814-234-9613
814-234-9614 Fax

VVI Data Control Specialist

311 Adams Ave.
State College, PA 16803

Graph Object Library

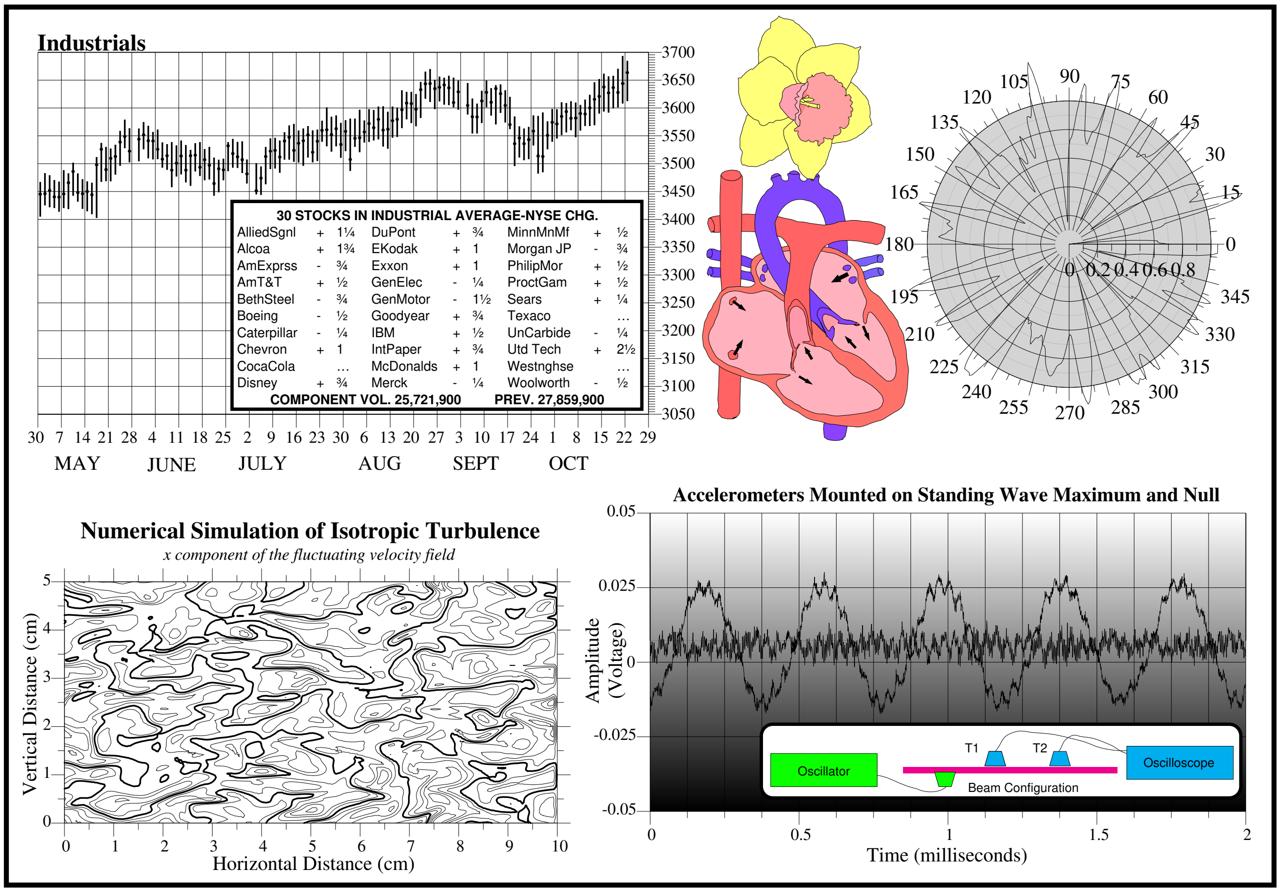
Features and Services

The VVI-Graph object library delivers professional interactive, animated, and programmable objects for data control, acquisition, display, and graph document layout. It is used for trading-systems, forecasting, engineering and scientific data display, and any process where quality custom data presentation and control software for production or interactive data needs are required.

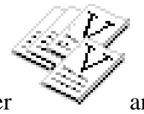
This library enhances your projects by providing:

- Reliable, powerful, and optimized performance.
- Substantial savings in development time and quality assurance.
- Access to your data in a quality and interactive format.
- Substantial user interface, data importing/exporting, and general graphic features and effects.

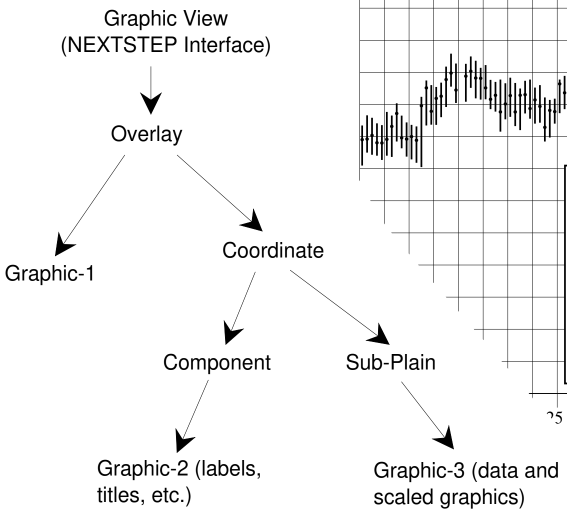
Graphs and figures, such as those below, are effortlessly constructed without extensions. Animation and user selection are built in, producing an implementation responsive to changing data and your changing requirements.

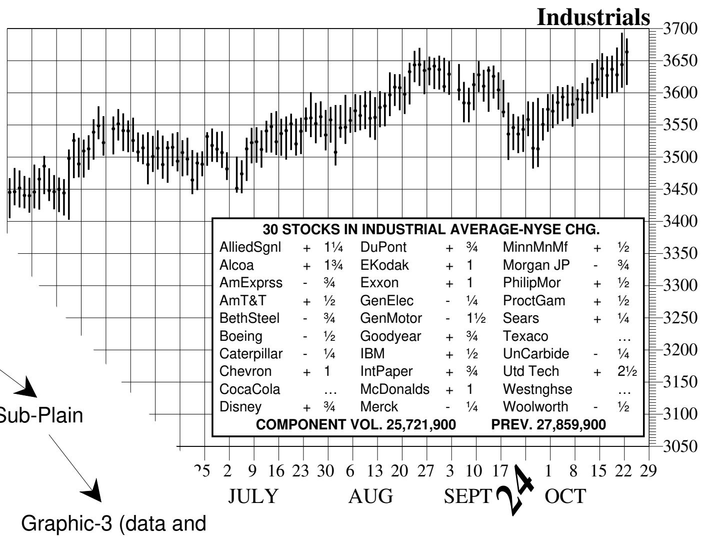


Architecture Overview

The VVI libraries, including the Graph library, are sorted UNIX® archive (.a) libraries, just like the NEXTSTEP appkit library, and are pre-built in Motorola and Intel versions. The library headers, resource files, and archive files are contained in the VVLib folder  and are sorted much like the appkit and regular unix headers and resources so importing linkage specifications to your source code is efficient and systematic. The library consists of about 50 Objects and 90,000 lines of code. The application *GraphBuilder* is included to prototype and test your custom graph interactively.

The architecture and accompanying user interface and program control permits easy to do page layout of complicated and animated graphics in a modular easy to work with way. The architecture is based on an inter-dependent hierarchy of *graphic*, *coordinate*, and *graphic-plain*. The *graphic* is the basic component and encapsulates the graphic and data attributes and functions. The *coordinate* is the axis and accompanying coordinate system where the *graphic* resides. The *graphic-plain* maintains *graphic* dependencies and events. In addition the *coordinate* and *graphic-plain* are themselves graphics. At a particular point in time the graph at the right below corresponds to the chart at its left.





In the above chart, the *graphic view* is the interface to the NEXTSTEP window system. The *overlay* is a list of *graphic-plains* which represent overlays (transparencies). The *coordinate* is divisible and consists of two orthogonal *graphic-plains*. These in turn group additional graphics. Any *graphic-3* may be a *coordinate* or *graphic-plain* so the hierarchy can be extended indefinitely. Each *graphic-plain* (*overlay*, *component*, *sub-plain*) maintains several orthogonal lists of graphics and graphic state information. This results in efficient interleaving of selected, animated, and focused graphics.

This structure permits a very powerful hierarchical coordinate and page layout system. In the graph above, for example, the axes labels are contained in the *graphic-2* . Since *component* controls user events, the single x-axis label,  , may be rotated, skewed, scaled, and otherwise changed by user or timed events, inspectors, or program control. Similar control may be exerted at will via mouse, keyboard, inspector, and program control for all graphics. If this end user control is not preferred it can be disabled. Because values are cached at multiple levels of the hierarchy as well as within the same level the implementation is very efficient yielding a very responsive system overall.

Features

A major feature of the library is the high degree of reliability, completeness, and inter-operation of its components. The following are some additional distinct features available without extensions.

- **Prototyping.** The graphics, document, and control loops can be developed and tested using the application *GraphBuilder*. This methodology is similar to the NEXTSTEP Interface Builder.
- **Data Entry.** Data may be entered by drag/drop of file icons, palettes, pasteboard, program module hooks, or mouse editing without extensions. Minimal code is required for database and data acquisition server integration.
- **Document System.** Documents are stored as multi-level UNIX directories. Internal document file and program module links maintain relations for large data sets and relocatable object code.
- **Controllers.** The graphics may be stored and locked programmatically so the end user has control over only the features you permit. The graphics issue *owner* and *target* messages so your application can react to changes in graphic attributes and editing. Thus the graphics can be used to control auxiliary devices.
- **Data Graphics.** Any number and combination of histogram, line, smoothed curve, areas, scatter, contours, and signals on multiple coordinate systems. Since graphics, such as arbitrary connected spline graphics, maintain their data attributes they can be used to display data appropriate to that format.
- **Axes.** Axes and associated coordinate systems are highly formalized within the system. Rectangular, polar, and semi-log axes are provided. Other axes and coordinate systems can be easily integrated.
- **General Graphics.** Bezier curves, lines, ellipses, ovals, parallelograms, polygons, rectangles, rtf, eps, quadrilateral image warping, text wrapping, groups, smoothing, shadows, interpolated fill, etc.
- **Palettes.** Any graphic, and hence data or database interface, may be stored and retrieved from palettes. This includes entire graphs.
- **Point and Click.** Every graphic implements a full complement of point and click mouse control.
- **Inspectors.** Graphic attribute can be modified directly from inspectors. Inspector-Editors maintain inspector user interface controls and relations between the associated graphic attributes. Inspector-Editors are linked to the graphics and between each other in a general network topology by a set of inspector-editor links and hierarchies.
- **Function Modules.** Control loops, animation, drawing and any other feature can be controlled programmatically using relocatable object modules accessible through the program module editor which supports RTF formatted Objective-C, c++, postscript , and server source code as input.
- **Optimized Drawing.** Drawing is optimized and takes full advantage of the most advanced postscript optimization techniques as well as bitmap caching and minimum bound clipping.
- **Standard Conversions.** Conversion to eps, tiff, and ascii formats are included and supported in the pasteboard and file dragging mechanisms of NEXTSTEP.
- **NEXTSTEP Compliant.** Standard NEXTSTEP utilities such as the color panel, font panel, print and fax panel, services, save and open panels, and pasteboard, are seamlessly incorporated and used to their full extent.

Support And Service

The Graph object library delivers a high level of data and graph capabilities. To help make the most of these capabilities VVI provides quality support which focus on your requirements for timely and smooth results. These include:

- **On Site Instruction.** Introduces the architecture and features of the VVI-Class, Kit, and Graphic libraries. Shows how to write applications that control and display data in a graph format, and how the libraries fit into NEXTSTEP. Although experience with NEXTSTEP is recommended, it is not required.

  Syllabus
  - *Demonstration*: Demonstration and use of the application *GraphBuilder*.
  - *Conceptual Model*: Graphic-Plains, Coordinates, and Graphics.
  - *Class Overview*: Segmentation and explanation of the Class, Kit, and Graphic libraries and their respective classes.
  - *Custom Graphics*: How to write your own graphics, with examples provided.
  - *Postscript Optimization*: General discussion of user paths, caches, user objects, and selection of coordinate systems.

- **Technical Support.** Provides support for the VVI libraries in areas such as custom graphics, user interface integration, database integration, graphics optimization, and Objective-C, c++, and postscript language coding.

- **Contract.** Provides reliable and proven expertise which focus on and satisfy your projects goals.

- **System Integration.** Provides comprehensive support and service for all aspects of your custom data application and computing needs. VVI is an authorized system integrator, developer, and reseller for many computer manufacturers as well as an authorized reseller of NEXTSTEP.

Licensing

Licensing is designed for both commercial and in-house developers and include the following formats:

- **Developer.** For use where the object libraries and resources are incorporated into applications for in-house use and distribution only. The libraries and technical support for the first year or on-site instruction must be purchased. Up to 5 in-house developers may use the libraries without additional fees.

- **Commercial.** For general use and distribution of your application with the VVI object code incorporated. The libraries, technical support for the first year, and on-site instruction must be purchased. Additional fees are structured on a royalty basis or single up-front purchase. Contact VVI for additional information.

- **Source Code.** For distribution of your application with the VVI source code incorporated in object form. The libraries, technical support for the first year, and on-site instruction must be purchased. Contact VVI for additional information.

VVI Graphic Documentation V3.2 September 1993; Rev. 1993091901