .
    iii) Rectangular or Rounded Corners
    iv) Frame Pattern and Colour with optional Background Picture .
    v) Background and Foreground Pattern and Colours .
    vi) Raised or Lowered Lands .
    vii) Left , Central and Right Aligned Labels .
    viii) Default style and sub-style handling .
23) Additional Associated Styles :-
    i) Fonts - type , style , size - screen text , areas and fields .
    ii) Spacing - areas and fields .
24) Framing of Text Displays , Areas , Fields , Tables and Table Cells .
25) Setting of Font and Graphics Styles .
26) Areas - defines start position , size and frame .
27) Tables with Default and Custom Cells - handled as areas .

28) Data Fields - with full data editing facilities :-
    i) Character , Word and Field Cursor Movement - Left and Right , Up and Down and between fields .
    ii) Overflow and wrap around .
    iii) Go To Line Start , Line End , Data Start and Data End .
    iv) Page Up , Page Down , Page Left and Page Right .
    v) Previous Page and Next Page .
    vi) Delete Previous Character & Delete Current Character .
    vii) Insert & Revise .
    viii) Mouse , Cursor and Word based Data Selections . Select All .
    ix) Delete , Replace , Cut , Copy & Paste . Drag and drop - copy (out of the field) and move (in the field) operations .
    x) UnDo and Re-Do .
    xi) Increment and Decrement data .
    xii) Data fields display a ' window ' on the data - data is automatically moved into and out of the display area .
    xiii) Vertical and Horizontal Scroll Bars (sliders with movement buttons) .
    xiv) Grab Bars with Hide \ Show , Minimise \ Maximise and Close Controls . Movement and Re-sizing .
    xv) The cursor can be allowed to move to the outer right of the display area .
    xvi) Password - '*' - characters can be displayed on character input .
    xvii) Descriptive Messages - removed from data on selection of the data field .
    xviii) Display Only and Display and Edit .
    xix) Automatic association of data fields with associated data editor component - according to data type .
29) Button Fields:-
    i) Full range of types and frame styles provided for . Can - via the GDI - display any button type - no frame , standard frame types , check box , radio button etc.
    ii) Can be explicitly displayed or can be associated with text or screen areas .
    iii) Can be displayed Up , Down or Highlighted .
    iv) Can operate on Cursor Selection , Button Down or Down then Up - Mouse Button or Enter key , Right Click or Double Click .
    v) Can be toggled and auto toggled - eg. menu operation .
    vi) Can be cursor selected .
    vii) Field Selection , Mouse and Enter Key button press or Mouse Only button press .
    viii) When pressed causes the execution of the associated script .
    ix) The label can be specified inline or specified by data .
    x) The label can be left , center or right aligned .
    xi) Can be field selected - eg. the selection of files within a directory . The selections (button labels) being loaded into system data which can then be sent through to the Application . The selections can also be pasted or dropped into data .
    xii) Buttons can be clicked through - data labels can be edited - allows , for example , for file names to be changed .
    xiii) Menus with full page controls can be constructed .
    xiv) Link Section fields can be left mouse button activated , double click left left mouse button activated and right mouse button activated.
    xv) Activate and Link Section fields can be overlaid on other fields .
30) Link Section Fields - when the mouse button is pressed over the screen area (defined as a script or rectangular section) the associated script is executed .
31) Activate Fields - executes the associated script when the mouse cursor is held over the area . Can be used to display descriptive (eg. help) text .

32) Button presses , Link Scripts and Sections and Activate Fields execute events :-
    i) Script Execution .
    or
    ii) Data Setting .
    and \ or
    iii) Application Function Execution .
33) Grab Bar Field - can be associated with data and can be used as a grab bar for areas and data fields .
34) Slider Field - with movement buttons - can be associated with data and can be used as a scroll bar for a data field .
35) Drop Field - allows data that has been selected to be dropped indirectly into data areas .
36) Resource Field – Audio , Picture or Movie .
37) Graph Field .
38) Fields can be overlaid on top of each other .
39) Positions : -
    i) Absolute and Relative positioning .
    ii) Absolute and Relative rotation .
    iii) Movement relative to line tops and bases .
    iv) Movement relative to Area sides .
    v) Area associated positioning . Areas can be referred to from outside or from within the area .
    vi) Table associated positioning . Tables can be referred to from outside or from within the table .
    vii) Tabs with positions set by tab tables .
    viii) Indents .
    ix) Horizontal - left , center and right - Alignment .
    x) Vertical - top , middle and bottom - Alignment .
    xi) Inherited and Current Display States .
40) Ordered lists with type specification - ([{ a A 1 i I }]) - and with full nesting .
41) Data Operations :-
    i) General - Select Data (contextual based) , Select Current Field Data & Select Specified Field Data , Externalise Data .
    ii) Arrays - Externalise Data , Clear Array , Set Number of Entries , Get Number of Entries , Set Entry Width , Get Entry Width , Get Entry Height , Sort & Reverse .
    iii) Button Width . Menu Width & Height .
    iv) File Based - Load File Name , Search , Search extended , Load , Save , Store , Rename , Copy , Move and Delete .
    v) Data Base - Load , Save & Store , Insert & Delete Fields and Lines .
    vi) Arithmetic - Load , Load Array , Exchange , Append , Clear , Increment , Add , Decrement , Subtract , Multiply , Divide , Remainder , Percent , Minimum , Maximum , Decrement on Not Zero , Negative .
    vii) Arithmetic Conditional - Not Equal , Equal , Less Than , Less Than or Equal , Greater Than , Greater Than or Equal .
    viii) Logical - And , And Not , Or , Xor , Shift Left , Shift Right , Invert , Not .
    ix) Logical Conditional - No 1's Match , Some 1's Match , All 1's Match .
    x) Exit on Zero , Exit on Non Zero .
    xi) Application Function Execution - Synchronous and Asynchronous .

42) Graphics :-
    i) Lines .
    ii) Rectangles .
    iii) Arcs .
    iv) Fill Graphic Areas .
    v) Blank Graphic Areas .

    The graphics commands , as with all commands , use formatted data - where the data source is specified as a prefix . As such if a variable data source is used and if the data is altered the whole script that contains the graphic command is re-executed - thus causing re-drawing of the graphics in the script .

43) Graphics Styles :-
    i) Line Type .
    ii) Line Width .
    iii) Line Colour .
    iv) Foreground and background colour .

44) Automatic initialisation script execution . Setting of language section characteristics . Setting of default Display State characteristics – loaded on execution of scripts within script command specified ranges .

45) Execute Script (display script - constructor) , Re-display Script and Remove Script (script based and external application based removal of the script from display - destructor) operations . Removal of overlay areas , branches and screens .

46) Overlaid scripts and Areas . Transparent (actions go through) scripts and opaque (actions don't go through) Areas . Restriction of cursor movement to selected Areas . Localisation of actions to overlays . Overlay of screens (ensures no transparency to the underlying screen)

47) Automatic displacement of scripts of the same level previously displayed at the same position . Full management of scripts , script objects and associated objects . Automatic displacement of screens by screens .

48) Script Timer operation provided - causes the execution of scripts after the expiration of a specified time . The timers can be single shot or repeated . Scripts can also be removed from display . Timer management facilities are also provided .

49) Input Events can be generated within scripts and be executed on fields . This allows , for example , for keyboards to be constructed on the screen .

50) Layout controls :-
    i) Vertical - organise into columns .
    ii) Horizontal - organise into rows .
    iii) Offset Y - tile overlaid .
    iv) Offset X - tile overlaid .

51) Because the Module uses basic commands (rather than complex display object graphic commands) a very high level of control is available to specify the layout and appearance of complex screen objects . The commands provided can be easily put together to construct complex objects . As such Menus and Combo Boxes can , for example , have any button type and layout and lines between items . Further because these objects are constructed using component commands they can be widely varied in their design . As such , for example , overlaid buttons and dividing lines can be put on menu's and different button types can be used for combo boxes . As such full control is provided over the presentation of the User Interface .

52) Full Multiple Language handling - either included within the script file or within a separate language file that is loaded when the language selector is altered - is provided . 8 bit extended ASCII character sets catered for .

53) The Application can be remotely located from the User Interface . The User Interface can be located in a separate chip - single chip solution . This allows a distributed processing solution to be provided . The User Interface can be located on a separate computer - this minimises the byte space required on the embedded device .

54) The Front End Module is asynchronous - ie. no lock ups !

55) User Interfaces are asynchronous in nature as such they require :-

i) Ease of movement between menus - branches - not only going up a branch but directly across to another branch .

ii) Inputs from multiple sources :-

a) keyboard .

b) mouse .

c) the application .

d) scripts .

The User Interface must be able to react to these as they occur . As the Front End Module is asynchronous all these are handled .

56) The features can be compiled in or out by means of conditional compile switches thus allowing the build size to be minimised .

57) The Module is easy to port - it only requires minimal OS and GDI functions and is extensively instrumented .

Purchasing

The Front End Module is sold under a usage license , in source code form and comes with full support and an optional porting service and a user interface design - script design - service . We can also provide a full All In User Interface Design Service - please contact for details .

Prices

License Coverage	Price £'s	Price €'s	Price U.S. $'s
1 Platform type within the Company + PC Development Platform - single upfront charge , no additional royalty	£ 2 800	€ 4 000	U.S. $ 5 000
Each Additional Platform within the Company - single upfront per platform charge , no additional royalty	£ 1 285	€ 1 825	U.S. $ 2 325
Site wide license - for use in all projects developed by the company on or from a single site	£ 4 675	€ 6 625	U.S. $ 8 415
Company wide license - national - for use in all projects developed by the company in a single country or up to three sites internationally	£ 5 435	€ 7 725	U.S. $ 9 810
Company wide license - international - for use in all projects developed by the company in any country	£ 6 210	€ 8 825	U.S. $ 11 210
In projects developed by the company under a development subcontract for clients of the company	please contact		
Within Software Components (sold separate from a project) manufactured by the Company			
Within Hardware Components (sold separate from a project) manufactured by the Company			
Porting - On Site	£ 1 865	€ 2 650	----------
User Interface - Script - Design - per week On Site	£ 1 660	€ 2 355	----------
Porting - at base	£ 1 660	€ 2 355	U.S. $ 2 845
User Interface - Script - Design - per week at base	£ 1 445	€ 2 050	U.S. $ 2 485
Support for Each Additional Year - first supported site	£ 160	€ 225	U.S. $ 285
Support for each additional site per year	£ 16	€ 22	U.S. $ 28

Guarantees

The components we supply are guaranteed to work as we have intended . They have been developed to the highest possible standards and have been extensively tested . If you have any problems at all please contact us . We will either assist you as part of the support package or , if , for any reason , any the components are not functioning as they should we will rectify the problem and supply you with the new version .

It is extremely important to us that your project works and , as such , that any modules and components that we supply you with work fully as intended . Further it is also important that we fully cater for your current and future needs . We are prepared to work with you , during your project development , to expand the features and capabilities of the Module to suit your needs - at no additional cost .

World Computers Ltd.

Suite 12397 , 72 New Bond Street
London , W1S 1RR
U.K.

Phone :- +44 - 7785 - 362 319
Fax. :- +44 - 8717 - 334 254
Web Site :- wrldcomp.com