VisualAge for Smalltalk
A Powerful New Vision of Programming
(Updated for July 96 CD)
June 18, 1996
Overview Presentation Script

Software Solutions
RTP, NC 27709

Contents

Presentation Guidelines

Introduction

This presentation was developed to stress some key design features of the product and to address some of the key question areas that we have seen in numerous VisualAge for Smalltalk presentations . These notes are intended as an outline to the presentation and meant to provide both insight and guidelines in preparing for your own presentation. Questions, hints and feedback may be directed to ADOPP @ CARY.

VisualAge for Smalltalk Version 3.0 for AIX, OS/2, and Windows

Introduction

The flow of this presentation is:

1. Discuss the current state of IS and Client/Server in order to surface some of the problems that our customer set is facing. These are the IS Challenges which lead into an overview of the Client/Server Powertool market.
2. Drill down into some key purchase factors for customers evaluating Client/Server development tools and set up a framework for discussing the features of VisualAge for Smalltalk.
3. Review key product features and design points based on this structure.
4. Overview of product, services, prices and estimated footprints.

From Crisis to Competitive Advantage

The Software Crisis is a notion that was developed by author, David Taylor, to describe the multitude of pressures and issues that the IS manager is under in the 90s. These items tend to set the stage for why new development methodologies and tools are needed to tackle these problems. All of these issues are similar to those we are facing in IBM, so you may want to use some personal experiences to ensure buy-in on these charts.

Development Backlog: Most IS shops have more projects and requests than they can handle with existing staffs. This need is driving them towards Rapid Application Development (RAD), Reuse, CASE and Object-Oriented programming Projects cannot continue to be built from scratch as custom components.

Reduce Staff and Development Costs: Simply put, IS managers are having to do more with less.

End User Customization & Flexibility: As end users become more computer-literate, they are demanding more out of the IS developed programs. They want GUIs, multimedia, customization and systems that do not enforce rigid frameworks. Several customers have stated that if their users see something in their personal productivity apps, they expect this in their custom software within the next year.

Heterogeneous Environments: In the age of mergers and right sizing, there is a strong need to connect odd assortments of both hardware and software. It is becoming very rare to run into pure IBM shops, thus the need for software development tools that act like the glue of the organization and connect existing systems to new.

Reuse of Existing Systems: This is a key requirement of Client/Server tools to fit within existing Information Environments. Thus the need for support for multiple databases, communication protocol server systems and utilities.

Programmer Skills: Very few IS shops are graced with a deep supply of OO, C/S, and Workstation programmers. One of their key concerns is how do they get their current set of skills migrated to a new tool set. Are consultants needed? How long is the training period? Are the skills learned strategic?...

All of these issue together can simply bury an organization. Especially as the rate of change and the competitive pressures of today′s marketplace multiply. Businesses today need to be nimble, flexible, and responsive to change. They need to be able to strike quickly with new products and services. They need tools that can adapt readily and that can become a weapon for competitive advantage rather than a source of crisis.

Making this leap requires a new approach. Business Process Re-engineering, Client/Server, and Object-Oriented Programming are increasingly popular techniques that offer a new approach.

These types of issues set the stage for the search for the perfect Client/Server tools to help customer overcome these issues.

Choosing a Powertool

This chart is based on a recent report from CGI which outlined some key purchase decisions for selecting a Visual Programming Tool.

Scripting Language: Most of the Powertools are based on a scripting language for the actual programming. Popular ones are PowerScript, Basic, Digiscript and Smalltalk. Key decision points on this feature focuses on how proprietary and extensible is the underlying language. Does it allow for full programming flexibility or does it limit programmability. VisualAge for Smalltalk is based on Smalltalk, which is a true programming language (details next chart).

Object-Oriented Features: OO is becoming one of the most overused words in the industry right now and every vendor is using it very liberally in their marketing literature. Primarily you have to be based on a true OO language to meet all the OO criteria. POWERBUILDER is not OO!! VisualAge for Smalltalk, being based on Smalltalk is a complete OO environment.

Portability: Portability across numerous client platforms is one of the key requirements for a Powertool to address heterogeneous environment. In evaluating this claim, you need to look at how the product achieves portability: Some only provide a limited portable interface, thus forcing the developer to a least common denominator approach which limits their functionality; Some look for middleware to port one platform to another (PowerSoft is using middleware which replicates the Windows API on other platforms) which lead to applications which do not look native; some products require minor porting (Digitalk); and others offer excellent portability but lead to odd hybrid interfaces (ParcPlace). VisualAge for Smalltalk has learned from these efforts and been architect specifically for portability (more in next chart).

The world of widgets: It is highly unlikely that one vendor can ever hope to provide all of the necessary components for all applications. Thus there has been an increase in parts based Application Development. Some vendors have an excellent 3rd party parts program (Microsoft), others have none (Enfin). VisualAge for Smalltalk has begun the Object Connection program to address this need (detail chart included).

Tools Extensibility: Many tools do not allow the programmer to leave their environment. Key things to look for are: Call out/Call in, access DLLs, SOM support, DRDA support and communication protocol support. VisualAge for Smalltalk has an excellent story here (detail chart included).

Team Development Environment: Any project greater than 5 developers will require team development support for work coordination, version control and code maintenance. One weakness of several OO based solution is that they use PVCS based team environment which is not OO based, but file based. VisualAge for Smalltalk has integrated the OTI ENVY technology into the VA Team product to offer fully integrated OO based team development environment (detail chart included).

Enterprise Scalability: Does the tool support their other IS strategies for data, communications and transactions. Here again, this is a strong selling point for VisualAge for Smalltalk (detail chart included).

This chart should allow you to discuss and highlight some issues that your customer may not have considered in their evaluations to date. This chart also acts as a framework for the rest of the presentation to outline VisualAge for Smalltalk′s key strengths in these areas.

Enough introduction, on to the product...

VisualAge for Smalltalk: An Overview

Let′s start first with a quick overview of VisualAge for Smalltalk - what is it? Who′s it for? Then we′ll move on to talk about the details of the product using CGI′s evaluation criteria as a guide.

VisualAge for Smalltalk is an integrated development environment for professional programmers. It was designed with Client/Server in mind from the get-go, and with the idea that Application Development needs to be quick, flexible, and adaptable to change. It can be used for a variety of applications, from departmental decision support to applications that you run your business with. Best of all, it is open and extensible.

At the heart of VisualAge for Smalltalk is a new and complete Smalltalk development environment, including the full language, class libraries, browsers, and debuggers that are the state of the art today. This base enables Application Development and execution on AIX, OS/2 and Windows today.

We′ve extended the Smalltalk base with a broad set of class libraries for data access, remote application and transaction access, communications, GUI, Multimedia, Distributed Smalltalk, and Report Writer. In fact, a complete set of class libraries and frameworks has been developed around each of these major areas. New for July 96 is the Web Connection and Connection for Lotus Notes.

But the real essence of VisualAge for Smalltalk is the unique integration of this pure object-oriented base with a powerful new capability called Visual Construction. That is, the ability to create an application by dragging reusable software components (or parts) from a palette and piecing them together visually to form a new application. The idea here is to take reuse to a new level, and to ease your transition into the world of OO.

Finally, VisualAge for Smalltalk comes with a set of fully integrated professional programmer tools. For anyone serious about doing team oriented Client/Server Application Development, the need for capabilities like collaborative development, version control, configuration management, and performance tuning is clear.

Object-Oriented Features: Principles

These next two charts provide a general overview to the concepts of Object-Oriented Programming. If your customer is familiar with OO, then skip the next two charts and go directly to the check list.

This chart reviews the four main features and selling points of using an object oriented approach to OO:

Applications model your business: Object Oriented systems allow architects to design systems that replicate real world objects. Thus your designs can more closely reflect the real problems that they are intended to resolve and reduce some of the confusion that traditionally results when translating real world problems to computer design terms.

Construction from Components: Traditional application development does not provide a true mechanism for reuse, thus each new application is a custom piece of software. With OO, reuse mechanisms are built into its′ base model so that object reuse and sharing can be accomplished much easier.

Integrated Logic and Data: This aspect of OO allows for work and programs to be componitized, which eases both development and maintenance. It is not necessary to pass large static control blocks around the system. Communication is done through message passing between objects and each object is responsible for their own data.

Iterative Development: Object Oriented environments greatly facilitate iterative development. The notion is that programs can ″grow″ rather than be built. The interpretive nature of Smalltalk allows applications to be developed and tested quickly which then allows them to be reviewed quicker with the user community for approval or enhancements.

OO: A New Paradigm

As with any new technology, OO has its own vocabulary. This is not meant to be an exhaustive list, but helps with some of the key words.

Class: The basic building block of OO. A class defines what an object is through its′ data and the methods that it responds to. Thus, a circle would have data for radius, border, color etc. and it would respond to the messages: Draw YourSelf, ChangeSize, ChangeColor etc. Once a class is built, an object can be declared which are instances of that class. Circle A, Circle B, etc.

Inheritance: Inheritance is the notion that a new class can inherit some behavior from existing classes. A simple analogy would be with foil presentation programs that allow you to set up a master foil for the general outline of the presentation which can then be used as a template by the whole presentation.

PolyMorphism: Big word, simple meaning...Polymorphism is the ability to have multiple objects respond to the same message, but react in different ways. Imagine that you are at a stoplight and there is a pedestrian, a cyclist and a car. When the light turns green, they all respond to the same message (GREEN LIGHT= GO), but they react in different ways: The pedestrian takes a step, the cyclist starts to pedal and the vehicle presses the gas. Polymorphism helps to greatly simplify the message call structure and allow for designs with fewer distinct messages.

Encapsulation: As mentioned before, each object owns its own data. This simplifies maintenance and program modification since you don′t have to chase through your code figuring out who uses the BitFlip flag. Individual objects can be modified and changed with minimal impact to the system.

Frameworks: Frameworks is one of the newer OO terms and it refers to a reusable template for building user applications.

Class Libraries: Class Libraries are just collections of classes that provide similar functions. They may or may not build upon themselves to provide a whole subsystem like frameworks. Examples would be string Classes, communication classes, or data access classes.

Object Model: I threw this in just to drive home the point that there are different object models and distinctions between the languages. C++ is very different than Smalltalk based on language, memory management inheritance models etc. This is where SOM comes into play since it provides a mechanism for different object models to communicate with each other.

VisualAge for Smalltalk: Object-Oriented Features

This chart just summarizes some of the key OO features of VisualAge for Smalltalk that sets it apart from other products.

Pure OO Environment: Since VisualAge for Smalltalk is written and based on IBM Smalltalk, you have the purest OO environment available. Even when adding two messages together you are sending messages between two objects. Smalltalk also enforces OO since it is such a pure environment. It is very hard to write COBOL in Smalltalk.

Object-Based Configuration Management: This extends our focus on OO. The team version of both IBM Smalltalk and VisualAge for Smalltalk use an object, not file based library system (detail foil follows).

Support of Inheritance: Smalltalk is a single inheritance model. C++ is a multiple inheritance model and the wars rage on in the Object community as to which is better. Single inheritance tends to simplify both product design and support. No matter what the audience says, single inheritance is NOT a weakness. The key is that both visual and non visual objects can inherit behavior from other objects thus allowing reuse and customization of previously developed object.

Class-Based Visual Construction: Every part that you build in Visual Age is a new class thus the code generated from these parts can be browsed, edited, etc. with the Smalltalk tools. This model ties very directly to the underlying Smalltalk and does not introduce additional models to the programs. This is a VisualAge for Smalltalk strength against the Instance based model of Digitalk′s Parts product.

Parts Architecture for Extensibility: The notions of parts and components have gone a great way in helping legitimize the use of OO for commercial development. VisualAge for Smalltalk provides a published parts architecture that allows it to be extended by either custom or commercially available parts. These parts can be directly added to the VisualAge for Smalltalk Palette.

Reuse: Reuse is possible in VisualAge for Smalltalk through classes or parts. Providing several layers for reuse allows the customer the greatest flexibility.

SOM Support: VisualAge for Smalltalk is the first Smalltalk based product to support the use of SOM objects. SOM is IBM′s key strategy for object reuse between various OO languages. SOM is based on the CORBA (Common Object Request Broker Architecture) architecture from the OMG group.

VisualAge for Smalltalk Language: IBM Smalltalk

IBM Smalltalk is the underlying scripting language of VisualAge for Smalltalk. It is a complete programming language. We have a very competitive product when compared to ParcPlace and Digitalk. IBM Smalltalk was called ″Industrial Strength Smalltalk″ by PC Week when they did their initial review of VisualAge for Smalltalk. Additiionally, in an October 9, 1995 InfoWorld Product Comparison of Visual Smalltalk Enterprise 3.0, VisualAge for Smalltalk Team 2.0, VisualWorks 2.0, and Object Studio 4.1, VisualAge for Smalltalk Team 2.0 scored second. And with VisualAge for Smalltalk Version 3, no doubt, VisualAge have leap-frogged the competition.

This chart shows two views of the underlying Smalltalk: System Layers & Programming Subsystems.

System Layers

Development Environment: At the highest level of our Smalltalk is the development environment, which is a fully integrated set of browsers, debuggers and editors. They provide for a very dynamic development environment which is extremely productive.

Portable Classes/Methods: Smalltalk is made up of two major sub-component: the Image and Virtual Machine. The Image provides a very rich set of Smalltalk classes. IBM Smalltalk ships over 1250 classes. This is larger than any of the competitors. With VisualAge for Smalltalk, we add another 1250 classes, so the developer will find an extremely rich set of components to build with. IBM Smalltalk is a layered image, with the highest layer called our portability layer. If the developer programs to these classes, then their code will automatically port to other environments. Additionally, portable containers parts (i.e. notebooks,...) have been added to enhance portability between platforms. The portable notebook provide capability that may not exist on the target platform. These classes are labeled in the image so it is easy to govern their use. This set is very robust: VisualAge for Smalltalk itself is built in Smalltalk and approximately 93% of our code is at this level.

Platform Specific Classes & Platform Interface Classes: These classes allow the programmer to build code specifically for an environment. An example would be if you needed to write directly to the OS/2 desktop or do OLE programming for Windows. This code is not portable since it is platform specific. IBM Smalltalk does provide abstractions for many of these services, so that the programmer can deal with higher level objects. The Interface Classes, for example, allow the Smalltalk programmer to get the same level as any C programmer. Thus IBM Smalltalk provides a very flexible programming model that does not limit the programmer. They can maximize for either portability or for specific system services.

Virtual Machine: This is the heart of IBM Smalltalk and its′ role is to provide an interpretive environment that allows for immediate test and execution of applications.

Programming Subsystems:

This view of IBM Smalltalk reinforces some of our key strengths. One of the key issues that we heard from customers that was keeping them from moving to Smalltalk was the lack of Standards Support. All of IBM Smalltalk′s subsystems are modeled on existing or emerging Industry Standards.

Data Types, Language, Processing Model: These three subsystems are based on the emerging ANSI Smalltalk standard. IBM has been very focused on supporting an ANSI standard. In the early 90′s we lead an effort with the other major Smalltalk vendors (KSC, OTI, Digitalk, ParcPlace) to outline a model for commonality across these subsystems. This work resulted in a Red Book and was referred to as the Common Base. IBM then took the Common Base work and approached ANSI on formalizing a language subcommittee for Smalltalk. This occurred in 1993 and Dr. Yen Ping Shan (IBM RTP) was elected as committee chairman. A full standard is expected within the next three years.

Common file System, Widgets, Graphic System: For these subsystems we made the decision to make their programming interfaces based on industry standards, thus we selected POSIX for file, MOTIF for Graphical Widgets and X for the graphics layer. This provides a model where the programmer learns and uses industry standards that IBM in turn converts to native system calls. This provides for a highly portable system that allows applications to utilize native system services. This approach is a very key strength and is superior to both Digitalk and ParcPlace. Digitalk develops their classes very close to the API layer which limits portability (i.e. the coordinate systems are different across their Windows and OS/2 products). ParcPlace focuses on a degree of portability, that is applications appear to have a neutral graphic and behavior model but do not feel 100% native.

IBM Smalltalk is based on technology from Object Technology International which has been providing Smalltalk engines to the industry for over 7 years. They are a very well respected firm in the OO committee, thus you should use this fact to dispel any questions on the newness of our offering. IBM Smalltalk is a very rich, robust and stable environment.

VisualAge for Smalltalk: Portability

This chart is intended to drive home the fact that VisualAge for Smalltalk has been specifically designed for portability. As mentioned earlier in the presentation, Smalltalk is divided into two major components; the common Image and the Virtual Machine. All coding is done at the image level by extending or adding new classes. For those classes and frameworks provided by IBM, a great deal of effort has been placed into providing stable common APIs at the programming level to greatly assist in programmer flexibility and portability.

As mentioned in the Smalltalk Chart, we use industry standards for File System, Window Manager and Graphics APIs. We have also taken this notion to our Database and Communications feature to allow other databases or communication protocols to be interchanged with minimal or no code change. As with our other subsystems, the programmer will find the highest programming layer designed for portability, but they may drop down to more specific layers for specific programming needs.

The Virtual Machine line outlines our intended platforms. All platforms will be shipped with each new version, so new features will be introduced across all platforms concurrently. VisualAge for Smalltalk supports AIX, OS2, and Windows today. Windows 95 and Windows/NT support will be added in 1Q96, HP & SUN in 2Q96, and MVS Smalltalk in 3Q96. Our basic strategy will be that Smalltalk will be offered on a new platform first, followed by other products based on interest.

VisualAge for Smalltalk Widgets: Parts Architecture

It is highly unlikely that any vendor can produce all of the product extensions needed by a customer to build their specific application. As a result, it is a key selling point for a powertool to be both extensible and have a program where 3rd party vendors can contribute parts.

While VisualAge for Smalltalk ships with hundreds of parts in the base product, we offer the following additional packages:

MultiDatabase: access to Oracle, Sybase and SQLServer. COBOL: the ability to wrapper existing COBOL DLLs and incorporate into VisualAge for Smalltalk code. Communication & Transaction: support for popular communication protocols. MultiMedia: part abstractions and devices for Audio and Video.

On top of these packages, we also have other groups in IBM working on VisualAge for Smalltalk Parts:

AS/400 Connection: these parts will give direct access to AS/400 Data, Files and Programs. This work is part of our strategy integration with the ENVY/400 product. Image: our lab is currently producing parts that will interface with the ImagePlus product

Third Party Parts: one of the most exciting areas that we have undertaken is the Object Connection Program. We have set up a team, lead by Rick Runyan to develop relationships with third party developers for independent part development. This program has been readily accepted by the vendor community and we now have over 165 IBM and external groups producing over 325 products. These products are listed in the VisualAge for Smalltalk Resource catalog. Products which have announced or shipped include:

ObjectShare Inc: Advanced Widget Kit - Additional graphical widgets
including spreadsheets, business graphics etc.

Cirrus Technology: Image processing and display parts.

GemStone Object Application Server for VisualAge for Smalltalk

HOMSuite - Analysis and Design Tool

A catalog of the various parts available will be published on a regular basis and will be posted on the internet. The URL is
WWW.SOFTWARE.IBM.COM/SOFTWARE/AD/VISAGE/VARC/VARCOCPL.HTM

VisualAge for Smalltalk: Tool Extensibility

This chart simply shows how extensible VisualAge for Smalltalk actually is. A customer should view VisualAge for Smalltalk as fitting within their current environment and the high degree of interface to existing code and protocols will allow them maximum reuse of their existing systems.

This chart is fairly self explanatory but a few notes:

All Communication protocols are offered in the Communication Feature. IMS connectivity is offered in the IMS Connection Feature.

DB2 family native access is available in the base product as well as ODBC support. Through this support you are able to participate in the DRDA strategy for remote data access.

Oracle native access is available in the Oracle features. Note the three items in Database Access listed in blue. Sysbase and SQL Server are available via ODBC and Gemstone is available via 3rd party provider.

C Wrappering and COBOL are provided in the base product.

Our current SOM support allows one to use SOM objects, not create.

VisualAge for Smalltalk: Distributed

The Distributed Smalltalk feature allows you to create complete client/server applications from within a single development environment. And you develop these applications in the same manner as other applications with VisualAge for Smalltalk. This capability is implemented via our Distributed Smalltalk architecture. Each machine running distributed Smalltalk is considered an object space. Each object space contains a portion of the distributed application. The application is made up of classes, just like a non-distributed application. And these classes communicate with one another via standard Smalltalk message sends. This means that when you build a distributed Smalltalk application, you are not concerned about such things as the physical address of the object space you want to send a message to. It also means you can focus on the development of the application first, and then performance tune the application as you distribute the application.

A full set of tools has been provided to help developers and administrators build and deploy distributed applications. The tools are:

- Distributed debugger
- Distribution matrix
- Event profiler
- Event services framework
- Name server browsers

- Name server distributed part
- Object visualizer
- Remote browser
- Remote file dialog
- Remote inspector
- Remote transcript
- Remote workspace.

There are three application types that are well suited for distributed Smalltalk; they are:

- Object Server
- Active Server
- Peer-to-Peer

Object Server applications are based upon the idea of the machine where a data store is located, providing a object layer in front of the data store. This helps to reduce network traffic between the client and server by eliminate the flow of data manipulation language, (such as SQL) from the client to the server. It also helps to provide for greater independence from the underlying database by removing the direct use of a specific data manipulation language from within the client application. With the Object Server application type, objects on the client workstation send messages to objects on the server. The server objects take care of retrieving from the data store and passing the information back to the requesting client object.

Active server applications change the traditional approach of the client/server computing model by allowing the server to initiate communication with the clients. An example of this would be a publisher-subscriber scenario. The server publishes events it can handle, and client can then subscribe to those events in which they are interested. Let us say a stockbroker is interested in current prices of some specific shares. In an active server environment, the stockbroker can ask the server for the available share prices and then subscribe to those stocks that are needed. Whenever the price of a subscribed-to stock changes, the subscribers are notified automatically.

Peer-to-peer applications break the traditional approach of all clients being attached to a server, and therefore, passing through the server if they want to communicate with one another. This requirement of passing through a server adds overhead to a network and to the server. Peer-to-peer applications can be useful when two users on different client workstations have to establish a session for consultation or negotiation on a certain subject.

VisualAge for Smalltalk Team Development

Team development for VisualAge for Smalltalk and IBM Smalltalk is another key selling point for the products. We have embedded the OTI ENVY team development technology seamlessly within the VA and IBM Smalltalk team products. The OTI ENVY technology is widely respected in the Smalltalk community and is recognized as the best environment for multi-developer projects.

The Team product allows the team to set up an Object Based repository which manages new and existing objects. This repository may be used across numerous projects, thus allowing for reuse. Once the repository has been set up, then each developer may have their own custom image to develop their portion of the project. This motion is similar to giving users views of a database. This feature allows each developer to work independently without having to carry the overhead of the full repository.

This environment is also object based which allows it to seamlessly integrate into the other tools of the Smalltalk environment. Ownership can be set up to the object or method level, thus allowing for much finer ownership granularity than over traditional file based systems. The Team products also manage code migration and allow multiple version of the code to be maintained.

In addition to its library functions, the Team environment provides additional tools for profiling, performance tuning and packaging. With Version 3 you even have the flexibility of having AIX, Windows and OS/2 client using the same repository.

The notes on the chart cover the benefits from the user, team and company perspective.

VisualAge for Smalltalk IMS Connection

The IBM Connection provides the capability to create/instantiate business objects from data stored in existing mainframe databases. The business objects that are created can then be used in applications developed with VisualAge for Smalltalk. This is achieved by mapping the data elements that are retrieved via existing IMS transactions into the instance variables of the business objects. The IMS Connection provides all of the support required to define the business object, define the layout of the existing transaction, and to do the mapping of the data elements to the business object. The IMS Connection uses the communication base classes that come with the Communications and Transaction feature; the connection to the host IMS system can be via either LU2 (EHLLAPI) or LU6.2 (CPIC).

The existing transactions are not modified one bit. So that means the existing 3270 users are not affected by this. Also, once data elements are retrieved, if they are updated or changed, then the existing transaction can provide the persistency mechanism to keep the host database in sync with the instance variables of the business object on the workstation.

Business objects are defined by using the business object base classes that come with the IMS Connection; the transaction flow and data elements are defined by using the transaction record classes. The mapping of the data elements between the transaction records and business objects is done by using the transaction object base classes.

The business objects that are built can be used on any platform that supports IBM Smalltalk now, and in the future. Today that means that they are used on the workstation, but tomorrow they will be able to be placed on the host.

Reports

The VisualAge for Smalltalk Reports feature is an object-based report writer. This means that you can report on any object contained in the VisualAge for Smalltalk image; you can report on the classes contained in an application, you can report on the rows retrieved into a query result table, and you can even report on a visual part, (providing you a way to create documentation for your applications).

Reports can have headers, footers and a body. The body can include a number of different elements, including lines and images, (pictures). You can calculate averages, totals, count, minimum value, maximum value, and current. You can also add field breaks to your report.

A print previewer is provided to help you determine how the report will look, before it is sent to the printer.

A report view is provided to allow you to display the reports on the users display.

Since you can report on anything in your VisualAge for Smalltalk image, it is an excellent way to create documentation on the contents of you VisualAge for Smalltalk ″system″.

VisualAge for Smalltalk, Web Connection

- New Add-On Feature of VisualAge for Smalltalk, V3.0

  The Web Connection is a new feature of VisualAge for Smalltalk V3.0. It was announced on June 18 and is generally available July 25. The Web Connection runs with VisualAge for Smalltalk Professional for WIN 95/NT, OS/2, and AIX. As VisualAge for Smalltalk becomes generally available on other platforms such as HP and SUN, so will the Web Connection.

- Extends VisualAge for Smalltalk to build Web applications.

  The Web Connection extends the VisualAge environment allowing customers to build Web applications. You build a Web application just as you would a VisualAge for Smalltalk application, that is, by using the VisualAge composition editor to build dynamic Web pages to give you the user interface for your applications. The program logic is created by making VisualAge connections with pre-built parts and writing IBM Smalltalk scripts.

- Builds live sophisticated, business-critical Intranet and Internet applications.

  The Web Connection allows developers to build Web applications that integrate with databases such as DB2 and other middleware such as TCP/IP, MQ Series, CICS, AS/400 and IMS. Customers can now build web pages that are live and interactive rather than just static text and graphics.

- Key Features

  The Web Connection provides a set of visual and non-visual parts that can be used to design Web pages. In creating your Web application you can still use the existing VisualAge for Smalltalk non-visual parts and class libraries. Also provided is a communications link between your VisualAge Web application and the Web server.

Web Connection: Easily Create HTML Pages & Application Logic

With the Web connection, you can rapidly and easily create your Web application. Web Connection gives you:

- Visual Programming

  This set of new visual and non-visual parts is designed specifically for building Web applications. The new parts are in the Web connection category which is added to your Parts Palette on your composition editor. It is fully integrated.

- Construction from Component Technology

  Because Web Connection is feature of VisualAge for Smalltalk, you have the same ″construction-from-parts″ paradigm.

- Compatible w/Existing Non-Visual Parts & Class Libs

  With Web Connection, you can continue to use the existing non-visual parts to develop your Web application. All of the functionality that is provided with VisualAge for Smalltalk such as database access and communications can be used.

  Note to create your Web User Interface, you must use the new Web visual parts and cannot use the existing VisualAge for Smalltalk visual parts. This is because VisualAge must use the set of user interface parts that is supported by HTML, which is a smaller set than the standard VisualAge parts. These new parts represent the controls that can be displayed by a Web browser and these parts are fine-tuned to reflect true native HTML capabilities.

- HTML Generated by VisualAge

  The parts in the Web Connection category are special VisualAge parts that have the ability to translate themselves into HTML code. At runtime, instead of creating windows and user-interface controls on your screen, the parts generate HTML code that recreates your user interface in a Web browser. Web Connection generates HTML 2.0 compatible code.

- Integrated Development Tools

  Through VisualAge for Smalltalk, Web Connection provides a fully integrated development environment which includes the full IBM Smalltalk language, class libraries, browsers and debuggers that you can use to build your Web application,

- Object Oriented Techniques

  By using Web Connection, you can still gain all the benefits of OO as you are within the VisualAge for Smalltalk Development environment.

Web Connection: Application Flow Scenario

This chart describes the application flow of a Web application. The detailed process is listed below:

A single request from a Web client, and its results, follow a multi-step process:

1. The client Web browser submits a HTTP request that specifies your VisualAge Web connection application.

2. The Web server (a daemon running on the specified host) receives the HTTP request from the client. The server parses the HTTP request and runs the specified program (CGI Link), passing it the additional information contained in the URL.

3. The CGI Link program runs. CGI Link passes the client request to the running VisualAge image, communicating through a TCP/IP socket. The VisualAge image can be on the same machine as CGI Link or on a different machine.

4. The CGI Link Server, a process running in the VisualAge image, receives the request from CGI Link and creates a new instance of the specific VisualAge HTML part.

   CGI Link also parses any other data fields from the request so that they are available to the new part. Each HTML part can have a nonvisual subpart called HTML Form Data, which contains the parsed data from the request. By connecting to the attributes of the HTML Form Data part, you can access any of the data from the fields in the form.

5. The part generates HTML code, which is relayed back through CGI Link and the HTTP server to the client browser, which renders the page.

VisualAge for Smalltalk: Connection for Lotus Notes

1. New Add-On Feature of VisualAge for Smalltalk, V3.0

   The Connection for Lotus notes is a new add-on feature of VisualAge for Smalltalk V3.0. It was announced on June 18 and is generally available July 25. The Connection for Lotus Notes runs with VisualAge for Smalltalk Professional for WIN 95/NT and OS/2. AIX, HP, and SUN are forthcoming in future plans.

2. Builds Appls that Exploits Capabilities Provided by Lotus Notes

   The Connection for Lotus Notes enables you to develop applications that can directly access and modify Lotus Notes databases. Developers can use the power of the VisualAge for Smalltalk environment to build large-scale applications based on Notes.

3. Set of Smalltalk Classes and Non-Visual Parts

   Several new IBM Smalltalk classes and associated nonvisual parts are provided with the Connection for Lotus Notes. These parts will allow application developer to use VisualAge for Smalltalk to build applications that provide the following types of function for local and remote Lotus Notes databases. You can also continue to use the existing visual and nonvisual parts from VisualAge for Smalltalk.

4. Requires Lotus Notes R4.0

   This feature requires Lotus Notes-Release 4.x(32-bit client).

Connection for Lotus Notes: Classes and Non-Visual Parts

These new classes and parts are fairly self-explanatory. The key pints you want to make is that you can use these classes to build applications that can:

- Create new databases from templates
- Replicate databases
- Modify security rights in databases
- Open, modify, or delete existing databases
- Create new documents, or delete existing documents
- Add, delete, and update fields in a document
- Do a full text search in a database
- Read, write, and send Notes Mail

A key point in View/Forms is that a developer needs to create Views and Forms within the Lotus Notes environment. The databases. views, and forms are built in the Lotus Notes application environment using Lotus Notes as you normally would. Once the Notes database has been constructed, the views, forms and data can be accessed from VisualAge for Smalltalk by using the nonvisual parts provided with the Connection for Notes feature.

Getting Started with VisualAge for Smalltalk

Education: We are pleased to have IBM- Object Technology University excellent educational offerings. Including classes, and mentoring programs where they will do intensive Smalltalk and VA education for 6 week periods. We are also pleased to have ObjectPeople and KSC participating in our education offerings. Both ObjectPeople and KSC are well respected in the Object community and have several years experience providing excellent educational offerings.

Consulting: We have started a Consulting Partners Program where we can offer certified consultants to assist on projects. Jerry Leeman (LEEMAN @ CARVM3) can assist with any consulting requests.

Support: Online support for all of our current and Beta code is available on both CompuServe and TalkLink. 60 day phone support is also offered with the product. VisualAge for Smalltalk also participates in IBM extended service offerings. Contact Chuck Tripp (TRIPP) @ CARY) for any specific support requests.

VisualAge for Smalltalk: Features for V3.0

This chart is to only be used for customers who are on VisualAge Version 1.0 or Version 2.0 to contrast the difference. This chart highlights the new features of Version 3.0 which was GA in 11/95.

Performance: Significant improvements have been made to the underlying Smalltalk technology to increase performance of the VisualAge for Smalltalk development environment and end-user applications. Machine code executables can now be created on Windows and OS/2 (Intel based machines) that results in faster executing end-user applications. Based on industry benchmarks, VisualAge for Smalltalk V3 has increased its performance up to 5 times over the previous version.

AIX Support: You can build truly portable applications that runs on Windows, OS/2, and AIX, and utilize the native resources of the platform (NO ECULATION)!! Applications built using VisualAge for Smalltalk make use of the native X/Motif widgets and can run on Windows or OS/2 with no code changes.

Enhanced User Interface: VisualAge for Smalltalk V3 contains extensive new usability features. These include a new ″How To″ help systems, a ″quick start″ window, a high-level Organized Browser, hover help, drag and drop tab ordering, and an improved system of menu construction.

Native Oracle Support: Support for ODBC is now part of the base product. Database for ORACLE provides access to ORACLE data through native APIs.

Enhanced Database Parts: New and improved database access parts have enhanced the capability to quickly develop database applications. New multi-row query and single row query parts improve the data access performance of your application. Also, developers can now build applications that have multiple concurrent database connections. Additionally, support for DB2 has been extended to include binary large objects (BLOBs), user-defined types, and user-defined functions.

CICS Common Client Support:

IMS Transaction Wrappering: The VisualAge for Smalltalk, IMS Connection for OS/2, Aix, and Windows allows you to transparently integrate traditional online transaction programs and data into object-oriented applications developed with VisualAge for Smalltalk. This means you can use your traditional code on the host untouched, while using object-oriented technologies at the workstation.

MQ Series Support:

Object-Oriented Report Writer: The VisualAge for Smalltalk, Reports for OS/2 and Windows enables you to create reports on any object that can be used in the VisualAge for Smalltalk for Smalltalk environment, including parts provided through the VisualAge for Smalltalk database support, Communications/ Transaction and AS/400 feature.

New Portable Controls:

Distributed Smalltalk Feature: Enterprises can optimize their application execution environment with IBM′s new VisualAge for Smalltalk Distributed. Applications can now be partitioned across a distributed environment to create true second-generation client/server applications. VisualAge for Smalltalk Distributed includes features such as remote debugging, distributed garbage collection, and program visualization, and is tightly integrated with the VisualAge for Smalltalk team programming environment.

VBX Support: Support for Visual Basic Custom Controls (VBX) has been added to the Windows version of VisualAge for Smalltalk. Developers can now access the large base of VBX functionality from their VisualAge for Smalltalk application.

VisualAge for Smalltalk: Hardware Requirements

Use this chart as a backup to describe the development and runtime environment.

VisualAge for Smalltalk: List Prices

The next two charts describe the product and the add-on features. There are two versions: one with a price and one without. It is recommended to use the chart WITHOUT the price and just talk about the base product and the features without the context of price.

BACKUP CHARTS

The remainder of the charts are for backup.

- Application Development Productivity

  Use this chart to describe productivity improvements. This data is from The Prudential group and is their assessment based on a Smalltalk prototype they did.

- Customer/Consultant Quotes

  The next series of charts are quotes from customers and consultants on VisualAge for Smalltalk. The last chart lists awards.

+++EDF011W XMP end-tag found outside XMP and ignored. (Page 1 File: VAOVR96A SC
RIPT)
DSMMOM397I '.EDFEXMP' WAS IMBEDDED AT LINE 42 OF 'VAOVR96A'
DSMBEG323I STARTING PASS 2 OF 2.
+++EDF011W XMP end-tag found outside XMP and ignored. (Page 1 File: VAOVR96A SC
RIPT)
DSMMOM397I '.EDFEXMP' WAS IMBEDDED AT LINE 42 OF 'VAOVR96A'