IBM Application Development

Petr Leština

IBM Czech Republic, Murmanská 4, 100 00 Praha 10, Česká Republika

Abstract
Corporate down sizing, changing regulations, increasing competition, expanding markets ... all put pressures on application developers than ever before. The role of information technology departments is evolving to enhance support for the distribution of IT resources and provide global access to information on heterogeneous systems and protocols. This must be accomplished while continuing to control complexity, reduce cost and provide optimal business systems support for competitive advantage.

1. STRATEGIC DIRECTIONS OF IBM APPLICATION DEVELOPMENT (1)

IBM believes that our leadership in the application development business will require a complete set of tools:

- business and information modeling tools to support business process re-engineering including workflow modeling and control, prototyping and management
- visual programming to improve productivity, support a multiplicity of application types, e.g. client, server, transaction,network, data-centric, document-centric, and scalable execution environments, e.g. workstation, server, client/server, using construction from parts capabilities to enhance the traditional approach of building applications using programming languages, i.e. a compiler or interpreter for source code programs
- an object repository to manage and store the assets of development, i.e. models, source code, visual objects, test cases, execution control, documentation, etc.
- and software configuration management and build facilities such as version control, configuration management, problem tracking and change control, distributed and parallel builds and distribution services

IBM's marketing approach to application development is to offer solutions for:
- modernizing legacy application
- automating business processes
- developing object oriented applications

These solutions are offered as functional combinations of software tools with complementary service offerings that address the phases of application creation and maintenance:
- Modeling and design
- Build or construction
- Development and asset management

IBM's complete set of application development offerings provide significant value in a number of ways. Higher levels of developer productivity are enabled through intuitive and visual tools for automating business processes, modernizing existing applications, and leveraging the advantages of new technology. They enhance productivity through support for visually building applications from pre-built parts by teams of developers with different roles and skill levels. Parts are provided to leverage middleware and infrastructure based on industry standards such as those identified in the Open Blueprint, and implemented across a broad spectrum of heterogeneous client and server systems.

2. PRODUCT LINES

2.1 Modeling and design tools

Modeling tools help describe a business problem and define an information technology solution. Design models must be a complete representation of the business process. Modeling tools must be able to update existing models to implement desired changes. Tools need to work together and share model information for different operations, i.e.:

- Business process re-engineering tools to assist with business process improvement and automation
- Information engineering tools to organize data and applications construction for more efficient development and maintenance of information systems
- Object modeling tools to develop new applications that use object technology to improve maintainability, flexibility, and implement applications directly from the modeling tool.

IBM Business Process Re-engineering tools

IBM Business Process Modeler (ProModeler), built on a time-tested IBM methodology, the Line of Visibility Engineering Methodology (LOVEM), graphically maps the way work moves through an organization. Process design teams can actually "see" a process while documenting, evaluating, and redesigning it. Processes are described in standard business terms, including process goals, critical success factors and quality measures. In this manner, teams highlight crucial points and identify ottlenecks.

Through integration with FlowMark(TM), professional developers can take the business model to the next step by creating, defining, documenting, simulating, and testing the supporting workflow processes. FlowMark is used to assign key company resources including people, programs and data to get the job done. FlowMark then manages and tracks execution of the process, i.e. through measurement and monitoring functions provided by FlowMark, companies can ensure that the processes fulfill their identified business goals, creating a closed loop for continuous process improvement.

The combination of business modeling and workflow management results in a complete set of visual process design tools from IBM. They provide simulation, control, execution, and measurement for re-engineering a business process. IBM modeling tools also support developing applications from process models.

IBM Information Engineering tools

Information engineering tools model database designs and the relationships of data elements and programs. An understanding of these relationships helps to more easily translate business changes into applications. Tools that keep track of the implementation of business rules and the use of information and data elements help reduce redundancy and promote reuse.

TeamConnection DataAtlas allows conceptual modeling of the use of information by different applications. It keeps track of data element and record definitions, suggest physical database designs and generates these designs for optimum database performance. DataAtlas also includes data dictionary functions that are available to all other application development tools.

DataAtlas, the business rules tool, and the build tools provide a complete development environment for many of the tasks associated with maintaining, modernizing, and extending most of today's enterprise applications. They can be used in an integrated way with IBM's business process re-engineering tools.

IBM Object Modeling tool

Movement to object oriented technology is growing in popularity. Many organizations find that this technology fits their business needs better. Critical to the successful use of object technology is a good design for the application. IBM is developing a new form of modeling that makes use of object oriented technology. The developer is able to model applications using business objects that represent business processes and messages and the tool produces an application based on the model.

2.2 Build tools

In order to successfully implement and manage application solutions, developers need build tools and development management capabilities. IBM is providing a comprehensive set of build tools to improve developer productivity. IBM build tools support three key application development principles:

- Visual construction from pre-built parts for build time productivity,
- A programming language of choice,
- and a platform of choice to leverage programming skills for scalable client and server applications,
- Evolutionary utilization of objects to mask the complexity of new technology and ease future application maintenance.

Visual Construction from pre-built parts

IBM's strategy is to enable the developer to focus on business issues while lowering the barriers (depth of skills) to developing applications. Developers will be able to build applications from pre-built components -- construction from parts. This will dramatically improve development productivity while lowering the required entry skill for developers. IBM offers this visual development environment across a broad set of existing programming languages.

The VisualAge family of products, VisualAge for Smalltalk, VisualAge for C++, and VisualAge for COBOL, and VisualGen(TM) provides power programming for a distributed environment. These tools use visual construction from predefined parts on a developers palette to more easily build applications without using a programming language. Palettes are provided that combine easy client graphical user interface (GUI) construction from parts with powerful parts that represent application function such as business logic, data, transaction and network access, communication protocols, and multimedia to quickly build an application. A developer can use tools that come with VisualAge to build queries to data bases without knowing the structured query language (SQL). The tools visually take the developer through the data access using information from the database and pre-built data access parts. VisualAge supports C++, COBOL, and Smalltalk applications for both responsive development and optimized object application execution.

VisualGen is a high productivity visual tool for three-tier (or one- or two-tier) transaction based client/server application generation. It is an object based tool that uses the power of visual construction for both client and server applications. VisualGen has the unique ability to develop client and server parts as a single unit, then generate the tested applications to several different execution environments without changing specifications. This generator will be enhanced to support Smalltalk client functions such as partitioning and the use of SOM and Smalltalk parts.

IBM is extending its 3GLs with the inclusion of new technology and visual development environments. For example, VisualAge for COBOL for OS/2« and IBM COBOL for MVS add object function to enable the many developers familiar with its specifications to use this next generation COBOL to add object technology and visual construction of client/server applications to their skills without leaving the popular commercial COBOL language. COBOL and PL/I are offered on MVS, OS/2 and AIX« with the same compiler technology. The visual tools for PC and workstation environments can be used to cooperatively develop applications for execution on other platforms such as MVS.

Ultimately, the visual builders of the VisualAge family will converge onto a single visual construction technology. This will enable parts libraries to be provided in a standard form, i.e. SOM/DSOM, that can be used with any programming language within the family.

The construction from parts paradigm can extend skills to new technology based upon the use of visual tools. In addition to the access to Notes mentioned above, parts to integrate VisualAge with workflow and access the Web, new tools based on technologies such as Java will extend the power of visual building to those application types as well. Java is a technology for developing and executing logic attached to a Web page. It has been licensed by IBM from SUN(TM) and is being ported to OS/2 and AIX. A VisualAge for Java tool will enable such logic to be developed using visual construction from parts.

A programming languages of choice and a platform of choice

IBM's strategy is to leverage existing skills, code and systems by providing the widest choice of languages across the widest choice of execution environments. IBM will provide its key compilers for IBM systems -- S/390«, OS/400«, AIX, and OS/2 and non-IBM environments -- Windows (3.1, 95, and NT(TM)), HP(TM) and SUN. This will enable developer skills to apply to scalable client and server applications for a broad range of computing capacity.

For example, the languages of VisualAge are being offered on many platforms today (OS/2, Windows, and AIX) and the list is growing (C++ was made available on MVS, OS/400, and SUN, Smalltalk for AIX, PL/I for AIX, and COBOL for OS/2 and AIX, all in 1995). This enables both client and server applications to be developed in the same language with the same development environment. C++ and Smalltalk will lead the trend to multiple platforms. They were introduced for Windows 95 and NT.

VisualGen includes a 4GL language for developer productivity. This allows it to generate the same application to different platforms without changing application specifications. VisualGen also generates character based terminal applications, enabling developers' skills to easily evolve to building both these host-based applications and client/server applications using the same tool. VisualGen supports data access across several data structures, a variety of middleware, clients on OS/2 and Windows and servers on S/390, OS/400, VSE, AIX, OS/2 and 32-bit Windows platform support.

Traditional host 3GLs are being offered on additional platforms. Both COBOL and PL/I come with a visual development environment on OS/2 and AIX. COBOL for 32-bit Windows and UNIX will soon extend the range of COBOL skills to applications on those platforms as well.

IBM Open Class(TM), introduced with VisualAge for C++, and IBM Smalltalk class libraries enable a single development environment to provide portability to multiple execution platforms without changes to the application itself. They enable your organization or industry software suppliers to build upon these classes with across-industry and industry specific classes and frameworks that provide additional improvements in developer productivity and in application development responsiveness to business' needs.

The use of class libraries masks the complexity of underlying systems infrastructures and provides pre-built and pre-tested functions for building applications and running them on multiple systems. For example, with the introduction of VisualAge for C++ for Windows, IBM integrated Taligent's Compound Document Framework (CDF) into IBM Open Class. This simplifies OLE programming by enabling developers to create OLE applications with as few as two or three lines of code. Additional frameworks from Taligent, such as 2D graphics and print/view, will be integrated into IBM Open Class over time.

Evolutionary utilization of objects

For those who recognize that object technology may be the best way to develop distributed applications, IBM offers visual construction from pre-built parts in C/C++ and COBOL that support both procedural and object application development. This enables the migration of skills to the best technology for solving business problems especially in conjunction with visual construction that can mask the underlying technology from the developer.

VisualAge for Smalltalk supports new object oriented applications using visual construction that can reuse many existing resources, including access to databases, transactions, and communication protocols, support distributed applications, and wrappers for VisualBasic(TM) controls (Windows only) and PC applications written in C or COBOL. This enables existing applications to be extended with new components that use object technology enabled on a wide choice of platforms.

The latter is part of our systems strategy to allow application components written in different programming languages to work together. This can be accomplished through the use of IBM's System Object Model (SOM). These applications consist of components that can be distributed across multiple servers and clients using Distributed SOM (DSOM). Our object strategy is to include a set of IBM pre-built parts with IBM tools, e.g. IBM Open Class and IBM Smalltalk class library. IBM is also working with industry suppliers through programs such as Object Connection(TM) to build parts in an open architecture that enable those parts to be used and reused in application solutions built with the VisualAge family.

VisualAge for C++ and VisualAge for COBOL and ultimately IBM BASIC support direct-to-SOM compilation to provide objects for management by each operating system that supports SOM/DSOM, i.e. OS/2, Windows, AIX/6000«, OS/400, and MVS as well as those that conform to the Object Management Group (OMG) Common Object Request Broker Architecture (CORBA). In addition to these direct-to-SOM compilers, build tools such as VisualAge for Smalltalk can reuse SOM objects as part of the application even though it uses a different object model (i.e. Smalltalk) for execution. VisualAge for Smalltalk includes SOM and DSOM Client support.

Today, class libraries are provided with tools like VisualAge for C++ and VisualAge for Smalltalk. Functions include collection classes of basic application building blocks, user interface classes, communications classes, data access classes, and multi-media classes to be reused as a part of the application being built.

IBM Open Class and IBM Smalltalk class libraries represent IBM's strategy to promote productivity from reuse of code and design. They contain a growing number of classes and frameworks that will simplify access to resources and the development of applications. This is done by abstraction of middleware interfaces and other functions (selecting parts is much simpler than coding procedural APIs) to protect the developer from the complexity of client/server environments and to enable rapid development of functionally rich client and server applications that access resources such as database, transaction, Web and Notes in a networked environment.

2.3 Development and Asset Management tools

TeamConnection, groupware for developers, integrates software configuration management functions and object-oriented repository services. It provides a model for tool integration to support roles-based team development in a client/server environment. TeamConnection supports the development of host based applications and client/server workstation and distributed applications. The developer has the choice of using either a procedural or an object development paradigm. Source, object and model assets will be stored, updated and managed for distribution to a variety of developers and for a variety of systems infrastructures. Other development resources such as JCL, README files, and documentation can be managed consistently with modeling and workflow information and the executable application assets.

TeamConnection integrates problem tracking and change control to ensure that application developers are more productive and project leaders can effectively manage and track the development process. TeamConnection automates and streamlines the application build process and integrates it with version and change management. The build function is tied to release management and extended to provide a framework for delivery to clients and servers using NetView(TM) Distribution Manager.

An open, extensible information model provides the vehicle for data sharing among the modeling and design tools and the build tools like VisualGen, VisualAge for C++, VisualAge for COBOL, and PL/I. The information model ensures continued support for new versions of existing tools and the integration of new tools with those that already exist. It has an extensible architecture and an object oriented information model that is an integral part of TeamConnection. Its repository services include constraint checking, version management, concurrent and distributed access, and data exchange for information model assets produced by several different modeling tools.

TeamConnection's LAN based implementation allows developers to browse and reuse development assets with great speed. Host applications could be developed by checking out an entire program from a host based library and little additional control was needed. The iterative object development technique using modular parts to assemble complete applications is significantly different from developing those traditional applications. A LAN based implementation of asset management supports browsing different development objects and even different versions of the same object stored in a the repository. To enable easy migration to new techniques and coexistence of both object and procedural development, TeamConnection supports storage, retrieval and consistent management of all data elements, design, model, code parts, and objects in the same LAN based client/server development environment.

References

[1] IBM Application Development White Paper, November 1996