Visix Software, Inc.

11440 Commerce Park Drive
Reston, Virginia 20191

703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

Java Application Development with Vibe

A Visix® White Paper

Vibe™ from Visix Software is the linchpin in the Visix strategy to provide Java™ developers with technology that allows them to develop and deploy large-scale intranet applications. Vibe consists of a visual Integrated Development Environment (IDE) that enables the development of application components in the Java programming language introduced by Sun Microsystems®, and an enriched virtual machine (VM) environment for deployment and execution of Vibe-built application components across multiple heterogeneous platforms.

Contents

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

About Visix Software

One way to escalate the introduction of viable new technology is to leverage significant industry expertise and prior investment in object-oriented tools to the greatest degree possible. With Vibe, Visix Software is leveraging its industry leadership in object-oriented application development concepts since 1989. The company has invested over 250 man years of top notch engineering talent in the development and refinement of the Visix Foundation Classes, the industry's most mature, feature-rich and high performance object classes available for application development.

Visix established a very early start in Java language development tools - choosing the language early in 1995 as the native language for a new development environment now called "Vibe". By building upon the most important features of the Java language and integrating the raw power of the Visix Foundation Classes, Visix has created a revolutionary product which will take a position of prominence in the software development industry.

Vibe delivers solid advanced functionality that can justify its immediate adoption for Intranet application development. Meanwhile, its maturity by virtue of Visix's early start and incorporation of proven object class technology, help to mitigate risks of adoption and ensure longer term viability.

About Vibe

Vibe contains two integrated elements. The visual Java-specific Integrated Development Environment (IDE) is designed specifically for rapid prototyping and development of large scale, distributed, applications. In addition to the visual developer interface, the IDE includes a Java compiler and debugger to support immediate testing as components are constructed.

The second element, the Vibe VM is an implementation of the Java virtual machine. It affords greatly enhanced functionality and performance by virtue of a native implementation of the Visix Foundation Classes. The Vibe VM supports the dynamic execution and distributed processing of large-scale Intranet applications across multiple desktop and server platforms.

Java application components can be developed and executed on Microsoft Windows™ 95, Microsoft Windows NT™, Hewlett Packard HP-UX®, SunOS™, Sun Solaris®, Digital UNIX®, IBM AIX® and OS/2®, Linux, SGI IRIX™ and Mac® OS. The Visix approach to cross-platform portability overcomes the least common denominator or "layered" approach that is a hallmark of most other Java development tools. Visix delivers identical, broad, and powerful application functionality on every platform through its "superset" approach to cross-platform compatibility.

Vibe yields Java-built components with extraordinary performance improvements and greatly expanded functionality over Java applets that can be rendered by any competing implementation.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

Vibe's General Features and Benefits

Using the Vibe IDE, corporate developers familiar with products like Microsoft's Visual Basic® or Borland's Delphi™ are able to easily translate their skills into a familiar development environment providing rich Java functionality without requiring developers to learn the intricacies of more complex programming languages. The Vibe VM is the most powerful, high-performance deployment vehicle available for Java-built application components. Because powerful native functionality is built into the runtime environment, Java developers require fewer lines of Java code to build components. Smaller bytecode components are more efficient in transmission over networks and faster in execution on the target platform. The combined architecture yields the following general features and benefits:

- Fully integrated development and deployment facilities.
- Unprecedented User Interface (UI) functionality. The underlying class libraries provide Java-defined graphical user interfaces which are vast improvements over what can be built today with any other Java development tool.
- Uniform Class Libraries. A superset of all key platform functions affords consistent application functionality across all platforms while providing "native" look and feel.
- Identical application functionality across multiple platforms including Microsoft Windows 95, Microsoft Windows NT, HP-UX, Sun Solaris, AIX and Mac OS. This eliminates the need to choose between disparate, inconsistent sets of functionality from Microsoft or Sun or anyone else.
- The distributed object architecture allows the use of Java for desktop components (where Java's component portability features yield the greatest benefit) and the continuing use of more traditional languages such as C or C++ to develop or retain server side components where Java affords fewer benefits.
- Integration of application components or pre-built "controls" using another development technology. The runtime environment includes Vibe components, standard Java components, standard ActiveX controls, and some native Galaxy-based components thereby maximizing the benefits of code reuse.
- Accessibility to, and execution of, applications from within desktop browsers. The VM is delivered as a plug-in to popular browsers thereby taking advantage of this type of broadly proliferating "universal" client.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

Vibe's High-Level Design Philosophy

Vibe makes use of the Visix Foundation Classes, which were built using high level design concepts that have been proven over many years to yield enhanced productivity, reduced redundancy, improved fault detection, improved flexibility and adaptability, and ease of maintenance. These design principles manifest themselves in the form of the following design "models".

The Abstraction Model

Vibe's object-oriented design has multiple levels of abstraction using the following native platform services as a base: operating system, network, and windows. All abstractions start at the lowest level of each native service, building a foundation for successive levels that support Vibe's comprehensive higher level class libraries and frameworks. This architecture provides the developer with a wide range of "entry points" in the class structure. For simpler applications, the developer employs object classes at a very high level of abstraction pulling together larger chunks of reusable code to quickly build applications. For experienced developers, accessing and incorporating object classes at a low level of abstraction facilitates fine grain control over application construction, allowing high levels of customization needed to support extraordinary requirements.

The Ideal System Model

The Vibe Ideal System Model defines a uniform superset of application services (operating system services, window system services and network system services) across all platforms. All of these services are accessible through a common API. Vibe also supports a superset of cross-platform user interface objects delivering a standard look-and-feel on every platform. The product also provides a consistent set of services to the application, even when the native platform does not support a particular primitive function. For example, if the native platform's available resources are unable to support the color requirements of an application, Vibe can compensate by using closest match colors, dithered colors, or even allocating a color map at execution time. If a platform does not support the concept of a notebook dialog item, for example, Vibe will implement the notebook for the developer on each deployment platform. Vibe, therefore, provides the ability to develop ideal application components based on business related functional requirements rather than sacrificing application characteristics in order to conform to platform constraints.

The Resource Model

Vibe implements a robust resource model for efficiently managing the development and maintenance of application resources. Resources may include dialog windows, user interface objects, text, error messages, color images, geometry management or other structured data. The resource model treats high level items as persistent objects, storing them in a machine-independent binary form. This model provides a powerful abstraction that supports visual, incremental development of the graphical user interface. It also eliminates the need to build and maintain a large code base to define the user interface and other application resources. The resource model enables highly flexible adaptation of application components that can change to meet business requirements without incurring large volumes of code revision.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

The Imaging and Drawing Model

The Vibe Imaging and Drawing Model is PostScript® based and provides a powerful abstraction that assumes a target graphics output device with near infinite resolution, unlimited colors and algorithmic fonts. The product renders the most accurate image depiction possible through a variety of operations including error-diffusion dithering, re-sampling and color mapping. Vibe also provides high-performance, off-screen bit map rendering on every platform.

Vibe implements a state-of-the-art drawing model that allows developers to use the same drawing instructions for any raster device. Vibe's drawing model provides the closest possible match between the screen and printer output without special application coding. The product directly supports monochrome, gray scale, color-mapped, and true color displays of different depths. Vibe also supports a wide variety of hard copy output devices. The model provides scaling, rotation or any sequence of 2-D transformations on any display or output device.

The Vibe Development Environment

The Vibe Integrated Development Environment (IDE) is specifically designed to provide easy-to-use, object oriented development facilities for the Java developer. The developer interface is both familiar and intuitive, leading to increased productivity without the need for major investments in training. The Vibe IDE borrows from (and extends) some of the most popular and effective features among the best application development techniques in the industry.

Developers creating applications in Vibe will find a rich, configurable IDE that provides multiple ways to view, edit, compile, debug and run projects. The IDE is delivered as a visual interface that contains an editing canvas with various tool palettes and associated editors. The editing canvas is, in turn, a host to a variety of specialized editors for dialogs, images, menus, lists, control items and other UI objects. Geometry management tools (Springs and Struts™) are available for laying out dialogs that become automatically adaptable to changes in the presentation environment at runtime. The IDE provides a navigator for viewing and editing the classes and other project components. A full-featured text editor is available to edit source files. A compiler can be invoked from within the IDE and, upon successful compilation, the application can be immediately run and tested. A robust, interactive debugger is also available from the IDE to help identify and fix programming errors.

Class Libraries

The Vibe Class Libraries are the heart of the product's strength. They deliver the bulk of the functionality embodied in a Vibe-built application component. The Class Libraries are derived from the Visix Foundation Classes which also form the underpinnings of The Galaxy Application Environment®. These class libraries are widely recognized as one of the most powerful class libraries available anywhere - but until now, only available for C and C++. Through the Visix implementation of the virtual machine for Java, these libraries are now available for direct implementation in Java applications. The Class Libraries include a wealth of application functionality in the following general categories: Basic Classes, Utility Classes, Drawing Classes, User Interface Classes, NotebookItem Classes, Text Classes, List Classes, Domain Classes and Database Classes.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

The GUI Class Hierarchy

In standard Java runtime environments, the Java VM relies upon the native windowing tool kit implemented on each target platform to manage the presentation and manipulation of the user interface. The functions, styles and behaviors supported by those various native tool kits vary significantly from platform-to-platform.

Because Java is a fully portable cross-platform language which cannot anticipate or accommodate platform-specific variations in runtime functionality, developers building user interfaces in Java are necessarily restricted to the small subset of functionality that is common among native windowing tool kits to be supported. This "layered" approach to runtime delivery is used by nearly all current Java implementations - including the Abstract Windowing Toolkit (AWT) implementation. The inherent inability to support anything but a very small subset of the most common user interface functions renders this approach uninteresting and, in most cases, insufficient to meet business computing requirements.

In contrast to AWT, the Vibe user-interface classes are built on Galaxy's powerful, "superset" approach to delivering cross-platform services. The Vibe classes provide a greatly improved, complete, native implementation of all user interface functions, styles and behaviors that occur across all supported platforms. This approach is part and parcel of the Ideal System Model discussed earlier in this document. It allows the developer to pick and choose from among the best and most appropriate interface features in building the "ideal " user interface for an application - without regard for whether or not certain features are supported natively on any known or unknown platform.

The Vibe GUI class hierarchy affords the following benefits:

Extensibility - In AWT, simply changing the appearance of the button content requires the user to re-implement the *entire* button class from scratch. By contrast, Vibe allows the developer to override any or all of the individual properties of a component at any time so that it never has to be rebuilt in its entirety. If a completely new component is being created, Vibe provides access to all of the underlying properties required to define the component it its entirety. For example, Vibe provides access to low level resources such as colors and fonts when the developer is creating a Windows button.

Completeness - As discussed above, AWT provides a relatively small set of controls that are common across supported target platforms. Vibe's interface component set was designed, for use in the most demanding top-of-the-line applications. It integrates real-world capabilities like printing, on-line help, undo/redo and drag-and-drop support. Vibe gives developers everything required to build polished, engaging applications in Java with the ability to easily deploy those applications to most major platforms.

Consistency - Vibe's superset approach provides truly consistent behavior across platforms. Vibe provides this consistency by delivering native implementations of the superset of platform functionality across all target platforms.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

Graphical Object Editors

A rich family of Graphical Object Editors is available to assist developers in building application interfaces. The Vibe Graphical Editors implement intuitive, drag-and-drop techniques to create and lay out dialogs. User interface objects for application development are available from customizable palettes within the Editors. Choosers and editors are also available for specifying colors, images, fonts and other features for dialogs. Samples of key Graphical Object Editors include the Dialog editor, the Toolbar Editor and the Image Editor.

The Integrated Debugger

The IDE contains a full-featured, high performance visual debugger to help isolate and fix programming errors. It provides functionality to set breakpoints, browse variable values and the call stack, and view an application's output. The debugger also manages multiple threads.

Support for ActiveX Controls

Any ActiveX control created using Visual Basic or other development tools can be easily embedded in a Vibe application component. The Vibe application functions as a control container where each control site houses a specific ActiveX control. Vibe enables the developer to set the object's type and then easily convert it to a Java Class. Vibe's support for ActiveX enables organizations to leverage their investments in existing object-oriented application components implemented as ActiveX controls. In addition, new object controls, which developers can download from the Web or purchase though a third party software provider, can easily be imbedded into a Vibe application.

The Vibe Execution Environment

The rich object-based VM ensures consistent powerful functionality at runtime with high performance on each platform. It is delivered as part of the IDE for testing, but it is also delivered separately as a browser plug-in or a Vibe "runner" (Vibe VM and runtime) to support the execution of Vibe-built components on target platforms other than the development platform. In addition to Vibe-built Java components, the VM supports the execution of standard Java bytecode and standard ActiveX controls.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

How Vibe Promotes Ease-of-Use

Vibe is easy for developers to use because it is designed to be familiar, efficient, self documenting and configurable for different developer skill levels. It is also designed to meet personal preferences of the developer. Features such as Application Frameworks also promote code reuse to improve development and maintenance productivity. These ease-of-use features are discussed in more detail below.

Application Frameworks

The Vibe IDE provides Application Frameworks as a basis for building applications. An Application Framework is a base class that manages the resources of an application. Frameworks allow developers to easily incorporate standard components, such as Splash Dialogs, Main Windows and Help Menus into their applications. They also provide convenient routines for locating and retrieving objects in the application. Application Frameworks allow developers to focus more energy on application logic and less on housekeeping matters. Developers may customize one of the Frameworks available through the IDE or they may select from Frameworks used in sample projects.

Dynamic Geometry Management

Springs and Struts are visual metaphors that allow developers to interactively define the size and placement of objects in an application screen relative to other objects or the four sides of the dialog window. The application interprets these specifications at execution time to provide automatic adaptation to changes in screen sizes across different platforms or in response to window resizing events. This allows a developer to specify a self adapting interface that is correctly displayed across all supported platforms. The spring solution within a dialog box also allows the application to intelligently and automatically handle difficult item sizing problems such as platform or language specific fonts. This adaptive behavior is accomplished with no incremental coding on the part of the developer!

In a typical development scenario, a developer builds the User Interface (UI) with the Vibe IDE family of UI editors simply by pointing, dragging and clicking. The information on the UI is stored in a platform independent, binary resource file which includes all the information necessary for the VM to perform automatic and dynamic geometry management on the user dialog in the running application. This approach also separates the interface definition from the program logic making it easy to modify the layout of a dialog without having to modify source code or recompile application components.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

Configurability

- Variable API Exposure - This feature makes it easy to configure the level of detail in which the API is presented to various developers with varying levels of skill. For example, an entry level developer might be interested in only basic methods pertaining to simple object rendering. A more sophisticated developer might instead require access to much greater detail in the API in order to change how an item responds to user input from the keyboard. All IDE editors and browsers are eligible to be configured in specific ways to meet the needs of a development organization.
- Interface Customization - The Vibe IDE is also completely user configurable to meet individual developer preferences. It enables editors to be placed in their own windows or to be included in one window. Developers can also specify that editors and associated tools remain self-contained in the host application.

On-line Documentation Management

Vibe's on-line help and documentation management is divided into three related areas:

- On-line help for the Vibe IDE
- On-line manuals (Vibe Reference Manual, Programmers Guide, Tutorials and Building Vibe Applications Manual)
- Authoring tools for developers to create application-specific Help functions

All hyper-text help is HTML-based so developers can use an HTML browser of choice.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

The Future of Vibe

Database Connectivity

Vibe has been architected to support a client/server model that uses high-performance intermediate database services between the application and the database. Low level database classes will provide support for SQL operations and data access. Higher level classes providing a graphical query builder, a database-aware “form” object with a sophisticated validation hierarchy, and database-aware lists and combos will be available in a future version of the product. These classes will not be a part of Vibe DE 1.0 but will be available in beta during Q2 1997. Oracle, Sybase, and ODBC will be supported in the first database release, with JDBC support in a future release.

Distributed Object Programming

Most existing Java development tools provide support for distributed programming through Sun's RMI and Java IDL packages. Vibe will soon allow objects created in one application to be shared by any other application without modification. A Vibe application will be able to export any object it creates to the network and obtain access to any object exported by another Vibe application. The distributed object system in Vibe preserves standard "pass-by-reference" calling semantics for remote objects. The distributed object system also transparently integrates with the local garbage collector, so remote references are automatically reclaimed when unreachable. Marshaling data to and from the network is automatically handled by the Vibe VM.

Vibe's distributed object model is under development and a beta version will be available for testing in Q2 1997. Vibe's distributed object system will use the Internet Inter-Orb Protocol (IIOP) as defined by the Object Management Group in the CORBA 2.0 specification. Visix intends to provide a full set of CORBA 2.0 Core services, including Interface Repository and general ORB facilities. In essence, a full CORBA 2.0 Core ORB will be built into each Vibe VM.

Visix Software, Inc.
11440 Commerce Park Drive
Reston, Virginia 20191
703.758.8230/Telephone
703.758.0233/Facsimile
800.832.8668/Toll Free
http://www.visix.com/Web

Summary

Use of Intranets for corporate computing is becoming a reality and it makes strong business sense to consider this alternative architecture for many types of applications. Development staffs must utilize appropriate development tools to take full advantage of the benefits offered by Intranet computing and the Java language.

Vibe provides developers with a reliable and proven development capability attuned to Java. With Vibe, Visix leverages its strength in client/server computing and Java to provide organizations with next-generation application development solutions. Visix technology simplifies the complexities of building and deploying business-critical applications for distributed, multiple computing environments.

Visix Software pioneered the design of modern, windows-based, application development tools that allow large companies to move to client/server environments. With Vibe, Visix becomes a leader in both client/server and Intranet development.

For more information, call 1.888.876.VIBE, e-mail vibe@visix.com or go to http://www.visix.com