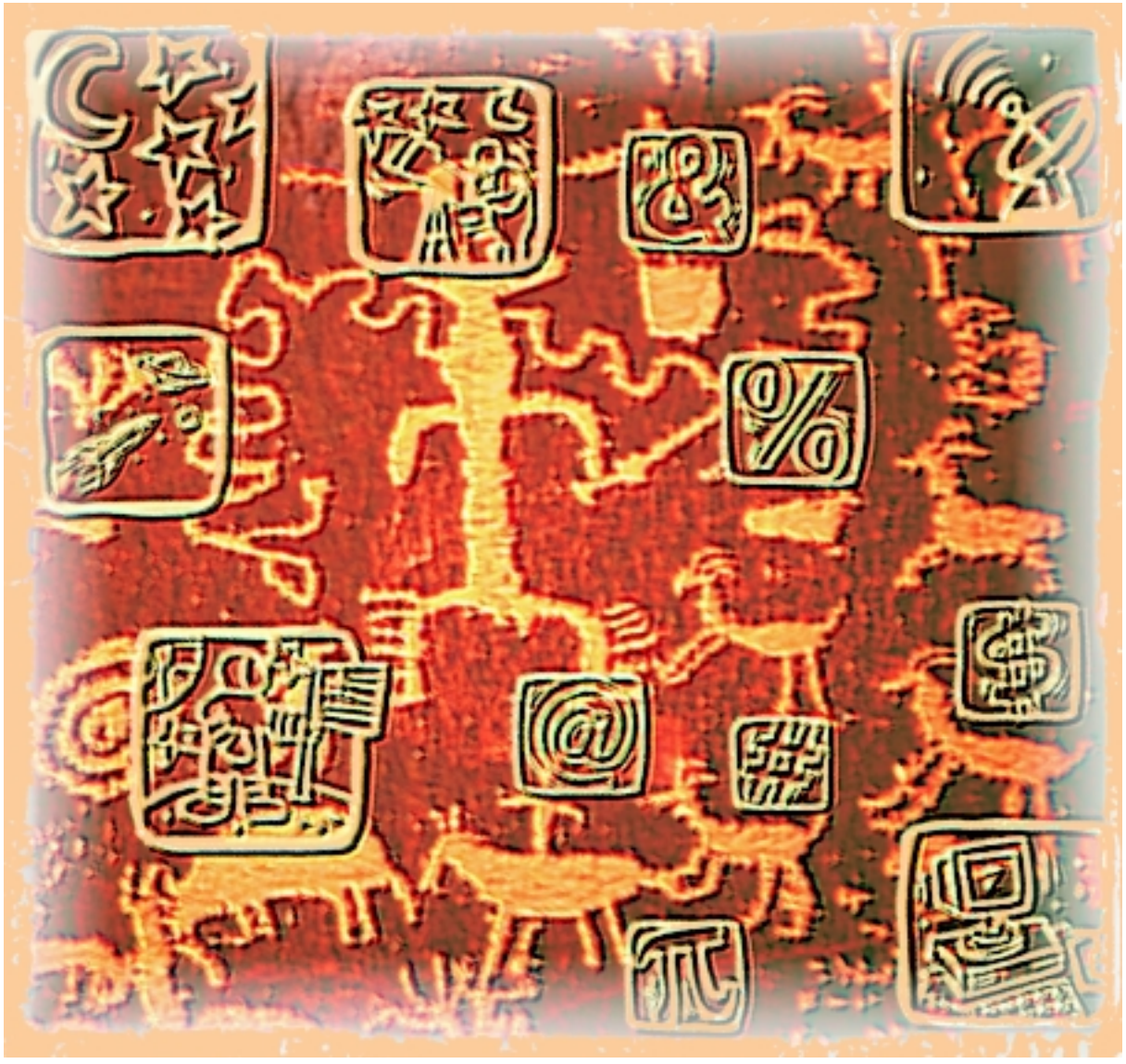
2000

VSE/ESA™ Software Newsletter

News and information for VSE/ESA customers and vendors



G225-4508-20

Trademarks

The following are trademarks or registered trademarks of International Business Machines Corporation in the United States of America and/or other countries: AIX, AS/400, CICS, CICS Transaction Server for VSE/ESA, CICS/VSE, DB2, DRDA, Enterprise Storage Server, IBM, MQSeries, Multiprise, Netfinity, OS/2, OS/390, OS/400, Parallel Enterprise Server, RS/6000, S/370, S/390, SecureWay, TCP/IP for VSE/ESA, VisualAge, VM/ESA, VSE, VSE/ESA, WebSphere

Domino Go Webserver, Lotus, and Lotus Domino are trademarks of the Lotus Development Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft Windows, Windows NT, and the Windows Logo are trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

Other company, product or service names may be the trademarks or service marks of others.

About This Newsletter

The *VSE/ESA Software Newsletter* is published twice a year as a service to users of VSE systems. On the Internet, the *Newsletter* is available in PDF format (Adobe Acrobat Reader) at:
http://www.ibm.com/s390/vse/vsehtmls/newslett.htm

The information contained in this document is based on the best knowledge of the authors and on current development, has not been submitted to any formal IBM® test, and is distributed on an "As Is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead. Any performance data contained in this document was determined in a controlled environment and, therefore, the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country or not yet announced by IBM. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services.

Editor: Joel Hermann, IBM Entwicklung GmbH, Department 3257, Schönaicher Str. 220, 71032 Böblingen, Germany (jhermann@de.ibm.com)
Subscriptions: See the subscription form at the end of the *Newsletter*.

Contents

Editor's Notes

Address Corrections / Subscription Changes

If you are subscribed to the *VSE/ESA Software Newsletter* and would like to make address changes or other corrections via e-mail, simply send a note to Margarete Büttner (**buettnem@de.ibm.com**) with that information. Thanks!

News That's "Fit to Print"

Welcome to Issue 20 of the *VSE/ESA Newsletter*! As always, I want to express my thanks to all contributors.

If you'd like to submit an article, please let me know. The rules for publishing something in the *Newsletter* are simple:

- Contact me using my e-mail address, and describe the topic that you'd like to address.
- The Editorial Board will consider the article's benefit to our readers and then either accept or reject it for the *Newsletter*.
- What should articles cover? I think the answer is: “That's up to you.” We're as open and unbiased as possible.
- Naturally, the accuracy of articles you submit is your responsibility, not IBM's. But that's business as usual.

Joel Hermann (jhermann@de.ibm.com)

Related Web Resources

VSE/ESA Home Page

Besides the *VSE/ESA Newsletter*, VSE Development also supports a VSE/ESA home page. The site has a wide range of information about VSE and VSE-related products from IBM and other vendors. Check out: **http://www.ibm.com/s390/vse/**

VM/ESA® Home Page

VM Development has an excellent home page with a wealth of information. This site is a must for VM users and is at: **http://www.ibm.com/s390/vm/**

VSE-L Discussion Forum

VSE-L is an Internet community where VSE users gather to share tips, experiences, and support. To subscribe to this forum, which is sponsored by Lehigh University, send an e-mail to: **listserv@lehigh.edu**

In the first line of the message body, enter:
SUBSCRIBE VSE-L *your full name, your company name*

Articles Related to VSE/ESA

VSE Applications – How e-business Fits

Authors

Anette Stolvoort
VSE Technical Marketing and Sales Support
ahaller@de.ibm.com
G. M. (Jerry) Johnston
Senior Advisor, IBM Böblingen Laboratory
p798000@us.ibm.com

Editor's Note

This article is a reprint of the IBM white paper, GF22-5137, which also is available on the Web at:
http://www.ibm.com/s390/ftp/vse/docs/vsepos.pdf

Overview

VSE customers have a wider range of options than ever before.

This paper examines application choices for VSE customers. The focus is on how IBM's Application Framework for e-business helps protect their investment in existing information assets while enabling them to exploit emerging e-business opportunities.

Because of affinity, OS/390® is a traditional growth path for many VSE customers. It offers state-of-the-art e-business capabilities, along with advanced CICS® function, applications (including popular ERP applications), and superior systems management, capacity, and availability.

For other VSE customers, IBM's Application Framework for e-business can be a guide to extend CICS-based applications to the Web and combine S/390® with non-S/390 servers (i.e. RS/6000®, Netfinity®, or AS/400®) for state-of-the-art e-business solutions.

The recent preview of VSE/ESA Version 2 Release 5 shows how VSE plans on interoperability as a means of helping customers exploit the power and the state-of-the-art capabilities of the IBM Application Framework for e-business. (See the article, “VSE/ESA Version 2 Release 5 to Provide Improved Interoperability” on page 23.)

Priorities

Two significant trends are converging and there are pivotal implications for VSE customers.

First, companies have been distracted by the Year 2000 and euro challenges and haven't been able to make many of the investments needed to update existing applications or add new ones.

Second, the Internet and e-business are maturing. Most of the technology is now in place or well along. For many customers, it's largely a matter of setting a sound business strategy, then moving forward to implement the strategy.

The combination of newly available resources and critical business needs may be setting the stage for a dramatic surge in application solutions beginning in 2000.

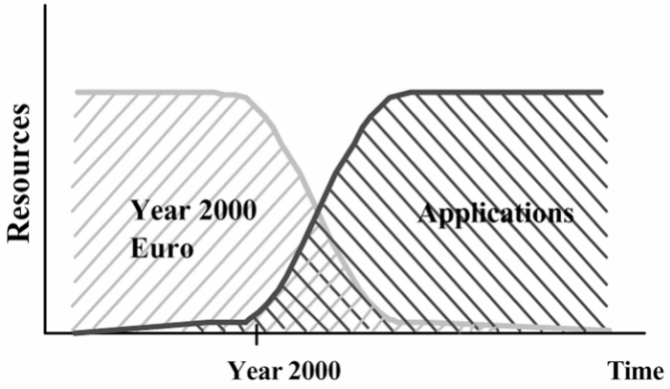


Figure 1. Priorities

Applications

VSE customers have three types of applications in their portfolio. Some are in production, some in transition, and others on the waiting list. These definitions form the basis for the discussions in this paper:

- Core
- Replacement
- New

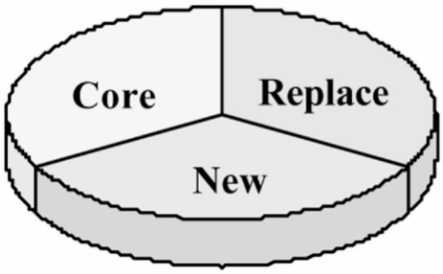


Figure 2. Applications. Pie arbitrarily divided into thirds. Percent of applications in each category varies widely among individual customers.

1. **Core Definition**

   A "core" application is one that is expected to remain in production more or less indefinitely. It does what is expected of it and there is no overwhelming business need to replace it within the foreseeable future.

   Core does *not* mean "frozen." Core applications may be updated due to changes in business processes or to support new regulations. They may be modified to accommodate new reporting requirements or database changes. Core applications are sometimes extended and enriched by complementary new applications. Thus they may need to be linked to related applications at the same or another location.

   In summary, if you'd rather keep it than throw it out, it's a "core" application.

   Q and A

   Q. Aren't core applications obsolete?

   A. Core applications aren't static. Many will provide the IT backbone for a long time to come. Any application will eventually become obsolete, but for now core applications are those that (with some ongoing effort) are expected to remain relevant within a reasonable planning period.

2. **Replacement Definition**

   "Replacement" applications totally supersede existing VSE applications.

   An application may be replaced for one or more reasons, such as:

   - It no longer supports business needs and cannot be modified or extended at a reasonable cost.
   - After a merger or acquisition, previously unrelated IT systems may be standardized for consistency.
   - It was purchased from an Independent Software Vendor (ISV) who no longer offers even basic enhancements or error correction.
   - Senior management may want to establish a new strategic direction.

   In summary, if you've already got an application but want a newer one, the new one meets this definition of "replacement."

3. **New Definition**

   "New" applications are those implemented for the first time. That is, there is no existing, precursor VSE application like it running now on the customer's system. New applications may be, but aren't necessarily, "leading edge." They're just new to this customer.

   In summary, if you do not already have an application like this one, then it's "new" to you.

Core Applications

Before we begin with e-business, let's first take a moment to consider the options for core applications. Later we'll discuss ways to extend and enhance core applications using Web technology.

At the risk of oversimplification, most VSE core applications:

- Are CICS-based (more than 95% of VSE customers use CICS)
- Are written in COBOL (70-80% write mostly in COBOL)
- Access VSAM files (Even when DB2®, DL/I, etc. are used, lots of data remains in simple VSAM files.)
- Interface to end-users via 3270-type "green screens" using SNA

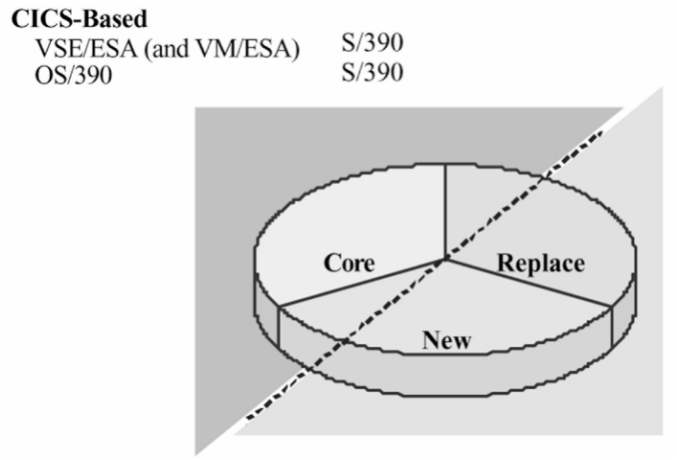


Figure 3. Core Applications

CICS sets the standard for industrial-strength "Online Transaction Processing" (OLTP). CICS processes 20 *billion* core business transactions all day, every day, in every industry, in every part of the world. VSE's basic OLTP technology was recently updated with CICS Transaction Server for VSE/ESA™ Release 1.

Of course, core applications may be batch-oriented (also mostly COBOL and VSAM) as well.

Core applications frequently represent massive investments in Information Technology (IT) systems (including hardware), IT skills, end user training, and business procedures and processes. Major changes may be costly, disruptive, and risky. Replacing fully functional applications isn't something most companies undertake lightly without sound financial or strategic justification. They find it's often more cost-effective to modify and adapt existing core applications.

Migrating Core Applications to OS/390

What if core applications meet the needs of the business, but VSE itself doesn't quite measure up for whatever reason? For some VSE customers, OS/390 may be the preferred option.

Historically, OS/390 has been a primary growth path for VSE customers. For many of them it still is. Because of affinity between VSE/ESA V2.4 and OS/390, core CICS-based applications can be adapted to OS/390 *relatively* easily (compared with conversion to platforms such as UNIX, Windows NT, or AS/400). In addition, many VSE skills apply to OS/390 as well.

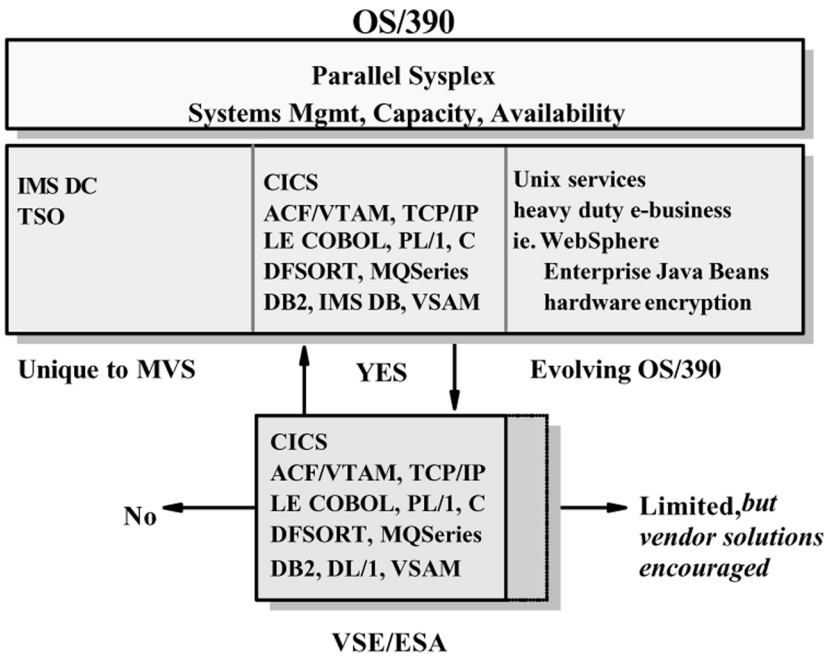


Figure 4. Scope of Affinity

By running core applications largely unchanged *from an end user perspective*, OS/390 avoids many of the indirect costs (end user training, new processes and procedures, etc.) of converting to other platforms.

It's a good idea for every VSE customer to review their situation and consider if OS/390 is appropriate for them within a 2-5 year time horizon.

Several Things to Consider

Several factors influence a decision to migrate to OS/390. Here are just three:

1. **Rate-of-change**: VSE/ESA averaged about one new release per year since 1990. Nevertheless, the rate-of-change of OS/390 is far greater. It's reasonable to expect the gap will grow. That's because much of the expansion of VSE capacity and OS/390 affinity is largely complete. For example, the Turbo Dispatcher exploits up to about a 3 or 4-way system today; and there is no expectation it will expand beyond that limit. In addition, internal VSE changes needed to support CICS Transaction Server for VSE/ESA are mostly done. Future focus will be on incremental enhancements, mainly hardware and connectivity improvements.
2. **Availability of new applications**: OS/390 UNIX System Services bring new applications (i.e.. SAP R/3, PeopleSoft, etc.) to S/390. Even for CICS-based applications, ISVs often offer more choices for OS/390. In addition, OS/390 is especially well positioned to handle new e-business applications.
3. **Systems management**: Customers sometimes find that, although VSE has the capacity needed, overall center management becomes an issue as the number of applications, database size, and transaction volumes grow. OS/390 has a rich set of IBM systems management tools to help reduce the overall cost-of-computing.

Q and A

Q. Does this mean IBM is trying to force VSE users to go to OS/390?

A. Not at all. Thoughtful planning is in everyone's interest, and OS/390 is clearly one option VSE users should consider. As always, the individual customer is the best judge of what's right for their specific situation.

Replacement / New Applications

After thoughtful consideration, many customers may decide that VSE continues to be the best platform for their core applications. That's a valid choice. However, they're likely to realize their total IT needs can't be satisfied by core applications alone. Changing business requirements may lead them to seek replacement and/or new applications as well.

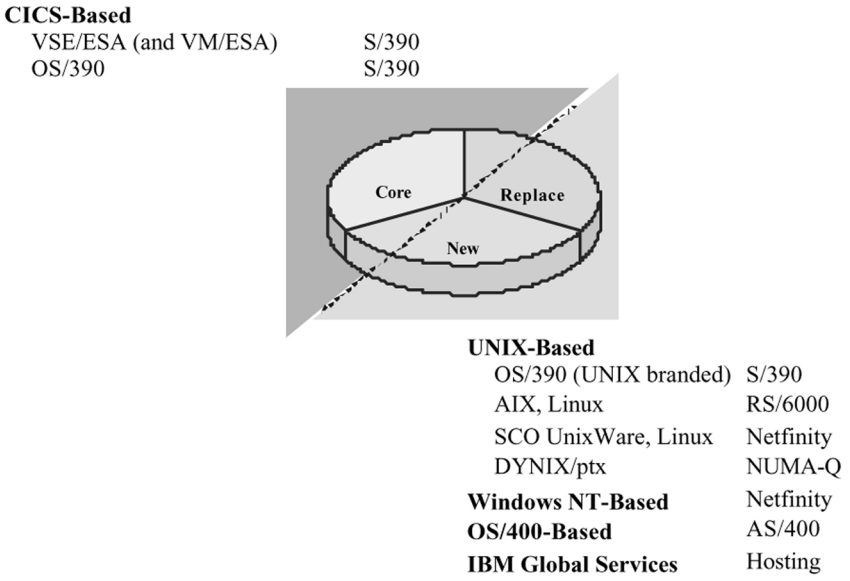


Figure 5. Replacement / New Applications

Replacement or new applications may be developed by in-house staff. Alternatively, off-the-shelf packages may offer a better combination of fast time-to-market, low total cost, best industry practice, and state-of-the-art IT technology.

When CICS-trained resources are available, or competitive packages are offered by reputable ISVs, CICS-based applications are excellent candidates for many replacement and new applications. CICS has proven itself to be a reliable, scalable, and high performance transaction server. It often yields the lowest cost per transaction.

In practice, executive line management may select applications without regard for the platform. Since the selection of packages on non-S/390 platforms (including UNIX, Windows NT, or OS/400®) is more extensive, sooner or later most VSE customers will inevitably install non-S/390 applications for at least a part of their replacement or new application needs.

Although there's no typical profile, today's non-S/390 applications are often:

1. Based on a loosely defined "client/server" architecture using platform-specific interfaces and APIs.
2. Written in C and C++. (AS/400 applications may be written in RPG.)
3. Access relational data.
4. Interface to end-users using Windows-style "GUI" display and navigation standards.

It's now recognized that client/server projects sometimes failed to meet their objectives. The e-business model addresses many of the problems with client/server.

Again, at the risk of oversimplification, many e-business applications have these in common:

1. They're based on standards such as TCP/IP, HTML, XML, SSL, etc.
2. They include both server and client code written in Java and created using powerful visual development tools.
3. They access relational data locally or remotely.
4. They interface with end-users through a standard Web browser.

When an application is selected that is based on a non-S/390 platform, IBM offers these outstanding server options.

1. For UNIX-based applications:
   - OS/390 (UNIX services) on S/390,
   - AIX® on RS/6000,
   - SCO UNIXWare on Netfinity,
   - Linux (now or coming soon) on selected models of RS/6000 and Netfinity,
   - DYNIX/ptx on NUMA-Q (formerly Sequent).
2. For Windows NT-based applications:
   - Netfinity servers.
3. For OS/400-based applications:
   - AS/400 servers
4. IBM Global Services is a possibility with offerings such as Web hosting, application hosting, out-sourcing, etc.

For more information on IBM servers, please visit the home page at:
http://www.ibm.com/servers/

Q and A

Q. Maintaining multiple servers sounds complicated. As long as any VSE application is being replaced, doesn't it make sense to convert the entire VSE system to the server platform chosen?

A. When the applications replaced are a small part of the VSE system, conversion may be a waste of scarce resources. Converting simply to get to a single platform may be a futile, never-ending quest. OS/390 is the exception. It supports nearly all core applications, most new applications (especially e-business), and many of the most attractive replacement applications. Nevertheless, for some VSE customers conversion *may* be the best choice. Here are a few considerations

- If the decision is made to replace a big, pervasive, highly integrated application (examples may include SAP R/2, COPICS, or banking) that accounts for most of the VSE workload, it may be reasonable to consider eliminating VSE entirely.
- Some customers have lost even the most basic VSE skills and are unable to keep their systems and core applications up-to-date. It may be reasonable for some of these smaller, "back level" users to consider starting over and replace their VSE system with new packaged applications.

Connectors

Most VSE users already have non-S/390 systems installed. In one recent survey, more than 96% of VSE and VM/VSE customers already had one or more non-S/390 platforms. In fact, they had an average of 3.5 different types of platforms (ie. Windows NT, several UNIX varieties including AIX, Novell, AS/400, OS/2®, etc.).

However, in many cases they're not connected. Even when they are "connected," periodic file transfer is sometimes the highest level of "data sharing" used.

As customers begin to exploit e-business opportunities, "real time" access to current VSE production applications and data may be needed. Powerful, flexible, easy-to-use "connectors" are a key requirement.

VSE/ESA Version 2 Release 5 e-business connectors add to the extensive set of connectors that already includes MQSeries®, DB2 Connect, CICS Transaction Gateway, and other key middleware from IBM and Independent Software Vendors.

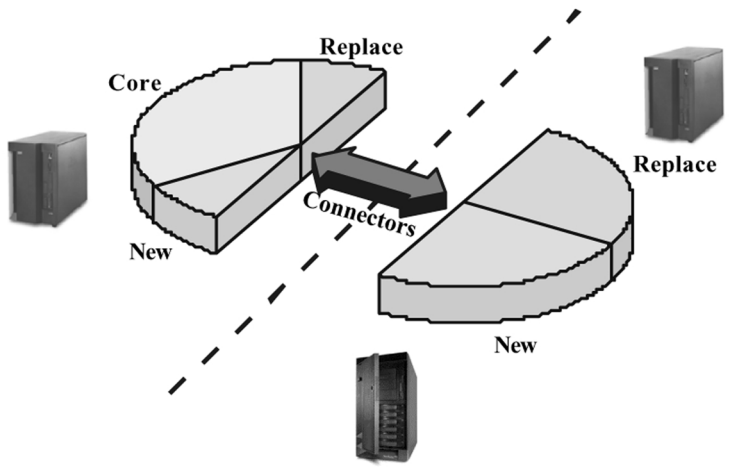


Figure 6. Mixed Server Environment

Whenever the IT infrastructure may consist of several kinds of servers, it's especially important to implement the connections based on industry standard APIs and protocols, plus reliable, productive IBM middleware. That's where IBM's Application Framework for e-business comes in.

IBM Application Framework for e-business

It's not the purpose of this paper to present the case for the Internet and e-business. Plenty of articles, speeches, and consultant reports do an outstanding job of laying out the grand vision. This paper will simply focus on how VSE fits within the e-business model.

In summary, the framework consists of three logical layers. There's a client (a PC with a standard Web browser, or perhaps even a wireless phone or PDA), a Web application server, and connectors to external services if needed.

1. The client contains logic related to the presentation of information (i.e.. the user interface).
2. The server is the hub that processes requests from clients, orchestrates access to business logic and data (using "connectors" if the logic or data are on a different system), integrates static and dynamic content, then returns Web pages to clients.
3. Connectors provide access to external services such as business logic and data.

Elements of IBM's Application Framework for e-business include WebSphere™ Application Server, Lotus, VisualAge® for Java, DB2 Universal Database, MQSeries, etc. For a more complete description of the IBM Application Framework for e-business please visit the home page at:

http://www.ibm.com/software/ebusiness/library.html

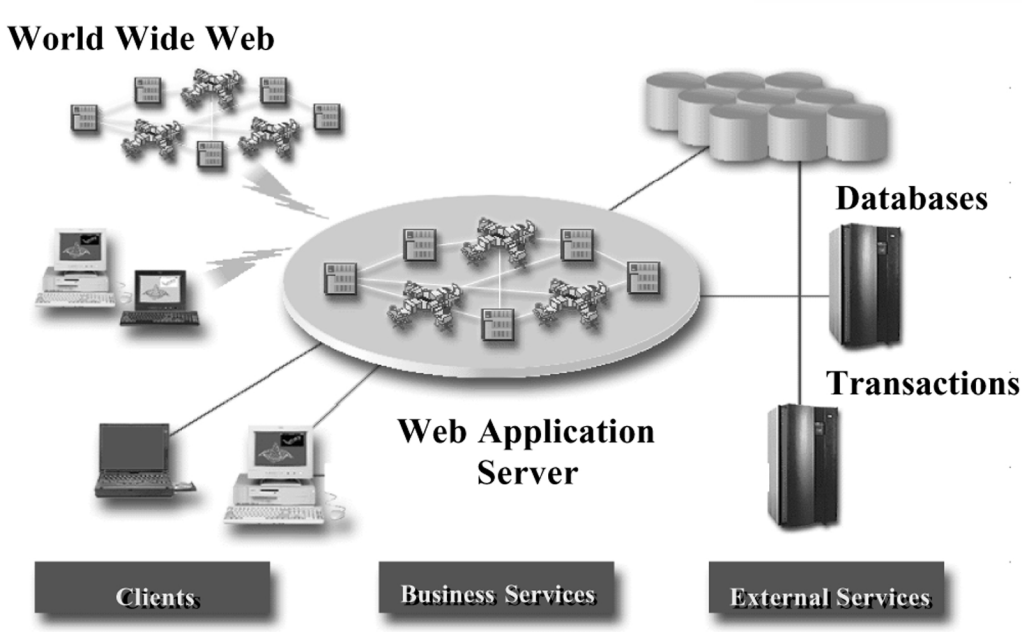


Figure 7. IBM Application Framework for e-business

Q and A

Q. What does WebSphere provide that I can't get with other Web Servers?

A. A basic Web server (really an HTTP server) coordinates, collects, and assembles Web pages and delivers them to the client. In fact, an HTTP server is a prerequisite for WebSphere Application Server. On some servers, it comes with an Apache HTTP server, although it works with others.

In addition, WebSphere Application Server provides a robust runtime environment and embraces industry standards such as Enterprise JavaBeans (EJB), eXtensible Markup Language (XML), Common Object Request Broker Architecture (CORBA), and more. On some servers, WebSphere comes in Standard, Advanced, and Enterprise Editions. WebSphere Enterprise Edition is a standard part of OS/390.

The WebSphere Studio (powerful application development tools and facilities) and WebSphere Performance Pack are complementary products. Tivoli TME-10 can be used for network management.

Q. What about security issues?

A. The Application Framework for e-business supports state-of-the-art security services designed to protect assets from unauthorized access from internal and external sources. For example, a "firewall" is a basic element of Internet security. IBM offers SecureWay® firewall solutions (including education and services) based on a choice of RS/6000, Netfinity, or AS/400 servers.

VSE and the Application Framework for e-business

In the IBM Application Framework for e-business, the application server may be WebSphere. For VSE, that implies a "3-tier" physical model since WebSphere does not run "natively" on VSE.

WebSphere is available (or planned) on OS/390, AIX, Windows NT, OS/400, plus a variety of platforms including Linux, Sun Solaris, HP-UX, OS/2, and Novell. *There is no plan to support WebSphere on either VSE or VM.*

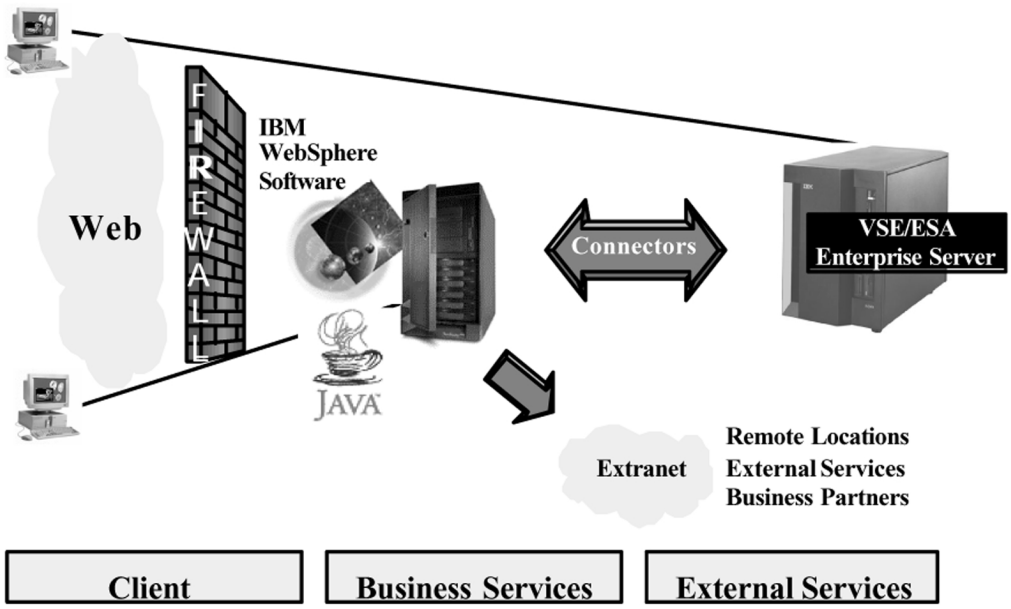


Figure 8. VSE and the Application Framework for e-business

In the 3-tier model, a WebSphere application server running on RS/6000 or Netfinity (for example) is the central "hub." Clients (tier 1) use standard Web Browsers to interact with a WebSphere-based application (tier 2). When required, WebSphere initiates an application or requests data from a VSE system (tier 3) using standard connectors. Upon completion, WebSphere packages any VSE information together with other static or dynamic components and sends a reply back to the client.

In the 2-tier model, clients (different clients, or perhaps the same clients using a different application) use standard Web Browsers to interact directly with CICS or a VSE-based HTTP Server (tier 2). When the work is complete, replies go from VSE directly back to the client.

VSE supports 2-tier as well as 3-tier implementations.

Q and A

Q. Does implementing e-business mean I must scrap my VSE investment and start from scratch?

A. On the contrary. A major objective of IBM's Application Framework for e-business is leveraging existing information assets (including existing VSE applications and data). Using the framework as your guide helps ensure you'll be able to accelerate time-to-market and reduce the overall cost-of-ownership.

Access to Applications

Many customers will find a 3-tier model is the best way to seamlessly integrate VSE core applications and data with replacement and/or new applications that run on OS/390 or a non-S/390 platform.

The 3-tier model based on IBM WebSphere Application Server, Java, DB2 Universal Database, etc. supports industry standards and offers state-of-the-art technology. Using WebSphere, VSE users can implement heavy-duty, world class e-business applications.

A 2-tier model can be a cost-effective way to protect an investment in CICS-based core applications, data, and skills.

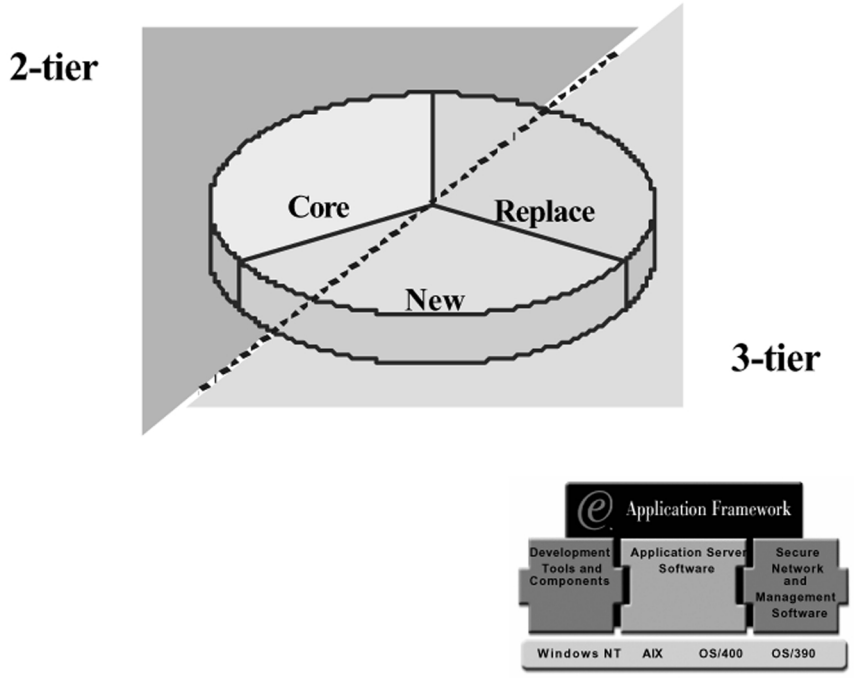


Figure 9. Access to Applications

For example, a VSE user may implement a new UNIX-based customer service application on an RS/6000 server using WebSphere. The application might allow regular customers to enter orders on their own over the Internet, exploiting appealing graphical user interfaces with the appropriate security layers.

The new application may also require customer credit information that's available on the VSE system as a by-product of core accounts payable (AP) and accounts receivable (AR) applications. Since the request for VSE data comes from the WebSphere application server, not directly from the client Web browser, this illustrates the 3-tier model.

Meanwhile, CICS-based core AP and AR applications continue to be used internally by trained employees. It may be desirable to update those applications by simply allowing access from an intranet using PCs (or network computers, etc.) and standard Web browsers. Enabling direct Web access to CICS applications illustrates the 2-tier model.

A variety of products available from IBM and ISVs allow access using a familiar browser interface. The transformation can occur more or less "automatically" without requiring extensive changes to the core application.

In the end, this example may be typical. That is, customers will exploit both 3-tier and 2-tier models as appropriate.

3-tier Implementations

A key to successful 3-tier implementations are the connectors between the WebSphere application server and VSE. Connectors of interest include **(a)** access to relational data stored in DB2 for VSE (and VM) using the Distributed Relational Database Architecture (DRDA®), or **(b)** access to CICS-based applications using MQSeries or CICS Transaction Gateway.

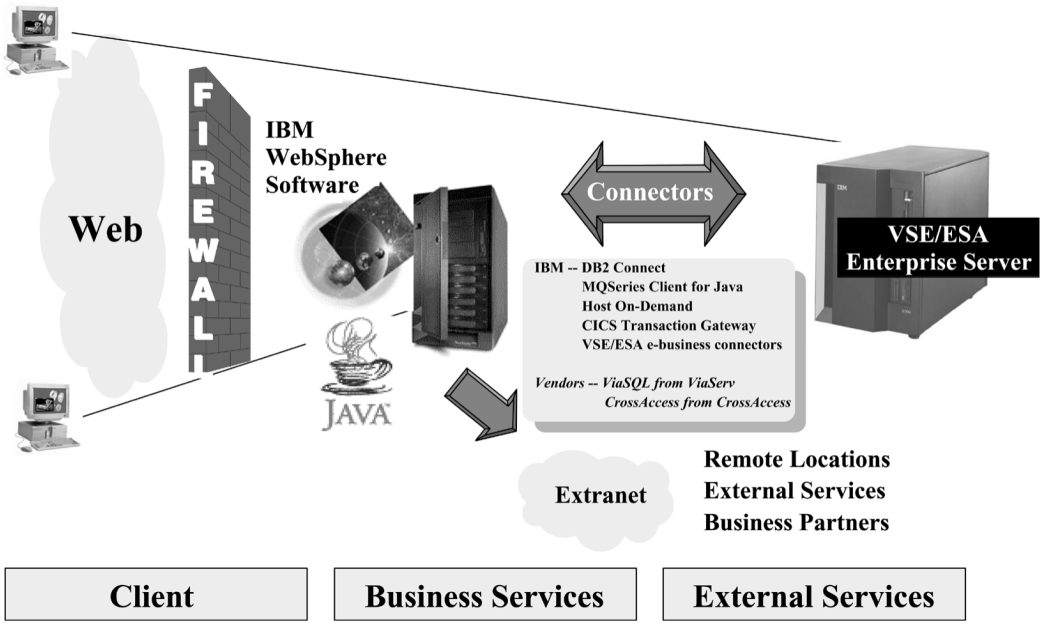


Figure 10. Three-Tier Implementations

Table 1. Connectors

	TCP/IP	SNA
Connectors to Applications		
MQSeries	yes	yes
CICS Transaction Gateway	no	yes
Connectors to Data		
DB2 for VSE and VM	DRDA (in beta)	DRDA
VSAM	VSE/ESA V2.5 Java-based connector	no
VSAM	VSE/ESA V2.5 DB2-based connector (DB2 required)	VSE/ESA V2.5 DB2-based connector (DB2 required)
DL/I	VSE/ESA V2.5 DB2-based connector (DB2 required)	VSE/ESA V2.5 DB2-based connector (DB2 required)
Connectors to Data (Products from ISVs)		
DB2 for VSE and VM	See note.	See note.
VSAM	See note.	See note.
DL/I	See note.	See note.
Selected non-IBM databases	See note.	See note.

Note: In some cases, ISVs offer capabilities that IBM currently does not, or alternative ways to achieve similar results. For example, ViaSQL (from ViaServ, Inc. (**http://www.viaserv.com/viaserv/mainpage.htm**) and CrossAccess (from CROSS ACCESS Corporation (**http://www.crossaccess.com/**) offer powerful data connectors that allow access to VSE VSAM and DL/I data, as well as data stored in some non-IBM databases.

Because products have different goals and/or implementations, making comprehensive "apples-to-apples" comparisons is difficult. In addition, ISVs often upgrade and improve their products in response to technology and customer needs. This paper cannot adequately describe the features and advantages of these and other products from ISVs. It doesn't attempt to provide definitive comparisons or make recommendations. More information is on the VSE home page (**http://www.ibm.com/s390/vse/**), where links are provided to the home pages of these and other ISVs offering a variety of VSE solutions. Customers with an interest are encouraged to contact the ISVs directly.

The new e-business connectors in VSE/ESA Version 2 Release 5 add even more options. Using these connectors, WebSphere applications will gain easy access to VSE resources such as VSE/VSAM data, VSE/POWER queues, and VSE Librarian files. For more information, see “e-business Connectors with VSE/ESA Version 2 Release 5” on page 25 and “A New Java-Based Connector for VSE/ESA Version 2 Release 5” on page 30.

2-tier Implementations

Earlier we said we would discuss ways to extend and enhance core applications using Web technology. That is a requirement frequently mentioned by VSE customers. They want to access core applications from an intranet or the Internet using a standard Web browser. Here are two options available from IBM.

1. *CICS Web Interface and 3270 Bridge (new function to be delivered as part of CICS TS for VSE/ESA in 2000)* – allow access to unchanged CICS applications, both BMS and 3270 Terminal Control applications. Customers might also write new CICS applications with new APIs that exploit the Web.
2. *IBM TCP/IP for VSE/ESA Application Pak* – In addition to FTP, TN3270, and print distribution, the Application Pak contains an entry level HTTP server. For some users, this is a good place to begin with simple intranet applications.

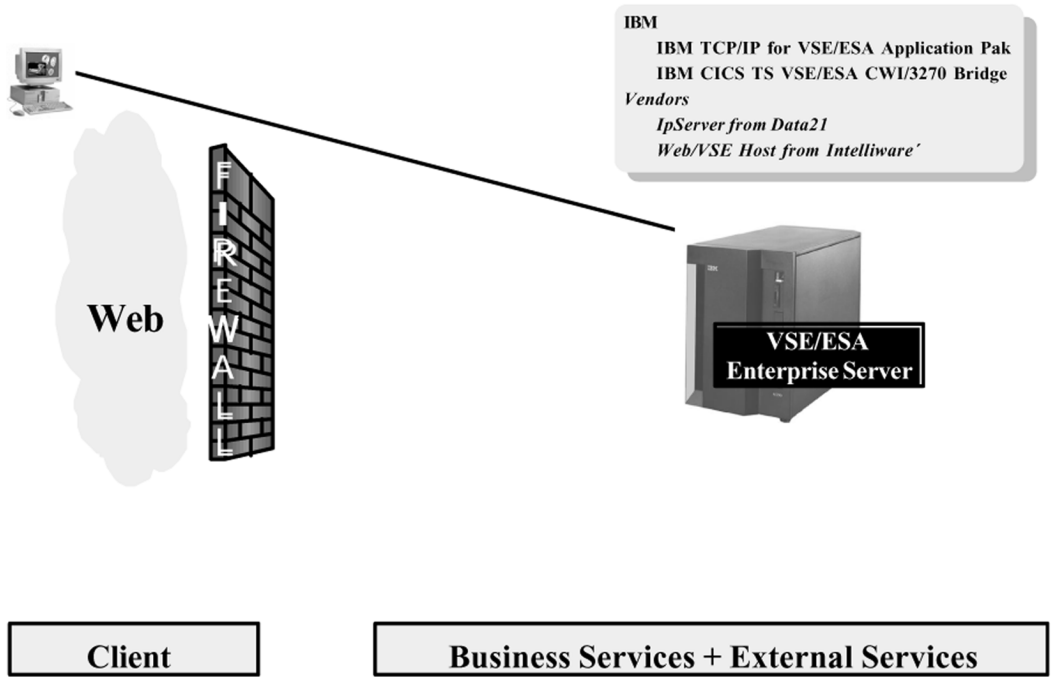


Figure 11. Two-Tier Implementations

The 2-tier approach can be a good way to start. However, over the long run the Application Framework for e-business (3-tier) offers more opportunities to create real competitive advantage for your company by implementing truly world-class e-business applications.

Note: In some cases, ISVs offer capabilities that IBM currently does not, or alternative ways to achieve similar objectives. For example, Web/VSE-Host (from *Intelli*Ware Systems, Inc. (**http://www.vseesa.com/** or **http://www.vsebusiness.com/**) automatically converts any 3270 screen to HTML. IpServer (from Data 21, Inc. (**http://www.data21.com/**) offers a Web server that runs in the familiar CICS environment.

Because products have different goals and/or implementations, making comprehensive 'apples-to-apples' comparisons is difficult. In addition, ISVs often upgrade and improve their products in response to technology and customer needs. This paper cannot adequately describe the features and

advantages of these and other products from ISVs, and it doesn't attempt to provide definitive comparisons or make recommendations. More information is on the VSE home page (**http://www.ibm.com/vse/**), where links are provided to these and other ISVs who offer VSE solutions. Interested customers are encouraged to contact ISVs directly.

Q and A

Q. What about VM-based Web servers from Sterling and Beyond? Can't I use one of those to provide Web access to my VSE system?

A. VM complements VSE in many ways. For some users this is another example of a long tradition of cooperation between VM and VSE. Here are a few considerations.

1. The suitability of a VM-based Web server is enhanced if **(a)** the customer already has VM and is familiar with its capabilities, and **(b)** the VM-based Web server can be justified based on access to VM applications or data. When customers fit this profile, a VM-based Web server is a reasonable option for accessing VSE applications and data as well.
2. *Clearly because of the strategic importance of the IBM Application Framework for e-business, the products and tools that are part of the framework, and the standards supported by the framework, WebSphere should be the preferred IBM solution in most situation.*

Summary

The IBM Application Framework for e-business is designed to enable world-class e-business applications and to protect and leverage existing investments in information assets.

While the 3-tier model offers a number of advantages and is the long term, strategic "technology-of-choice" for most e-business solutions, the 3-tier and 2-tier models aren't mutually exclusive. In order to achieve their e-business goals, VSE customers can choose from a variety of products from IBM and ISVs.

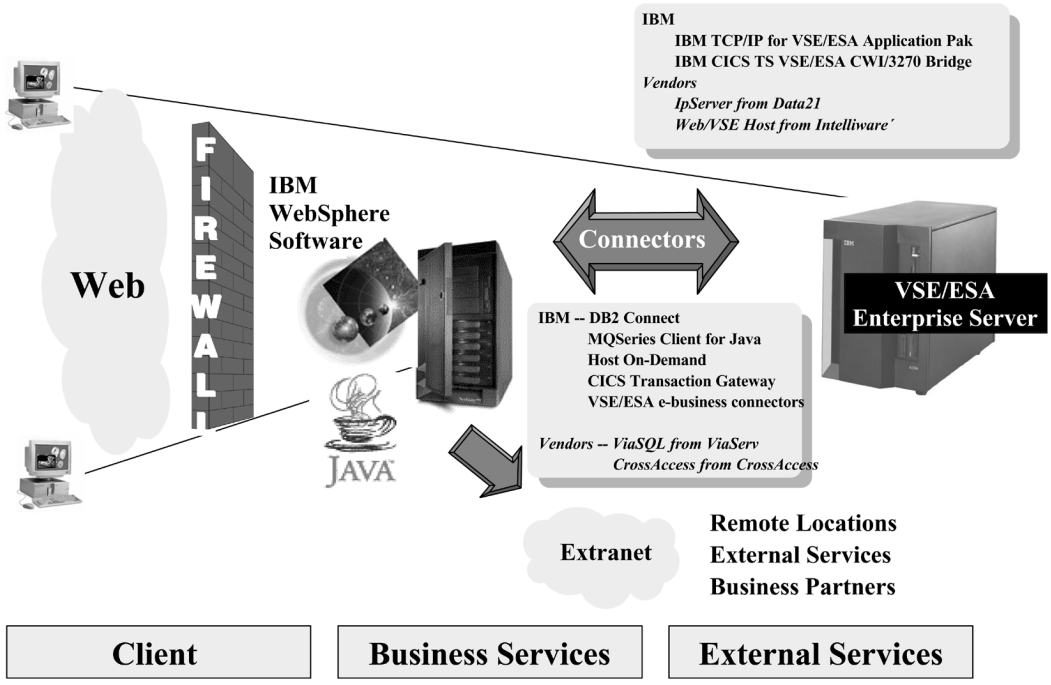


Figure 12. Summary

For a more technical, detailed description of VSE e-business solutions, please see the redbook:

e-business Solutions for VSE/ESA, SG24-5662-00.

Other related redbooks are at:

http://www.redbooks.ibm.com/

Outlook

The direction taken by each VSE customer will be based on their most pressing application needs.

- Those with significant CICS-based core applications and good skills will continue to include VSE in their IT infrastructure.

  However, the growing need for replacement and new applications (mostly e-business) applications means they're likely to implement *mixed server solutions that include both S/390 and non-S/390 servers.*

  Even within this group, there will be a range of implementations. For example, here are two situations representing different ends of the spectrum:

  1. Those with a strong need for innovative new e-business applications with world class capabilities will implement 3-tier models. They'll integrate CICS-based core applications and data with non-S/390 IBM servers (ie. RS/6000, Netfinity, or AS/400) running advanced enterprise-scale WebSphere-based applications. Because of the expertise needed and accelerated time-to-market demanded, many will rely on services from IBM or IBM Business Partners. Over time, the portion of their IT workload running on S/390 may decline relative to (but perhaps continuing to grow in absolute terms) the non-S/390 portion.

2. Those with a somewhat less aggressive e-business strategy will implement simple 2-tier solutions that focus on leveraging CICS-based core applications by enhancing the user interface and extending access across an intranet. They will rely primarily on their own skills and resources.

Of course, there's no "typical" or "average" customer. Most VSE customers will fall somewhere in the middle. They're likely to rely on both 3-tier and 2-tier models in combination to meet their own unique needs.

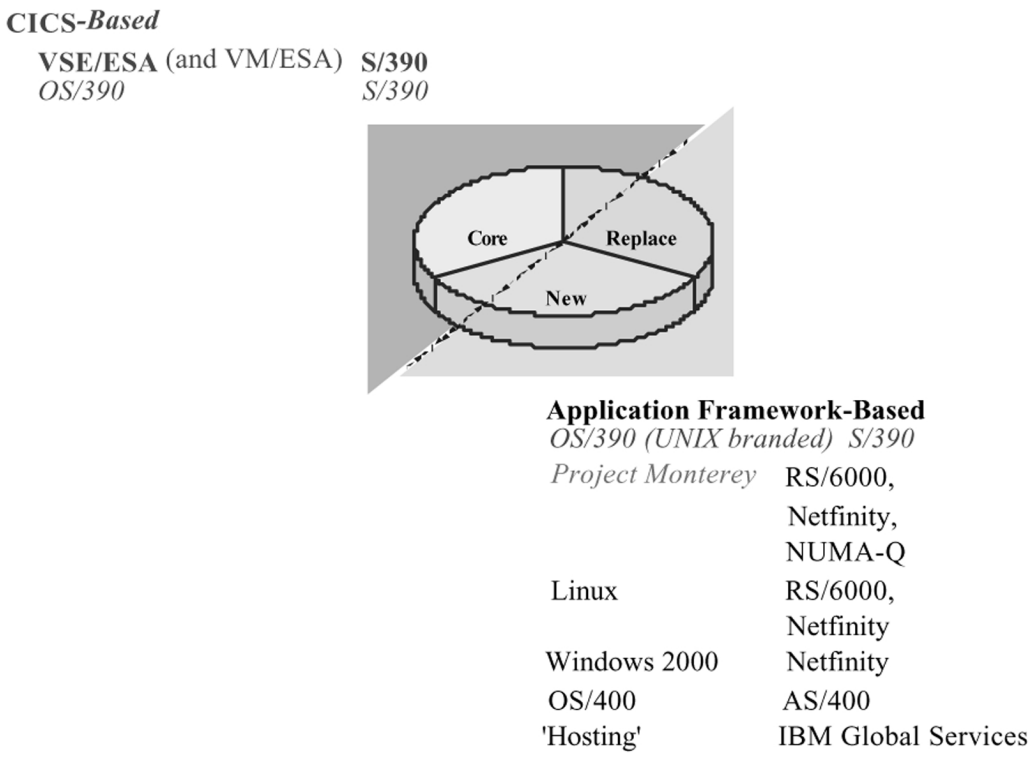


Figure 13. Outlook – Part 1

- Other VSE customers may follow a different path. For example;
  1. Some with strong core applications and one or more of these needs may migrate to OS/390
     - UNIX-based replacement applications such as SAP R/3 or PeopleSoft
     - the kind of secure, highly available e-business environment that WebSphere provides,
     - the operational advantages that OS/390 provides
     - a strategic reason (for example, a merger or acquisition)
  2. Some may replace their entire VSE system with a non-S/390 platform for the following reasons.
     - They replace major applications and choose to simplify their IT infrastructure.
     - They have modest replacement or new application needs, but nevertheless select packaged replacement applications because they've lost

even the most basic skills needed to keep their VSE systems and core applications up-to-date.

– a strategic reason (for example, a merger or acquisition)

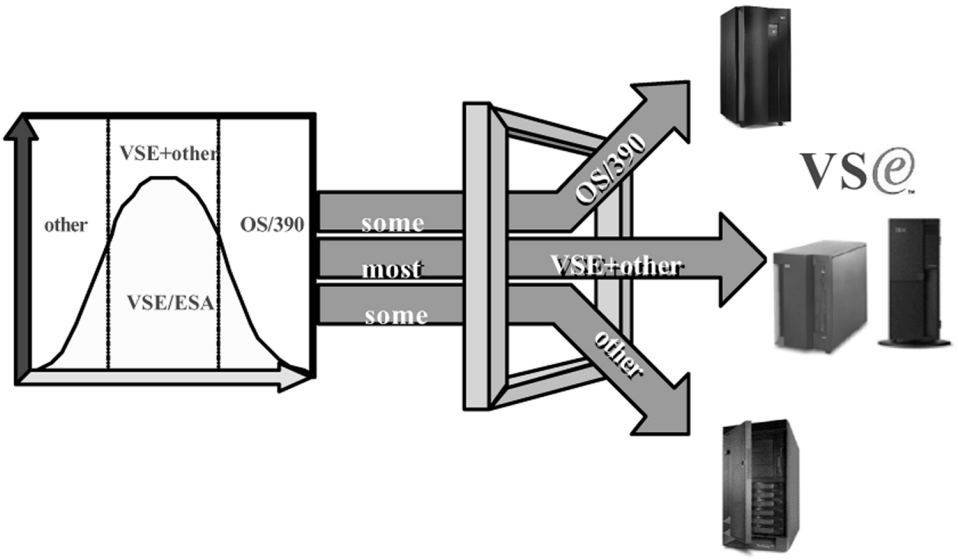


Figure 14. Outlook – Part 2

Q and A

Q. So what's the bottom line?

A. *VSE remains an outstanding, cost-effective solution for CICS and batch applications.* The "native," 2-tier e-business capabilities of VSE are (and are likely to remain) modest. *Nevertheless, VSE customers are able to implement even the most demanding, world-class e-business applications using IBM's Application Framework for e-business as their guide.*

Appendix

A few additional points are included here for completeness. However, they were not discussed in this paper:

- Using middleware products such as CICS Transaction Gateway or IBM SecureWay Host On-Demand, customers can design 3-tier access to core CICS-based applications. That is, it is possible to pass through a middle tier (such as a RS/6000 or Netfinity) to interact with VSE. End users may view 3270 "green screens" with a standard Web browser or add familiar "GUI" interfaces to CICS-based core applications.
- Using products from ISVs, it's possible to interact with VSE by way of a VM-based Web Server. One can think of this as a physical 2-tier (only a client and S/390 need be involved) but logical 3-tier (client, Web Server guest virtual machine, and VSE guest) system.
- In either model (2-tier and 3-tier), interactions may pass through an external firewall. Firewalls provide a specialized security function and, at least for purposes of this paper, aren't considered a separate "tier."

- Within the IBM Application Framework for e-business, Lotus Domino can be used as the Web Application server instead of WebSphere. There may be situations where Domino may be a better fit than the WebSphere Application Server.
- Another consideration for mixed server environments is IBM's new Enterprise Storage Server™. With ESS, a single high availability, high performance, and high capacity storage subsystem provides cost-effective storage for both S/390 and non-S/390 servers. Storage isn't "shared" in the sense of two systems being able to read and write the same bits. Instead, disk arrays can be flexibly partitioned so that the subsystem provides storage for several servers at the same time.

  VSE supports ESS. For more information on ESS, please visit the ESS home page at:
  http://www.storage.ibm.com/hardsoft/products/ess/ess.htm
- Project Monterey is an IBM-lead initiative whose objective is to develop a common UNIX for Intel-32 bit, Intel-64 bit, and Power architectures. Joining IBM in this initiative are industry-leading companies such as SCO, Intel, Compaq, and more. Information on IBM Project Monterey can be found at:
  http://www.ibm.com/servers/monterey/overview/index.html

VSE/ESA Version 2 Release 5 to Provide Improved Interoperability

Author

G. M. (Jerry) Johnston
Senior Advisor, IBM Böblingen Laboratory
p798000@us.ibm.com

The VSE customer environment is increasingly characterized by multiple servers – including different types of servers – and the need to exploit Internet technologies and e-business opportunities. VSE customers also want to protect and leverage their extensive investment in existing S/390 information assets. To support those environments and needs, IBM previewed VSE/ESA Version 2 Release 5 on February 29 with the announcement, *VSE/ESA Plans Improved Interoperability*.

Mixed Server Environment

In one recent survey, VSE customers reported having an average of 3.5 different types of servers in addition to S/390. Almost everyone had at least one server running Windows NT and at least one server running a variation of UNIX.

These servers had been acquired for a variety of reasons. Sometimes they resulted from mergers and acquisitions. Sometimes they were acquired by end-user departments along with a preferred application. In other instances, they were in use simply because they were a more capable or more cost-effective solution than VSE (or VSE with VM) for the intended application.

Whatever the reason, such mixed VSE environments are here to stay; and their numbers are likely to grow. As VSE customers implement e-business solutions and integrate their IT systems – no matter what platform they use – requirements for improved interoperability and connectivity based on industry standards become critical.

e-business Connectors

As described in the articles, “e-business Connectors with VSE/ESA Version 2 Release 5” on page 25 and “A New Java-Based Connector for VSE/ESA Version 2 Release 5” on page 30, interoperability for VSE environments will be enhanced with the addition of VSE *e-business connectors* in VSE/ESA. These connectors will allow easy access to VSE resources such as VSE/VSAM, VSE libraries, VSE/POWER, VSE/ICCF, and the system console. VSE/ESA Version 2 Release 5 will include server code that runs on VSE itself, plus associated Java beans and servlets that run on Java-capable clients.

VSE/ESA 2.5 e-business connectors are designed for a three-tier environment, illustrated in Figure 10 on page 15 The three-tier model allows VSE customers to build world-class e-business applications, yet protect their sizable investment in VSE skills and information assets. Leveraging VSE applications and data may allow customers to implement e-business applications at lower cost, less risk, and quicker time-to-market than alternatives requiring conversion or replacement of core VSE applications.

VSE e-business connectors will provide a basic interface. Proven IBM middleware such as MQSeries and some products from independent software vendors provide more capabilities. You also should note that VSE/ESA 2.5 e-business connectors are not intended to be a replacement for, or alternative to, such products. Customers who need access to VSE from another platform should consider these existing products now.

While interoperability enhancements support the three-tier e-business model, a two-tier model (Figure 11 on page 17) enables timely, cost-effective solutions that extend the robust CICS environment. VSE/ESA 2.5 also will integrate CICS Web Support and the 3270 Bridge for use in two-tier configurations.

Other Enhancements

Other enhancements for VSE/ESA Version 2 Release 5 include:

- A VSAM hashing algorithm for faster access to large VSAM Local Shared Resource pools. See “VSE/VSAM Buffer Hashing for VSAM LSR Pools with VSE/ESA Version 2 Release 5” on page 42.
- VSAM exploitation of IXPF/SnapShot. See “IXFP/SnapShot Support for VSAM Datasets with VSE/ESA Version 2 Release 5” on page 44.
- An increase in dynamic classes
- Support for the Enterprise Storage Server (or "Shark") Flashcopy feature.

Fast Service Upgrade (FSU) from VSE/ESA Version 2 Release 4 will be provided.

Additional Information

Previews provide insight into IBM plans and direction. General availability, prices, ordering information, and terms and conditions will be provided when the product is announced. For more information about VSE/ESA 2.5, see the IBM Software Preview Announcement of February 29, **200-031**.

Other sources of information related to VSE/ESA and e-business are:

- The white paper, *VSE Applications – How e-business Fits*, GF22-5137. It's reprinted as the first article in this issue of the *Newsletter*.
- The ITSO redbook, *e-business Solutions for VSE/ESA*, SG24-5662. The redbooks home page is:
  http://www.ibm.com/redbooks/
- More information about the IBM Application Framework for e-business is at:
  http://www.ibm.com/software/ebusiness/

e-business Connectors with VSE/ESA Version 2 Release 5

Authors

Wilhelm Mild
Thomas Weber
VSE Development, IBM Böblingen Laboratory
mildw@de.ibm.com
tweber@de.ibm.com

VSE/ESA 2.5 provides you with the resources to extend your core applications to e-business applications and to thus protect and leverage existing core-application investments. Core applications (typically CICS, COBOL, VSAM, DL/I) typically run on the VSE/ESA host, are critical to a company's operations, are expected to remain in production for many years to come, and usually represent an enormous investment of past resources.

On the other hand, e-business applications are typically based upon common standards such as TCP/IP, HTML, XML, and Secure Sockets Layer (SSL). They include both server and client code written in Java, access relational data locally or remotely, and interface with end-users via a standard Web browser.

The implementation of e-business connectors in VSE/ESA Version 2 Release 5 enables access to VSE/ESA resources from Web-based applications in heterogeneous environments. Two new connectors allow VSE/ESA to participate into an e-business environment:

- **Java-based connector** – provides access to VSE/POWER, VSE Librarian, VSE/VSAM, VSE/ICCF, and the VSE console. For more details, refer to “A New Java-Based Connector for VSE/ESA Version 2 Release 5” on page 30.
- **DB2-based connector** – takes advantage of a relational database environment and allows access to VSE/VSAM and DL/I resources.

Figure 15 on page 26 shows how you can access VSE/ESA resources through the new connectors. The remainder of this article explains how VSE/ESA 2.5 can take advantage of the DB2-based connector.

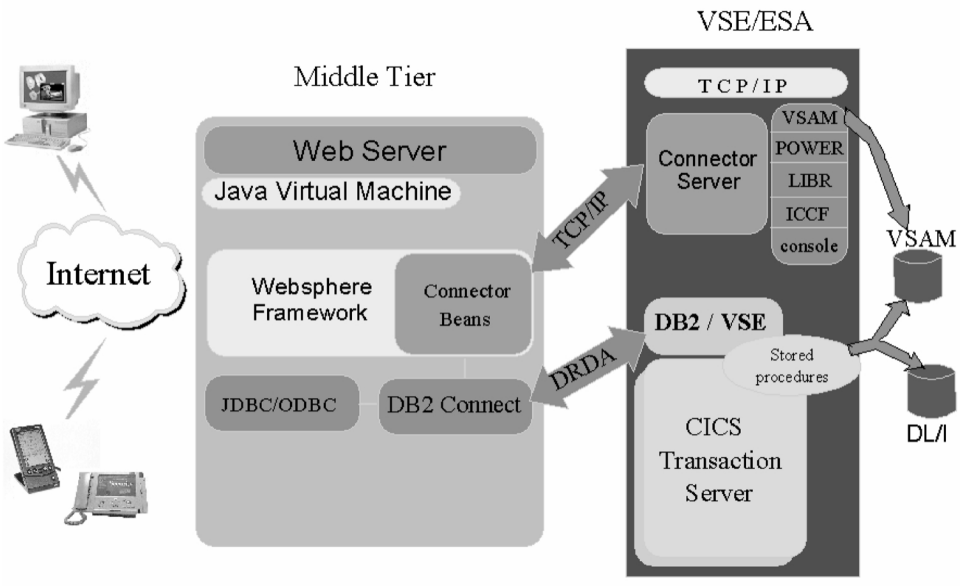


Figure 15. Connector-Based Access to VSE/ESA Resources

DB2-Based Connector Overview – Stored Procedures

The DB2-based connector uses Distributed Relational Database Architecture (DRDA) to access non-relational resources like VSE/VSAM and DL/I. The access from the requester is via standard interfaces like JDBC, ODBC or Call Level Interface (CLI). The implementation is based on the concept of *DB2 stored procedures*, available with the DB2 Server for VM and VSE Version 6 and above.

A stored procedure is a user-written application program that is compiled and stored at the server. When the database manager is running in multiple user mode, local applications or remote DRDA applications can invoke the stored procedure.

A DB2 stored procedure on VSE/ESA can be written in any LE (Language Environment)-compliant language. The API (Application Programming Interface) within a stored procedure differs depending upon whether VSAM or DL/I data is being accessed on the VSE/ESA host.

Advantages of using DB2 stored procedures for remote data access include:

- Reduced network traffic, since the SQL statements issued by a stored procedure are local to the VSE/ESA host. They do not incur the high network costs of distributed statements. Instead, single network send and receive operations can be used for processing all the statements contained in a stored procedure.
- You can use a stored procedure to hide the details of the database design from application programs running on a Web client or middle tier.
- If a database is modified, only the stored procedure (and not the client application programs) need to be modified.
- You can use a stored procedure to hide sensitive data from specific application programs.

- You can encapsulate business logic at the VSE/ESA host, instead of having to include this business logic in client application programs.
- It is easier to maintain an environment in which DB2 stored procedure applications are maintained at the VSE/ESA host, instead of being distributed across a number of Web clients or middle tiers.

Functional Description

The DB2-based connector in VSE/ESA Version 2 Release 5 consists of a client (requester) part and a server part. The client part invokes a stored procedure on the VSE/ESA server via JDBC, ODBC or CLI. On the server, a stored procedure must be coded and defined in DB2.

In the business logic within a stored procedure, different data types can be accessed and processed, like DB2, VSAM and DL/I. By issuing a stored procedure call, a client has the possibility to send parameter to the stored procedure. After completion of this stored procedure, the client receives the result parameters from the stored procedure.

Client Access

At the requestor (client) side, stored procedures can be invoked on the VSE/ESA system via stored procedure calls implemented in relational database interfaces like JDBC or ODBC/CLI.

Here are short examples of stored procedure calls on the client side:

```
JDBC:  String sql="Call<proc_name>(?,?,?,?,?,?)";
       statement=con.prepareCall(sql);
       statement.execute ();

ODBC/CLI: CALL procedure_name (?,?,?,?,...)
          SQLExecDirect() or
          SQLPrepare() followed by
          SQLExecute()

Embedded SQL:  CALLproc_name[(parm1[:parmind1],...,parmn[:parmindn])]
            or CALL proc_name USING DESCRIPTOR :sqlda
```

Access to VSAM Data within a Stored Procedure

Until now, VSAM data was mainly accessed using application programs which needed to know the internal structure of the VSAM data. The record layout was represented by data structures within the application programs. The disadvantages of using this method are:

- Impossibility to dynamically share a given record layout.
- The data structure is dependent on the programming language used.
- No way to know which program uses which data structure after compilation.

With VSE/ESA 2.5, it is possible to define global structures for the VSAM records and access them from all programs via the new VSAMSQL Call Level Interface (CLI).

VSAM Data Mapping

Since relational databases are column and table based, VSAM data must be mapped before it can be used. VSAM Data Mapping means creating a data map for a given VSAM record or record type. A map splits the VSAM record into columns (data fields) that have a name, a length, a data type, and an offset within the record. Optionally, a data view can be created which contains a subset of the fields contained within a map. A view always points to a subset of the data fields of a given VSAM map. Therefore, if you change a VSAM map, the views that use this map also will be affected. For example, deleting a map also will delete all views of this map. You can use views to give different user groups different views of the same data (for example, to hide some information from specific users).

The information about maps, views and columns of all VSAM clusters are stored in one single VSAM file, VSE.VSAM.RECORD.MAPPING.DEFS. This file is created automatically during the installation of VSE/ESA.

VSE/ESA 2.5 provides you with two methods of mapping your VSAM data for use with e-business applications:

1. The new IDCAMS command RECMAP. Within a VSE/ESA batch job stream, multiple maps and views can be defined or modified.
2. Writing a Java program that uses the VSE Java Beans class library, which provides classes to represent and manipulate maps, views, and fields.

Both methods allow to create, delete, change (alter), or list the contents of a map or view.

VSAM Call Level Interface

The new VSAMSQL Call Level Interface (CLI) is derived from the standard DB2 Call Level Interface to access data in a relational database. The VSAMSQL CLI consists of a set of VSE/ESA functions that can be used in a DB2 stored procedure to access VSAM data, as if it were relational.

These SQL statements are supported when you use the VSAMSQL CLI on mapped VSAM data:

```
INSERT INTO table (col1,col2,...)
       VALUES (val1,val2,...), (val3,val4,...),...

UPDATE table SET col1=val2, col2=val2,...
       WHERE col3=val3 AND col4=val4 AND ...

DELETE FROM table WHERE col3=val3 AND col4=val4 AND ...

SELECT col1,col2,...FROM table WHERE col3=val3 AND col4=val4 AND...
```

Access to DL/I Data within a Stored Procedure

To access DL/I data from within a stored procedure, a new interface was created. The AIBTDLI interface allows VSE batch programs to issue DL/I calls, without a DL/I batch environment having been established (using DLZRRC00 or DLZMPI00).

The client program issues a stored procedure call; and the parameters sent in the call will be forwarded to AIBTDLI, which will then build the DL/I calls.

AIBTDLI passes the DL/I calls to DLZMPX00, which connects to a running CICS/DLI online system in the form of an MPS batch task. DLZMPX00 associates each VSE batch task with a DLZBPC00 mirror task on the CICS/DLI online-side, which acts on its behalf in the known MPS scheme. The databases reside (are "opened") on the CICS/DLI online side.

All DL/I calls passed to (and results and feedback information returned from) the CICS/DLI online system are routed through DLZMPX00 via a unique XPCC connection between the VSE batch and the CICS/DLI BPC mirror task. Database access contention, resource logging, and recovery are performed on the CICS/DLI online system using existing CICS/DLI functions.

Prerequisites

To take advantage of the DB2-based connector, you need:

- DB2 Server for VN and VSE Version 6 or above
- DB2 Connect to access DB2 data on VSE/ESA
- The new VSAMSQL CLI shipped with VSE/ESA 2.5
- The new DL/I functionality AIBTDLI

More Information

A new VSE manual, *VSE/ESA e-business Connectors, User's Guide* (SC33-6719), describes the new connectors in detail and provides coding examples.

The e-business link on the VSE/ESA home page will contain the latest updates for these new connectors.

A New Java-Based Connector for VSE/ESA Version 2 Release 5

Authors

Ingo Franzki
Jörg Schmidbauer
VSE Development, IBM Böblingen Laboratory
ifranzki@de.ibm.com
jschmidb@de.ibm.com

The first issue of the *VSE/ESA Software Newsletter* in 1999 contained an article about a new Java-based user interface we call the "VSE Navigator Function". This was the initial approach to bring VSE together with Java. Since then, many of you have downloaded the VSE Navigator code from our FTP server.

The VSE Navigator, however, is just the beginning of a new Java-based infrastructure for VSE. The VSE Navigator not only provides a graphical *user interface* for VSE. There is also a Java-based *programming interface*. But until now, you did not have a complete set of Java classes that can be used in any kind of Java program, such as applets or servlets.

Today, about one year later, you can see the complete picture.

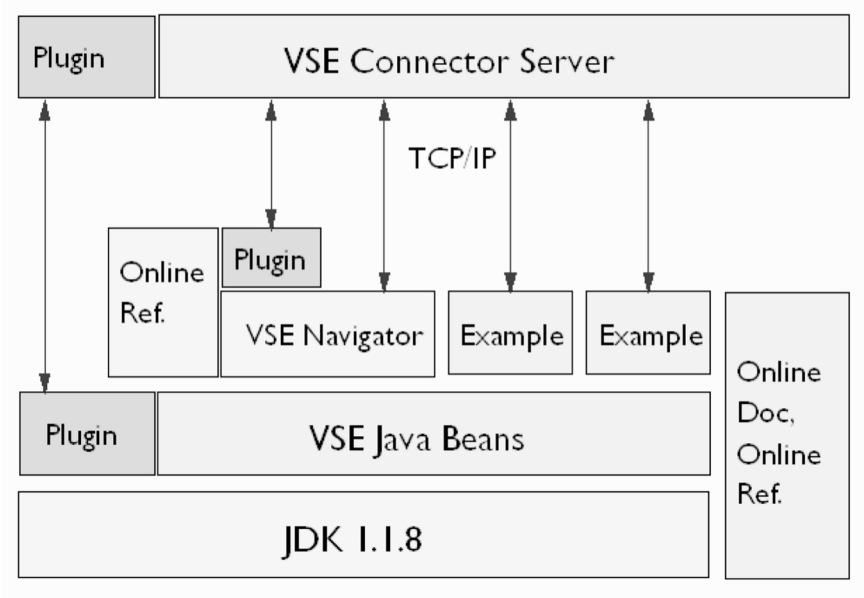


Figure 16. Overview

The VSE Connector Server – together with its counterpart, the VSE Connector Client – represent a new Java-based connector for VSE/ESA. There is other new functionality which makes use of the DB2 infrastructure to access VSAM and DL/I data. This is described in “e-business Connectors with VSE/ESA Version 2 Release 5” on page 25.

The VSE Connector client provides a library of Java Beans (VSEConnector.jar) that allow to access VSE file systems (VSE/POWER, Librarian, VSE/ICCF, VSE/VSAM),

to submit jobs, and to access the operator console by communicating with the VSE Connector Server over TCP/IP. There is a special mechanism to map VSAM data to a relational structure. Access to VSE/ICCF is read-only. This class library is currently implemented on the basis of Java 1.1.8.

The VSE Java Beans library, its programming reference, and many coding examples that show how to write Java applications, applets, servlets, server pages, and Enterprise Java Beans are included in a HTML-based online documentation. Many samples are ready to run. Others need to be modified and recompiled to match your networking environment.

The VSE Navigator itself is implemented on the basis of the VSE Java Beans. There are no hidden interfaces. Thus the Navigator is just one example, showing the possibilities of the VSE Java Beans library.

While Java Beans are usually visual GUI components (like push buttons, check boxes, or sliders), the VSE Java Beans conform to the specifications for Java Beans, but are not visual components. Instead, they represent VSE-based objects such as:

- File systems (VSE Librarian, VSE/POWER, VSE/ICCF, VSE/VSAM).
- System components (such as the operator console).
- Data objects (such as VSE libraries, POWER queue entries, and VSAM catalogs).

It is possible to extend the VSE Java Beans library by your own plugins, which means "additional VSE Java Beans". The same is true for the VSE Connector Server, where plugins have to be implemented in C for VSE/ESA. So you can write a client/server plugin, where your client plugin talks to your server plugin and also client plugins using the existing VSE Connector server functionality.

But the first steps will surely be to run the samples, modify them, and then write own Java programs on the basis of the VSE Java Beans. If you are familiar with the class library, you may extend it by your own beans.

Providing Web-based access to VSAM data requires to bring the data into a standard format. The next topic describes how VSAM data is mapped to a relational structure.

Mapping VSAM Data to a Relational Structure

In the past, VSAM data was mainly accessed using application programs that understood the internal structure of the VSAM data. The record layout was represented by data structures within the application programs. The disadvantages of using this method are:

- There is no way to dynamically share a given record layout with other applications.
- If the record layout changes, all application programs must also be changed.
- The data structure is dependent on the programming language (for example COBOL, Assembler, or PL/I).
- Formatted data reports have to be created on the operating-system platform on which the application programs run.

However, the development of e-business applications requires:

- Sharing of data representation across operating-system platforms.
- Easy access to data representations from different applications.
- That data representation and data display are independent of operating system and programming language.

VSAM Data Mapping first means to create a *data map* (referred to simply as a map) for a given VSAM record or record type. A map splits the VSAM record into columns (data fields) that have a name, a length, a datatype, and an offset within the record.

Then you can optionally create a *data view* (referred to simply as a view) that contains a subset of the fields contained within a map. A view always points to a subset of the data fields of a given VSAM map. Therefore, if you change a VSAM map, the views that use this map will also be affected. For example, deleting a map will also delete all views of this map. You can use views to give different user groups different views on the same data (for example, to hide some information from specific users).

VSE 2.5 provides you with two methods of mapping your VSAM data so it can be used within e-business applications:

1. Use the new IDCAMS command RECMAP.
2. Write a Java program that uses the VSE Java Beans class library, which provides classes to represent and manipulate maps, views, and fields.

Both methods allow to create, delete, change (alter), or list the contents of a map or view.

The mapping definitions have the following hierarchical structure:

```
CATALOG (MY.USER.CATALOG)
  CLUSTER (MY.DATA.CLUSTER)
    MAP (Map One)
      COLUMN (Name, Offset=0,Len=10,Type=String)
      COLUMN (Street, Offset=10,Len=20,Type=String)
      ...
      VIEW (View One)
        COLUMN (PersonInfo, Ref=Name)
        COLUMN (Address, Ref=Street)
        ...
      VIEW (View Two)
      ...
    MAP (Map Two)
    ...
```

A *data field* represents one specific column of a VSAM record. It is always part of a given map or view. It consists of:

- a name (the column name)
- a length
- a datatype
- an offset within the record

On the VSE 2.5 host, the maps of all VSAM clusters are stored in one single VSAM file, VSE.VSAM.RECORD.MAPPING.DEFS. This file is created automatically

during the installation of VSE/ESA. The short name for this VSAM file is IESMAPD, and the cluster for this file is stored in VSESP.USER.CATALOG.

After mapping your VSAM data to a relational structure, you can now write Web applications that allow to display and even modify the data. Of course, it is also possible to include other VSE-based data into a Web page, but only VSAM requires data mapping. The following topic describes the major programming concepts.

Writing Java-Based Web Applications for VSE

In the past, the terms applets, servlets, and Enterprise Java Beans were usually not used together with VSE. Now, all of them will become natural parts of VSE server environments.

Applets

You can write Java applets that get downloaded to any Web browser requesting a Web page from your VSE-based HTTP server, or any other Web server running on a third platform (middle-tier platform) between the Web client and VSE. In both cases, the applet opens a connection to the VSE Connector Server, getting access to VSE-based data and functionality.

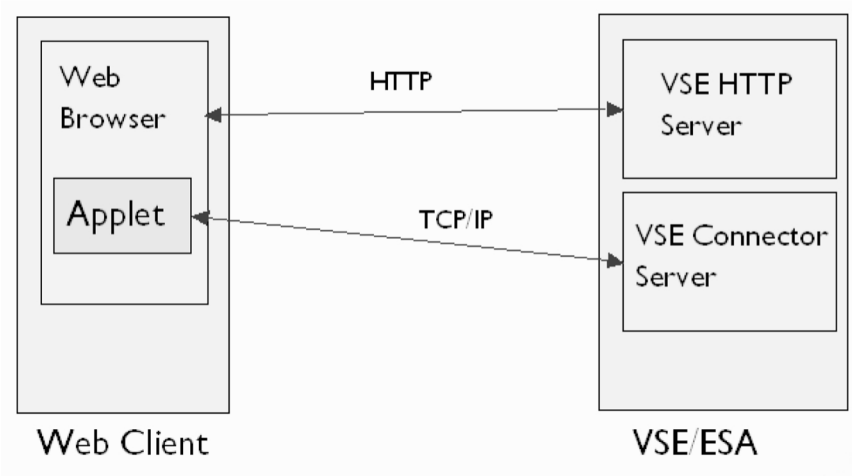


Figure 17. Applets

Applets, by nature, have many restrictions. They can open a new network connection only to the platform from where they are downloaded. In a 3-tier environment, this is the middle-tier. To get around this problem and allow applets to get data from the VSE Connector Server running on VSE, a simple router (VSEAppletServer) is provided. Thus the applet just connects to this router. The router, which does not have this restriction, then connects to the VSE Connector Server to get VSE-based data and pass it over to the applet.

The VSE Connector client provides applet examples to:

- Map VSAM data and
- Display used and free VSAM space.

As implied by these samples, you will usually use applets in your company's intranet for administration purposes and visualization of VSE-based data.

Servlets

You can write Java servlets that run inside the Java Virtual Machine of a Web Application Server, like IBM's WebSphere, to generate dynamic Web pages. So you can include any VSE data (e.g. VSAM or POWER data) into a dynamically created Web page. This requires a middle-tier platform running a Web server and WebSphere. The servlet then opens and maintains a connection to the remote VSE host. This is based on IBM's *Common Connector Framework* (*CCFL*), allowing for maintaining a pool of connections where a servlet can reuse an already opened VSE host connection in its next lifetime. See “Maintaining VSE Host Connections in Servlets” on page 35.

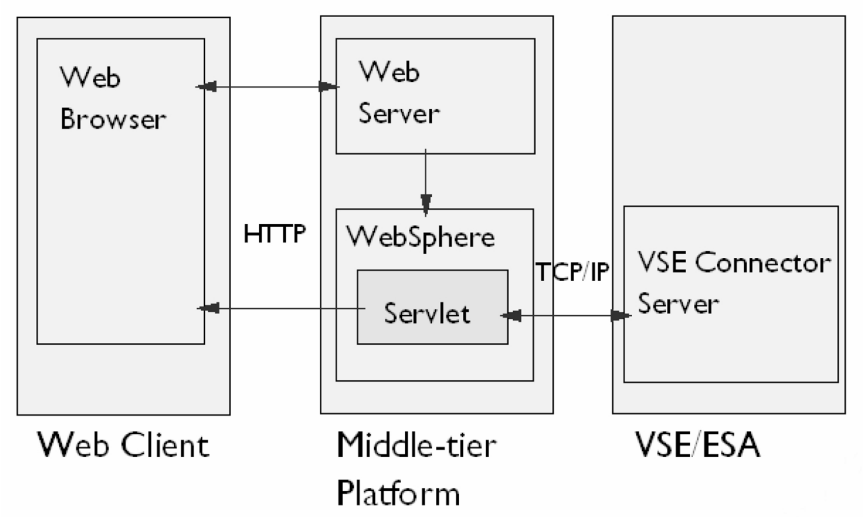


Figure 18. Servlets

The VSE Connector client provides the following servlet examples:

- A search engine to search the VSE library systems for files and containing text.
- A simple VSAM-based flight booking system.
- A servlet example demonstrating how to reuse VSE host connections.

Servlets are your first choice for implementing e-business and giving your customers access to VSE-based data over the Internet.

Enterprise Java Beans

You can even represent VSAM data by Enterprise Java Beans (EJBs) and thus make VSAM data accessible in a standard platform-independent format in a heterogeneous network. One EJB represents the data of one given VSAM record that is mapped to a relational structure. The bean can be accessed through Java applets or servlets from any Web client.

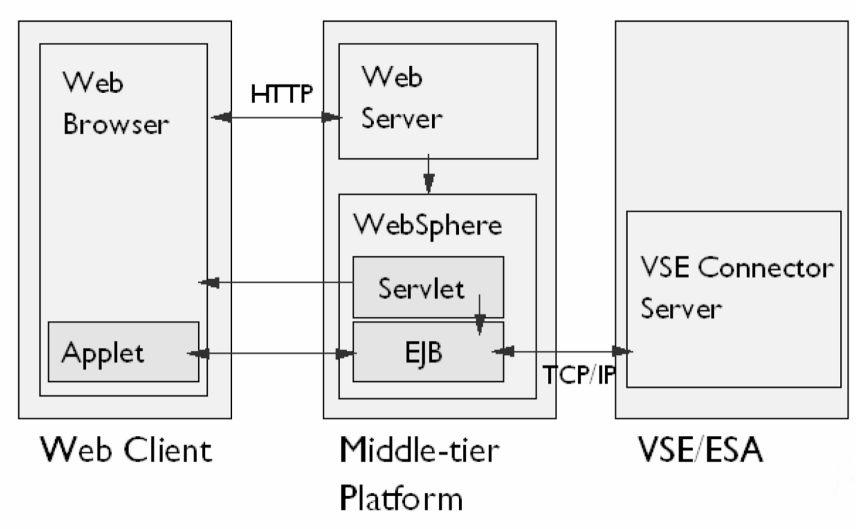


Figure 19. Enterprise Java Beans

The VSE Connector client provides the following EJB example:

- Representing VSAM records through EJBs and accessing the data through an applet.

You will use EJBs to represent VSE data, if the data is accessed from different platforms within a heterogeneous network.

The following topic goes in more detail of CCF and shows how the problem with maintaining VSE host connections in a 3-tier environment is solved.

Maintaining VSE Host Connections in Servlets

One general problem with writing short living programs, like servlets, in a 3-tier environment is how to maintain the connection from the middle-tier platform to the remote VSE server, because opening and closing connections is relatively expensive. To solve this problem, IBM's Common Connector Framework (CCF) allows for maintaining a pool of connections that can be reused by the same servlet in its next lifetime and also by other servlets running on the same Web application server. The CCF class library is part of IBM's Visual Age for Java and included into the VSE Java Beans Java archive (VSEConnector.jar).

When a new VSE connection is opened, it is not closed immediately when the related servlet terminates. Instead, it is put into the CCF connection pool. The lifetime of the connection in the pool is specified when creating the connection. The pool, and therefore the connection itself, is tied to a CCF-controlled ConnectionManager object. The connection manager object lives as long as the servlet is loaded in memory. Normally, this is as long as the Web application server is up and running.

Re-using a VSE host connection is achieved by simply storing the connection properties in the httpSession object, which represents the client connection and which is always the same as long as the Web browser stays connected to the Web server. Each time the servlet is invoked, it checks whether there is already a VSE host connection available for reusing. The httpSession instance is also created when

the servlet is invoked the first time and stored in the HTTPServletRequest instance, which is always the same for all servlet calls. Thus you must carry out two tasks:

1. Create the httpSession instance and store it in the HTTPServletRequest (which is a servlet input parameter), and
2. Create the VSEConnectionSpec for the host and store it in the httpSession.

All provided servlet samples show this mechanism. Only few lines of code are necessary to implement the concept. The following topic gives an overview of the parts contained in the new VSE Connector component.

Parts of the VSE Connector Component

The new VSE component "VSE CONNECTOR", 5686-06635-55N, consists of the following parts.

The VSE Connector Client

The VSE Connector Client is given by just one single file, *install.class*, which is a Java class file that contains the following parts:

- The VSE Java Beans class library VSEConnector.jar. The VSE Java Beans enable a Web browser to communicate with VSE/VSAM, VSE/Librarian, VSE/POWER, VSE/ICCF, and the operator console on the VSE/ESA host.
- A set of samples, including Java source code, that show you how to write Java programs that are based upon the use of VSE Java Beans.
- Online documentation (a set of HTML pages) describing the various concepts and samples.
- A Java-based installation utility that allows the client to be installed on any Java enabled platform.

On VSE 2.5, this file is shipped in PRD1.BASE as a W-member, *IESINCON.W*. Download the member in binary and rename it to *install.class*.

The VSE Connector Server

The VSE Connector Server is an LE batch application that runs in any static or dynamic partition (default is class R) and implements a TCP/IP socket listener. It communicates with Java-based client applications requesting VSE data and invoking VSE functions. The VSE Connector Server is derived from the VSE Navigator Server and is part of VSE 2.5. A job to start the server (STARTVCS) is placed in the Reader queue during initial install. There is a set of configuration files in PRD2.CONFIG (skeletons in ICCF library 59) that can be modified to define the server properties.

VSAM Call Level Interface

The new VSAM Call Level Interface (CLI) is derived from the standard DB2 Call Level Interface to access data in a relational database. The VSAM CLI consists of a set of C for VSE/ESA functions that can be, for example, used in a DB2 stored procedure to access VSAM data as if it were relational. An online reference is contained in the VSE Connector client.

Using the VSE Java Beans

This example shows how to display and update VSAM records through a simple standalone Java application. It operates on a VSAM KSDS cluster FLIGHT.ORDERING.FLIGHTS, which is also the basis for the flight booking servlet example.

VSE Java Beans classes are shown in italics. A complete reference (javadoc) is contained in the VSE Connector Client.

Step 1:

Initialize and prompt for the IP address of the VSE host, a valid VSE user ID and password.

```
package com.ibm.vse.samples;
import java.lang.*;
import java.net.*;
import java.io.*;
import java.util.*;
import com.ibm.connector.internal.*;     // include CCF classes
import com.ibm.vse.connector.*;          // include VSE Java Beans

public class VsamDisplayExample
{
   public static void main(String argv[]) throws IOException
   {
      VSEConnectionSpec spec;
      VSESystem system;
      VSEVsam vsam;
      VSEVsamCatalog catalog;
      VSEVsamCluster cluster;
      VSEVsamRecord record, newRec=null;
      VSEVsamMap map;
      RecordListener rl;
      byte[] inputArray;
      String ipAddr, userID, password;
      Vector vRecords;
      String flightNum, newField="";

      String catName  = "VSAM.MASTER.CATALOG";
      String fileName = "FLIGHT.ORDERING.FLIGHTS";
      String mapName  = "FLIGHTS_MAP";

      /* Prompt for IP address, user ID and password. We have to remove */
      /* trailing blanks from the input strings before using them. */
      ...
```

Step 2:

Create the connection specification, which holds the properties of the physical host connection and is stored permanently in the Common Connector Framework (CCF) connection manager instance. The VSEConnectionSpec contains the parameters of a connection. When the connection manager is asked to give us a connection for a specific VSEConnectionSpec, the CCF connection manager searches its connection pool for an unused connection that matches the given VSEConnectionSpec. If no unused connection is available, the connection manager creates a new connection using the parameters specified in the VSEConnectionSpec.

```
try {
   spec = new VSEConnectionSpec(InetAddress.getByName(ipAddr),
                                2893, userID, password);
}
catch (UnknownHostException e)
{
   System.out.println("Unknown host : " + e);
   return;
}
spec.setReapTime(5);
spec.setMaxConnections(5);
spec.setMinConnections(0);
spec.setUnusedTimeout(10000);   // milliseconds

/* Stay logon with this user for lifetime of this connection */
spec.setLogonMode(true);
```

Step 3:

Create a VSE system instance with the given connection specification. The following if-statement checks whether there is already an existing connection to this host. If yes, the existing connection is reused. If not, a new connection is opened.

This construct is used in all servlet examples of the VSE Connector client. It is not necessary for a standalone Java application. Applications cannot reuse connections, because the CCF instances are also terminated when the application ends. For servlets it's different. A servlet usually runs only for a small amount of time, but the code is loaded into memory of the Web application server as long as the server runs. And the same CCF instance can be reused by different servlets and servlet lifetimes.

Note: The VSESystem bean maintains the connection internally. Therefore, programmers normally will never use the connection directly. The VSESystem is created with a VSEConnectionSpec that is later used to get a connection from the ConnectionManager. If two VSESystem objects are created with the same VSEConnectionSpec (the parameters within the VSEConnectionSpec must be the same but not necessarily the same VSEConnectionSpec object), the two VSESystem objects can share one connection.

```
system = new VSESystem(spec);
if (!(system.getConnectionManager()
      instanceof ConnectionManager))
{
   system.setConnectionManager(new ConnectionManager());
}
```

Step 4:

Get the records of the VSAM cluster containing the available flights. A callback function (RecordListener) is created, which is called from the VSE Beans whenever a data instance is received from the host.

```
vsam = system.getVSEVsam();
rl = new RecordListener();
vsam.addVSEResourceListener(rl);
map = new VSEVsamMap(system, catName, fileName, mapName);
cluster = new VSEVsamCluster(system, catName, fileName);
cluster.addVSEResourceListener(rl);
cluster.selectRecords(map);
cluster.removeVSEResourceListener(rl);

/* Get vector containing all records from listener */
vRecords = rl.getRecords();
```

Here is the major code part of the RecordListener class. This class implements the Java interface *VSEResourceListener*, which is always used to get data instances from the host. The method listAdded() is called for each data instance that is received from the host. The caller is blocked until all data objects are received. Here, we just add the received instances to an internal vector, which can be accessed later by the caller. Of course, we could also process the instances here.

```
public class RecordListener implements VSEResourceListener
{
   Vector vRecs;
   ...

   public void listAdded(VSEResourceEvent event)
   {
      VSEResource resource = (VSEResource)(event.getData());
      if (resource instanceof VSEVsamRecord)
      {
         VSEVsamRecord rec = (VSEVsamRecord)resource;
         vRecs.addElement(rec);
      }
   }

   ...

   public Vector getRecords()
   {
      return vRecs;
   }
}
```

Step 5:

Now display the records on standard out. The outer loop goes over all elements of the vector vRecords, which was obtained from the RecordListener instance. The inner loop goes over all columns of the given record. The number of columns can be determined through method getNoOfFields() of class VSEVsamMap.

```
      int numMapFields = map.getNoOfFields();
      System.out.println("Records in file " + fileName + " :");
      for (int k=0;k<vRecords.size();k++)
      {
         System.out.println("Record " + k + ":");
         record = (VSEVsamRecord)(vRecords.elementAt(k));
         for (int i=0;i<numMapFields;i++)
         {
            try {
               if (map.isFieldPartOfPrimaryKey(i))
                  System.out.println(map.getFieldName(i) + " (Key) : " +
                                     record.getField(i).toString());
               else
                  System.out.println(map.getFieldName(i) + " : " +
                                     record.getField(i).toString());
            }
            catch (Exception e)
            {
               ...
            }
         }
      }
```

Step 6:

Now add a new flight to the cluster. In this step, we just fill the key field with data and try to add the record to the cluster. If this key already exists, we will get an exception. Let's loop until the user enters a new non-existing flight number or a negative number to quit.

```
boolean done = false;
while (!done)
{
   inputArray = new byte[map.getFieldLength(0)];
   System.out.println("Please enter a new flight number:");
   System.out.println("(Enter a negative value to quit)");
   System.in.read(inputArray);
   flightNum = new String(inputArray);
   flightNum = flightNum.trim();
   if ((new Integer(flightNum).intValue()) < 0)
      return;

   /* Check if new flight number already exists ... */
   try {
      newRec = new VSEVsamRecord(system, catName, fileName, mapName);
      newRec.setKeyField(0, new Integer(flightNum));
      newRec.add();
      done = true;
   }
   catch (Exception e)
   {
      System.out.println("Exception when adding new record.");
      if (e instanceof AlreadyExistentException)
         System.out.println("This flight number already exists.");
      else
         System.out.println("e = " + e);
   }
}
```

Step 7:

Now add all other data fields (columns) for the new flight. The field names (column names) can be derived from the given VSEVsamMap. Finally, make the changes permanent by calling the commit() method of class VSEVsamRecord. This writes the new contents of our local VSEVsamRecord instance to the VSAM cluster on the host.

```
      for (int i=1;i<numMapFields;i++)
      {
         inputArray = new byte[map.getFieldLength(i)];
         System.out.println("Please enter value for field " +
                            map.getFieldName(i) + " (Length = " +
                            map.getFieldLength(i) + ")");
         done = false;
         while (!done)
         {
            System.in.read(inputArray);
            newField = new String(inputArray);
            newField = newField.trim();
            if (newField.length() > 0)
               done = true;
         }

         if (map.getFieldType(i) == VSEVsamField.TYPE_STRING)
            newRec.setField(i, newField);
         else
            newRec.setField(i, new Integer(newField));
      }

      /* Make changes permanent... */
      newRec.commit();
      System.out.println("New record added to cluster.");
   }
}
```

Prerequisites

To take advantage of the new framework of connectors, you need:

- VSE/ESA 2.5
- The latest level of TCP/IP for VSE/ESA™ (needed for the VSE Connector Server).
- The latest level of the LE runtime (needed for the VSE Connector Server).
- A Java development kit (JDK) 1.1.6 or higher on the client side to use the VSE Java Beans class library and run the samples contained in the VSE Connector client.
- A Web browser to view the online documentation and programming reference of the VSE Connector client.
- DB2 Server for VM and VSE, with stored procedure support.

More Information

A new VSE book, *VSE/ESA e-business Connectors, User's Guide* (SC33-6719), describes the new connectors in detail and provides many coding examples. More related information also will become available via the VSE/ESA home page, specifically in the section, "e-business for VSE/ESA customers."

The first article about the VSE Navigator can be downloaded from the VSE/ESA home page. The URL of the *VSE/ESA Newsletter* issue is:

http://www.ibm.com/s390/vse/vsehtmls/vsenew18.htm

Click on the file **vsecnav.pdf**.

For more information, send an e-mail to:

- Ingo Franzki (ifranzki@de.ibm.com)
- Joerg Schmidbauer (jschmidb@de.ibm.com)

VSE/VSAM Buffer Hashing for VSAM LSR Pools with VSE/ESA Version 2 Release 5

Author

Guenter Weigelt
VSE/VSAM Development, IBM Böblingen Laboratory
guenter_weigelt@de.ibm.com

... from the IBM Preview Software Announcement Letter, 200-031 (February 29, 2000), for VSE/ESA Version 2 Release 5:

> *Large VSAM LSR buffer pools can improve response time and reduce I/O. However, until now, searching the pool to find the right buffer took time. Benefits can be offset and even reversed because of increased CPU time needed to search large buffer pools.*
>
> *VSAM Buffer Hashing: a VSAM hashing algorithm will allow direct access to the buffer needed. VSE/ESA Version 2 Release 5 customers gain the performance advantages of very large buffer pools without also incurring a significant CPU penalty that offsets the potential gains.*

What Is Buffer Hashing?

VSAM buffer hashing is a new function in VSE/ESA 2.5. It replaces the current buffer management for applications using the LSR (Local Shared Resource) option. The new buffer management technique provides the following improvements over existing sequential buffer management:

- Buffer search overhead is reduced by eliminating the need to do sequential search through the buffer pool. The new search technique uses a hashing algorithm. Through implementation of this algorithm, the path length of the search is significantly shortened.
- Not only is the path length shortened, but it is also consistent regardless of the number of buffers. In other words, search time is independent of the buffer pool size.

How Does Buffer Hashing Work?

The hashing technique is implemented using the following:

- A Hash Table – This is a table in main storage where each entry in the table is used as a pointer to a BCB. A BCB is the Buffer Control Block that contains the address of the buffer (for data or index) and information on the buffer itself. There is one and only one BCB for each buffer in the LSR buffer pool.
- A Synonym – When using a hashing technique, there is always the chance of having synonyms, because two or more entities may hash to the same anchor point. In this implementation, synonyms are chained together in the BCB. However, the chance of having synonyms is very small, and the chain is usually very short.
- The Hash Algorithm – The algorithm is calculated as follows:

  X = remainder of (RBA/2 + DSID1/2 + DSID2/2) / DIM.

  X (the remainder of the above calculation) is used as index into the hash table.

RBA = Relative Byte Address. This is the argument used by VSAM to identify a certain buffer in the LSR buffer pool.

DSID1 and DSID2 = Data Set Identifiers. DSIs are unique identifications of a certain opened VSAM data set component, either a data or an index component.

DIM = number of entries in the Hash Table.

DIM = (2N-1).

N = number of buffers in the subpool.

A simple example[1]:

1. Let's assume we have a LSR pool with 10 buffers. The Hash Table will have (2 * 10 -1) = 19 entries.

   DIM = 19

2. A VSAM GET operation reads a data record from a certain VSAM data set with the internal data set identifications DSD1 and DSD2 into a data buffer.

   DSID1 = 220, DSID2 = 32

   The BCB pointing to that data buffer is at storage location '640000'. The RBA (Relative Byte Address) of the VSAM data buffer is 800.

   RBA = 800

3. The hash algorithm
   X = remainder of (RBA/2 + DSID1/2 + DSID2/2) / DIM
   calculates the following index for the hash table:

   (800/2 + 220/2 + 32/2) /19 = 27, **remainder = 13 = X**

   "13" will be used as index into the hash table.

4. The BCB pointer **'640000'** will be stored into the **13th position** of the hash table.

5. Whenever another request is searching for a data buffer with RBA **800** from this certain dataset, the hash algorithm can calculate easily the index of **13** into the hash table and use the BCB at address **'640000'** and its related data buffer without a long pool search. This hashing technique works, of course, also with large buffer pools (e.g. **32767** buffers)!

Performance Improvements

Using of buffer hashing improves the performance. Very large buffer pools could be used without incurring a significant CPU penalty. The enhancements are:

- Increased buffer hit ratio without increasing CPU time
- Reduced I/O rate

[1] "Simple" means that the values of this example where simplified to decimal values (not hexadecimal) to give a better understanding of the technique.

IXFP/SnapShot Support for VSAM Datasets with VSE/ESA Version 2 Release 5

Author

Guenter Weigelt
VSE/VSAM Development, IBM Böblingen Laboratory
guenter_weigelt@de.ibm.com

... from the IBM Preview Software Announcement Letter, 200-031 (February 29, 2000), for VSE/ESA Version 2 Release 5:

> *Exploitation of IXFP/SnapShot: A new SNAP function will be added to IDCAMS that exploits the fast duplication of selected disk volumes.*
>
> *Enhancements to VSE/VSAM Backup/Restore will allow customers to back up selected VSAM datasets from "snapped" volumes. Benefits include improved availability and operations resulting from the ability to concurrently archive VSAM files while the production system is online.*

What Is IXFP/SnapShot?

RAMAC Virtual Array Storage (RVA) architecture enables you to produce copies of volumes via SnapShot. Snapping can take just seconds rather than hours. As far as the operating system is concerned, the snap is a real copy of the data. As far as the RVA hardware is concerned, it is a virtual copy of the data.

Why Is It Hard to Use IXFP/SnapShot for VSE/VSAM Datasets?

1. At DEFINE time of a VSAM dataset, all information about the dataset is saved in the catalog (dataset name, dataset attributes, VOLIDs, extent information, etc.).
2. Any access to a VSAM datasets is directed via:
   a. the name of its owning catalog
   b. the VOLID of the affected DASD volume, found in the catalog
   c. the name of the dataset, found in the catalog
   d. the extents of the dataset, found in the catalog.
   e. certain dataset attributes ("indexed", etc.)

Or in other words:

1. VSAM dataset names, VOLIDs and extent information are saved in the catalog.
2. Access to a VSAM dataset requires the correct names of the catalog, the correct name of the dataset and the correct VOLID.

and the general rules on a VSE system are:

- There **MUST NOT BE** duplicate catalog names on one VSE system.
- There **MUST NOT BE** duplicate VSAM dataset names in one catalog.
- There **MUST NOT BE** duplicate disk VOLIDs with VSAM objects on one VSE system.

IXFP/SnapShot for VSE/ESA before VSE/ESA 2.5 could not be used for VSAM datasets, because:

1. Dataset extents are hidden in the catalog.
   → SnapShot of files or extents is impossible.
2. All source VOLIDs regarding the datasets are hidden in the catalog and do not correspond to the ("snapped") target VOLIDs.
   → Access to VSAM datasets on "snapped" volumes is impossible.
3. Locking of VSAM datasets would not work in case of duplicate volume mounted.

Advantage of Using IXFP SnapShot for VSAM Datasets with VSE/ESA 2.5

Use Synonym Backup!

What is "synonym backup"?

- A new Synonym List is used to route the VSAM open and the IDCAMS BACKUP read routines to the SnapShot (target) Volumes.
- With the exception of using the new Synonym List, the backup process remains unchanged. That means all features of IDCAMS BACKUP could be used.
- VSAM Open controls the "Synonym Connection" to the catalog and to the datasets. Backup produces a normal backup medium (tape, disk) for restore purposes.

Why Synonym Backup?

Normal applications or traditional backup programs cannot access snapped datasets because:

- either the disk VOLIDs will be duplicate or
- the changed disk VOLIDs do not correspond to the catalog information.

Synonym Backup programs can access snapped datasets.

Under consideration of all the facts described above, a traditional backup of (or access to) VSAM datasets on "snapped" disk volumes would require:

1. **either** *unchanged disk VOLIDs* for the "snapped" disk volumes (but this is not possible in a VSE system, because *no duplicate VOLIDs* are allowed!)

   Duplicate VOLIDS would result in unpredictable access/results.
2. **or** *changed VOLIDs and lots of updates* to the "snapped" catalog to change internally:
   - all disk volume information (VOLIDs) in the catalog regarding the catalog and all datasets
   - all names of the catalog itself (catalog, data, index)
   - all index entries/pointers regarding the changed names and VOLIDs

   That is error prone and requires additional time effort. If one of the catalog changes fails, all data may be lost.

Solution: Run the Synonym Backup

The Synonym Backup does not require any changes to the catalog.

- No risk of losing data
- Faster access to the "snapped" datasets
- Contents of the target disk volumes stays identical to the source volumes at snap time (possibly may be helpful for disaster recovery)

How to achieve access to the "snapped" datasets:

1. **Use the new IDCAMS command SNAP.**

   Do SnapShot of all entire disk volumes where the catalog and all its datasets reside. Give the target disks different VOLIDs than the source disks. The target disk volumes will be online after SnapShot for further access.

2. **Use IDCAMS IMPORT CONNECT.**

   Import Connect the target catalog with a new name to the master catalog (pointing to the target catalog's disk volume, the snapped one).

   Notes:

   a. The "imported" catalog and its datasets are available, but NOT accessible by normal applications.

   b. A snapped Master Catalog could be handled by Backup like a User Catalog.

3. **Run the new Synonym Backup controlled by the Synonym List.**

   New IDCAMS BACKUP parameters control the Synonym Backup. The following Example will show how to do it.

Example of IDCAMS SNAP and Synonym BACKUP Job

The following example shows how to do a SnapShot from the disk volumes *SYSWK1* and *SYSWK2* to the disk volumes *VOLSN1* and *VOLSN2* and prepare/run the Synonym Backup.

1. The User catalog "VSESP.USER.CATALOG" on SYSWK1 will be "copied" via SnapShot to VOLSN1. The VSAM datasets on DOSRES and SYSWK1 will be "copied" to VOLSN1 and VOLSN2.
2. Import Connect will prepare the synonym catalog for further Backup. The copy of "VSESP.USER.CATALOG " will get the new name "VSESP.SNAP.CATALOG", which has to be the job catalog of the Synonym Backup.
3. Backup the snapped datasets, using the synonym catalog as job catalog.

Input:

```
// JOB SNAP and BACKUP from snapped Volumes
// ASSGN SYS005,180
// DLBL IJSYSUC,'VSESP.SNAP.CATALOG',,VSAM
// EXEC IDCAMS,SIZE=AUTO
 /* First: do the SNAPSHOT */       -
  SNAP
     SOURCEVOLUMES(SYSWK1,SYSWK2)  -
     TARGETVOLUMES(VOLSN1,VOLSN2)
 /* Second: Synonym Name for the snapped Catalog */-
  IMPORT CONNECT OBJECTS((VSESP.SNAP.CATALOG       -
     VOLUMES(VOLSN1) DEVT(3390)))                  -
     CATALOG(VSAM.MASTER.CATALOG)
 /* Third: Backup from snaped volumes */-
  BACKUP (*)                            -
     SYNONYMLIST(                       -
     SOURCEVOLUMES(SYSWK1,SYSWK2)       -
     TARGETVOLUMES(VOLSN1,VOLSN2)       -
     CATALOG(VSESP.USER.CATALOG)        -
     SYNCATALOG(VSESP.SNAP.CATALOG)     )
/*
/&
```

Output: Output of the Synonym Backup are normal backup media (tape or disk), like any other normal IDCAMS BACKUP output.

Some Background Information

1. Why do a SnapShot of full disk volumes?

   A backup of snapped VSAM datasets requires the availability of the catalog AND ALL of its related disk volumes keeping data of these VSAM datasets. That means all disk volumes keeping VSAM datasets owned by the catalog should be available.

2. Are VSAM data on "snapped" disk volumes safe?

   The target ("snapped") catalog remains unchanged; and, of course, all "snapped" datasets are "frozen". Standard applications can access only the SOURCE disk volumes. That means, the snapped catalog is not available for normal VSAM access. "Snapped" disk volumes are safe!

Performance Improvements

1. After SnapShot of all affected disk volumes, backup processing could be started immediately and can run during online processing.
2. During the very short "copy-time" of SnapShot (may be minutes or just seconds!), all online systems have to be shut down only for a very short time.
3. All following backup batch jobs are handling frozen data on the snapped volumes and are independent from any changes of data on the online systems.

IXFP/SnapShot with Synonym Backup will reduce the "Backup Window" significantly.

No catalog changes on the "snapped" catalog

No error-prone and time-consuming catalog changes are required on the saved ("snapped") catalog. The saved ("snapped") catalogs remains 100% unchanged.

The saved catalogs and the datasets could be used for disaster recovery (e.g. to replace one or more complete disk volumes).

Programming Requirements

The use of IDCAMS SNAP and Synonym Backup requires IXFP/SnapShot for VSE/ESA, 5686-066.

REXX Socket Programming

Author

Ursula Braun-Krahl
REXX/VSE Development, IBM Böblingen Laboratory
braunu@de.ibm.com

Interprocess communication within a computer or between computers is based on protocols. One concept for interprocess communication is the *socket*. It is the end point of a communication path.

The socket API is a low-level, general, and flexible application programming interface, very popular especially within the TCP/IP transport protocol. It allows to write communicating applications that receive data from a network and send data to a network.

The socket API is based on an *open/close/read/write* paradigm. Thus network communication is handled similar to reading from and writing to any other device.

All implementations of the socket programming interface are based on the original Berkeley Software Distribution (BSD) socket implementation with its roots in the UNIX environment.

TCP/IP Concepts

To understand socket programming, some basic TCP/IP concepts are listed here and explained shortly. For a more detailed explanation, one of the members of the rich set of TCP/IP books is recommended.

TCP/IP Protocols

A protocol is a set of rules or standards that two entities must follow to exchange and interpret messages. For communication via a network, TCP/IP defines a protocol stack consisting of different layers together with different protocols assigned to these layers. Famous protocols are:

IP - Internet Protocol

The IP layer provides the basic packet delivery services, but does not guarantee error-free arrival of IP packets in a determined order.

TCP - Transmission Control Protocol

TCP is a connection-oriented transport protocol that provides a reliable, full-duplex byte stream. It is used by the majority of TCP/IP applications.

UDP - User Datagram Protocol

UDP is a connectionless protocol that provides datagram services. It does not guarantee safe arrival of UDP datagrams. Such reliability features must be built into UDP applications.

IP Addresses

An IP address identifies the machine within a network. In the current version, it occupies 32 bits and is usually expressed externally in dotted decimal form, for instance "9.164.182.254".

Host name

In addition to the number form of IP Addresses, TCP/IP provides facilities to assign a symbolic name to an IP address, known as *host name*.

They are resolved in either some local host tables or via a special name server.

Ports A port number identifies a specific application on a given TCP/IP machine. It is a 16-bit integer ranging from 0 to 65534. A client application must know the port number of a server application to be able to contact it. Ports 0 to 1023 are reserved for well-known services. Ports 1024 to 5000 are used by TCP/IP for automatic assignments. Server applications should use port numbers above 5000.

Sockets The end point of a communication link between two application ports is uniquely identified by 3 components making up a *socket*

- protocol (TCP, UDP, IP)
- local IP-address
- local port

A *socket descriptor* or *socket number* is a 2-byte integer that acts as an index into a table of currently allocated sockets. Thus it represents the socket.

Socket Types

Three different socket types are defined:

Stream socket

connection-oriented (i.e. logical connection is established before data exchange), full-duplex, reliable, virtually unlimited size of byte streams, flow-control, TCP as default protocol, FTP as sample application

Datagram socket

connection-less (i.e. each datagram must contain the full set of addressing information for its delivery), unreliable, no flow-control, maximum size for a single datagram, UDP as default protocol, NFS as sample application

Raw socket

connection-less, unreliable, maximum size for messages, IP as default protocol, PING as sample application, not supported in REXX/VSE

Addressing Families

A socket identifies the address of a specific process at a specific computer using a specific transport protocol. The exact syntax of a socket address depends on the protocol being used, on its *addressing family* The only addressing family used in VSE is:

AF_INET

Addressing family Internet

Typical Socket Calls

A basic connection-oriented protocol like TCP consists of the following sequence of socket calls:

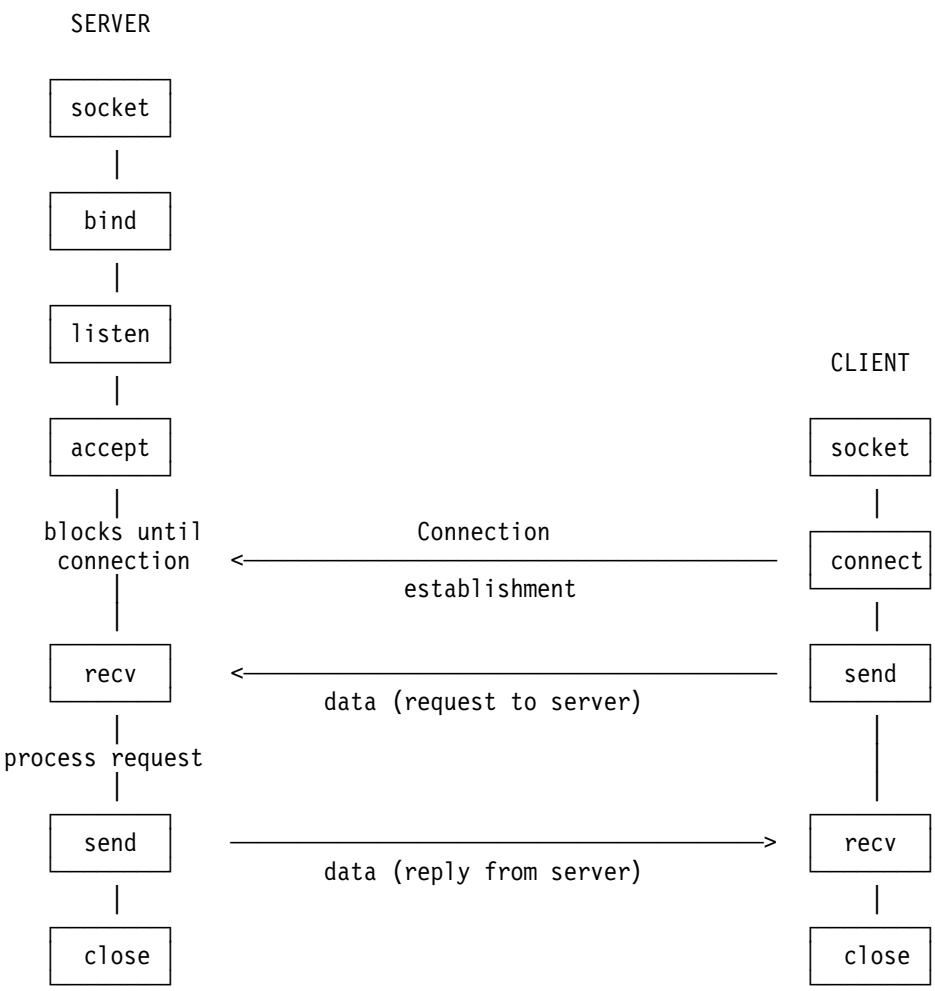


Figure 20. Connection-Oriented Protocol

A basic connection-less protocol like UDP consists of the following sequence of socket calls:

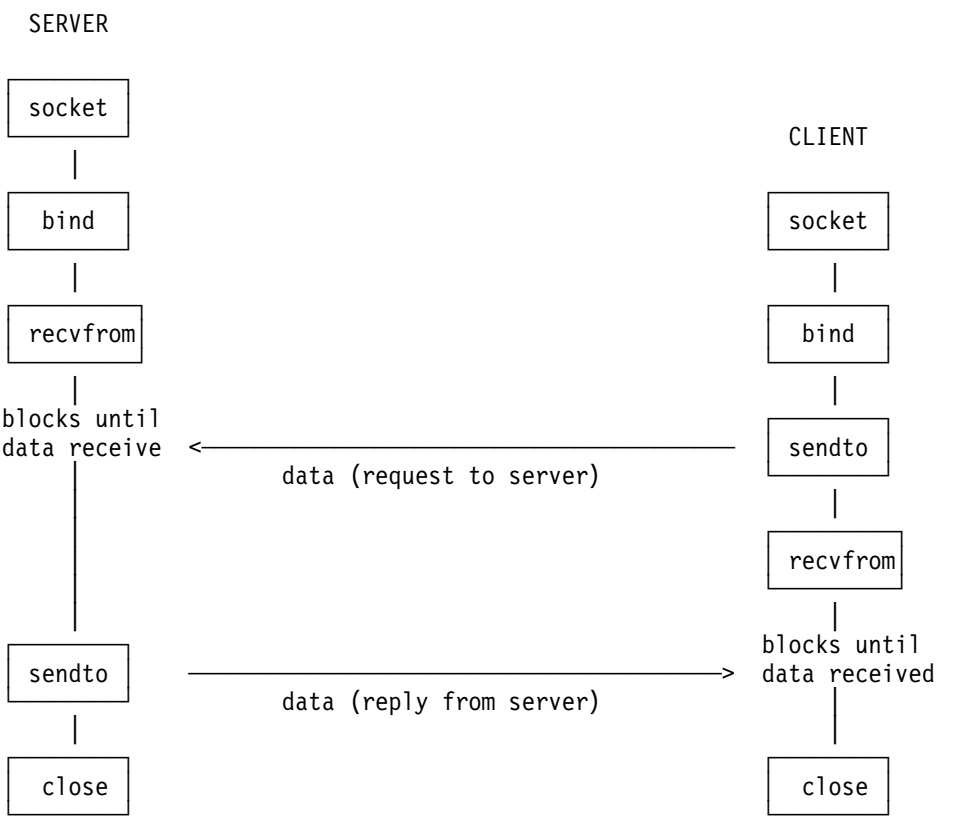
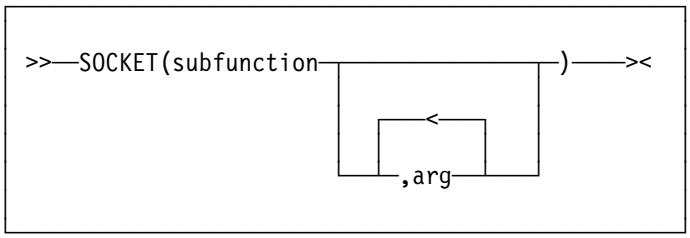


Figure 21. Connectionless Protocol

REXX Sockets Application Program Interface

With year-end 1999, REXX/VSE provided a REXX Socket API within PTF UQ37224 for VSE/ESA 2.3 and UQ37225 for VSE/ESA 2.4 (APAR PQ31258). It allows you to write socket applications in REXX/VSE for the TCP/IP environment. The REXX Socket Interface (i.e. a new function called "SOCKET") has the same syntax and semantic as the SOCKET function available for REXX/VM and REXX in OS/390.



subfunction
name of the socket call

arg
parameters for the socket call

A character string is returned from the SOCKET call that contains several values separated by blanks, so the string can be parsed using REXX. The first value in the string is usually the return code. If the return code is zero, the values following

the return code are returned by the socket subfunction. If the return code is not zero, the second value is the name of an error, and the rest of the string the corresponding error message.

Here are some examples:

```
SOCKET('GetHostId')    ==>  '0 9.4.3.2'
SOCKET('Recv',socket)  ==>  '1102 EWOULDBLOCK Problem on non-blocking socket'
```

The REXX/VSE SOCKET implementation uses the LE/VSE support to access the TCP/IP socket interface. Thus it maps the socket calls from the C programming language to REXX/VSE.

Tasks You Can Perform Using REXX Sockets

You can use REXX sockets to perform the following tasks:

1. Processing socket sets

   A socket set is a number of preallocated sockets available to a single application.

Table 2. REXX Socket Functions for Processing Socket Sets

Function	Purpose
Initialize	Defines a socket set
Terminate	Closes all the sockets in a socket set and releases the socket set
SocketSet	Gets the name of the active socket set
SocketSetList	Gets the names of all the available socket sets
SocketSetStatus	Gets the status of a socket set

2. Creating, connecting, changing, and closing sockets

   A socket is an end point for communication that can be named and addressed in a network.

Table 3. REXX Socket Functions for Creating, Connecting, Changing, and Closing Sockets

Function	Purpose
Socket	Creates a socket in the active socket set
Bind	Assigns a unique local name (network address) to a socket
Listen	Converts an active stream socket to a passive socket
Connect	Establishes a connection between two stream sockets
Accept	Accepts a connection from a client to a server
Shutdown	Shuts down a duplex connection
Close	Shuts down a socket

3. Exchanging data

   You can send and receive data on connected stream sockets and on datagram sockets.

Table 4. REXX Socket Functions for Exchanging Data

Function	Purpose
Read	Reads data on a connected socket
Write	Writes data on a connected socket
Recv	Receives data on a connected socket
Send	Sends data on a connected socket
RecvFrom	Receives data on a socket and gets the sender's address
SendTo	Sends data on a socket, and optionally specifies a destination address

4. Resolving names and other identifiers

   You can get information such as name, address, client identification, and host name. You can also resolve an Internet Protocol address (IP address) to a symbolic name of a symbolic name to an IP address.

Table 5. REXX Socket Functions for Resolving Names and Other Identifiers

Function	Purpose
GetHostId	Gets the IP address for the host processor
GetHostName	Gets the name of the host processor
GetSockName	Gets the local name to which a socket was bound
GetHostByAddr	Gets the host name for an IP address
GetHostByName	Gets the IP address for a host name
Resolve	Resolves the host name through a name server

5. Managing configurations, options, and modes

   You can obtain the version number of the REXX Sockets function package, get socket options, set socket options, and set the mode of operation.

Table 6. REXX Socket Functions for Managing Configurations, Options and Modes

Function	Purpose
Version	Gets the version and date of the REXX Sockets function package
Select	Monitors activity on selected sockets
GetSockOpt	Gets the status of options for a socket
SetSockOpt	Sets options for a socket
Fcntl	Sets or queries the mode of a socket
Ioctl	Controls the operating characteristics of a socket

Complete reference material for the new REXX/VSE Socket function is on the VSE/ESA home page:

http://www.ibm.com/s390s/vse/vsehtmls/vserxsoc.htm

REXX Samples Using the Socket Function

The following samples show how to make use of the REXX/VSE Socket function to establish and carry out (network) communication based on sockets. They demonstrate the simplicity of socket programming in REXX/VSE.

Two pairs of communicating REXX Socket programs are provided:

- SERVMIRR - CLIEMIRR : demo of basic socket concepts
- SERVSAMP - CLIESAMP : comprehensive demo of REXX socket programming

Source code for downloading as well as more samples will be available via the REXX/VSE home page:

http://www.ibm.com/s390/vse/rexxhome.htm

Here you'll also find counterparts to a REXX/VSE socket program that are written in Java for REXX for Windows NT/95. Thus they run on a different platform and communicate with the REXX/VSE socket program on a VSE/ESA host.

Even though the servers and clients shown here are coded in REXX/VSE, this does not mean that server and client have to run both on a VSE platform. A REXX/VSE socket program can communicate with any socket program written in any language supporting the socket API on any platform supporting TCP/IP, as long as the counterpart program uses the corresponding sequence of socket calls.

Sending and Receiving Data: Simple Server to Mirror Data

The mirror server receives a string from the client, reverses the order of the characters in the string and returns it to the client. After returning the mirrored string, communication is closed.

```
/*********************************************************************/
/***** SERVMIRR:                                                 ******/
/*********************************************************************/
/***** REXX program for a socket server procedure receiving     ******/
/***** data from a client, reversing the data, and sending      ******/
/***** the data back to the client.                             ******/
/*********************************************************************/

/***** initializations ************************************** ******/
rc = 0
trace off

/***** initialize socketset ********************************** ******/
fc = SOCKET('INITIALIZE','SERVMIRR')
parse var fc socket_rc .
if socket_rc ¬= 0
then do
  say 'INITIALIZE failed with return info ' fc
  exit 99
end
```

Figure 22 (Part 1 of 4). Server to Mirror Data

```
/***** create a TCP socket for client connection requests ***** ******/
fc = SOCKET('SOCKET','AF_INET','SOCK_STREAM','IPPROTO_TCP')
parse var fc socket_rc newsocketid
if socket_rc ¬= 0
then do
  say 'SOCKET failed with return info ' fc
  fc = SOCKET('TERMINATE')
  exit 99
end

/***** bind socket to well known port 1996 ******************** ******/
fc = SOCKET('BIND',newsocketid,'2 1996 9.164.155.114')
parse var fc bind_rc rest
if bind_rc ¬= 0
then do
  say 'BIND failed with return info ' fc
  fc = SOCKET('CLOSE',newsocketid)
  fc = SOCKET('TERMINATE')
  exit 99
end

/***** create a connection queue for a client ***************** ******/
fc = SOCKET('LISTEN',newsocketid,'10')
parse var fc listen_rc rest
if listen_rc ¬= 0
then do
  say 'LISTEN failed with return info ' fc
  fc = SOCKET('CLOSE',newsocketid)
  fc = SOCKET('TERMINATE')
  exit 99
end

/***** wait for a client to connect ************************** ******/
fc = SOCKET('ACCEPT',newsocketid)
parse var fc accept_rc rest
if accept_rc ¬= 0
then do
  say 'ACCEPT failed with return info ' fc
  fc = SOCKET('CLOSE',newsocketid)
  fc = SOCKET('TERMINATE')
  exit 99
end
parse var rest accepted_socket accept_socket_address
say "Client has established connection."
```

Figure 22 (Part 2 of 4). Server to Mirror Data

```
/***** we don't want any more clients, close request socket *** ******/
fc = SOCKET('CLOSE',newsocketid)
parse var fc close_rc rest
if close_rc ¬= 0
then do
  say 'CLOSE failed with return info ' fc
  exit 99
end

/***** read string from client ****************************** ******/
fc = SOCKET('READ',accepted_socket,'10000')
parse var fc read_rc num_read_bytes read_string
if read_rc ¬= 0
then do
  say 'READ failed with return info ' fc
  rc = 99
  signal SHUTDOWN_LABEL
end
say "String read from client: '" read_string "'"

/***** reverse string from client *************************** ******/
send_string = Reverse(read_string)

/***** send string back to client *************************** ******/
fc = SOCKET('SEND',accepted_socket,send_string,'')
parse var fc send_rc num_sent_bytes
if send_rc ¬= 0
then do
  say 'SEND failed with return info ' fc
  rc = 99
  signal SHUTDOWN_LABEL
end
/***** plausibility check *********************************** ******/
if num_read_bytes ¬= num_sent_bytes
then do
  say 'number of sent bytes does not match number of read bytes'
  rc = 99
  signal SHUTDOWN_LABEL
end
```

Figure 22 (Part 3 of 4). Server to Mirror Data

```
/***** close client socket ********************************** ******/
SHUTDOWN_LABEL:
fc = SOCKET('CLOSE',accepted_socket)
parse var fc close_rc rest
if close_rc ¬= 0
then do
  say 'CLOSE failed with return info ' fc
  fc = SOCKET('TERMINATE')
  exit 99
end

/***** terminate  socketset ********************************** ******/
fc = SOCKET('TERMINATE')
exit rc
```

Figure 22 (Part 4 of 4). Server to Mirror Data

A socket set must be initialized first before using any other REXX Socket function. Then a stream-type socket is created and bound to the well-known port number 1996 and the server IP address. Clients that want to connect to this server have to specify this port number and IP address.

With socket function LISTEN request, queues for possibly connecting clients are established; and afterwards the mirror server is prepared to accept its first client connection request. ACCEPT blocks the server until the first client starts connecting. It returns a new socket id to be used to communicate with the client. Connection of further clients may still be handled via further ACCEPT invocations using the original socket ID. Here it is not intended to support more clients. Thus the original socket is closed.

The client string is read, reversed and given back to the client. Afterwards, the server closes communication and terminates the whole socketset.

Sending and Receiving Data: Simple Client to Have Data Mirrored

A REXX client program able to talk to the previous mirror server begins on the next page:

```
/*********************************************************************/
/***** CLIEMIRR:                                                 ******/
/*********************************************************************/
/***** REXX program for a socket client procedure sending        ******/
/***** data to a server and receiving the reversed data from     ******/
/***** the server.                                               ******/
/*********************************************************************/

/***** initializations ************************************** ******/
rc = 0
trace off

/***** string to be reversed is given as parameter ************ ******/
arg read_string

/***** initialize socketset ********************************** ******/
fc = SOCKET('INITIALIZE','CLIEMIRR')
parse var fc socket_rc .
if socket_rc ¬= 0
then do
  say 'INITIALIZE failed with return info ' fc
  exit 99
end

/***** create a TCP socket *********************************** ******/
fc = SOCKET('SOCKET','AF_INET','STREAM','TCP')
parse var fc socket_rc newsocketid
if socket_rc ¬= 0
then do
  say 'SOCKET failed with return info ' fc
  fc = SOCKET('TERMINATE')
  exit 99
end

/***** connect new socket to the specified server ************* ******/
fc = SOCKET('CONNECT',newsocketid,'AF_INET 1996 9.164.155.114')
parse var fc connect_rc rest
if connect_rc ¬= 0
then do
  say 'CONNECT failed with return info ' fc
  rc = 99
  signal SHUTDOWN_LABEL
end
```

Figure 23 (Part 1 of 2). Client to Have Data Mirrored

```
/***** send string to the mirror server ********************** ******/
fc = SOCKET('SEND',newsocketid,read_string,'')
parse var fc send_rc num_sent_bytes
if send_rc ¬= 0
then do
  say 'SEND failed with return info ' fc
  rc = 99
  signal SHUTDOWN_LABEL
end

/***** plausibility check ********************************** ******/
if length(read_string) ¬= num_sent_bytes
then do
  say 'number of sent bytes does not match number of read bytes'
  rc = 99
  signal SHUTDOWN_LABEL
end

/***** receive answer from mirror server ********************* ******/
fc = SOCKET('READ',newsocketid,'10000')
parse var fc read_rc num_read_bytes received_string
if read_rc ¬= 0
then do
  say 'READ failed with return info ' fc
  rc = 99
  signal SHUTDOWN_LABEL
end
say "String '" read_string "' was mirrored to: '" received_string "'"

/***** close the socket ************************************ ******/
SHUTDOWN_LABEL:
fc = SOCKET('CLOSE',newsocketid)
parse var fc close_rc rest
if close_rc ¬= 0
then do
  say 'CLOSE failed with return info ' fc
  fc = SOCKET('TERMINATE')
  exit 99
end

/***** terminate  socketset ********************************** ******/
fc = SOCKET('TERMINATE')
exit rc
/*********************************************************************/
```

Figure 23 (Part 2 of 2). Client to Have Data Mirrored

After socket set initialization, the stream-type socket is created and connected to the well-known port and server address. Using the established connection, the given input string is sent to the server and the following READ blocks the client until the server has sent an answer. Then this specific socket communication is closed first, and afterwards the whole REXX-Socket communication setup is terminated.

Server for VSE Files and Console Commands

A more complex, but also more meaningful, REXX Socket sample follows. It is able to handle more than one client connected to the server at the same time. After receiving a retrieval command, it transfers library members, VSAM files, POWER queue entries, or console command responses back to its clients.

```
/*********************************************************************/
/*- SERVSAMP -- Example demonstrating the usage of REXX Sockets -----*/
/*********************************************************************/
/***** REXX program for a socket server procedure waiting for   ******/
/***** clients to connect, receiving one of the commands:       ******/
/*****    READLIB   -  retrieve a library member                ******/
/*****    READVSAM  -  retrieve a VSAM file                     ******/
/*****    READPOW   -  retrieve a POWER queue entry             ******/
/*****    console command  -  execute                           ******/
/***** and sending the resulting library member, VSAM file,     ******/
/***** POWER queue entry or console command response back to    ******/
/***** the client.                                              ******/
/*********************************************************************/

/***** initializations ************************************** ******/
/***** Port can be specified as argument. Default port is 5678. ******/
  trace o
  Parse Arg Port
  If Port = "" Then
    Port = 5678

/***** Open socket at well known port and wait for clients **** ******/
  SocketNr = ListenPort(Port)
  If SocketNr = -1 Then Do
    Call Socket 'Terminate'
    Exit 1
  End

/***** Close the socket when program is interrupted *********** ******/
  Signal On Halt
  Call Opermsg 'On'
  timeout = 15
  fc = ASSGN('STDOUT','SYSLOG')
  Say "Enter 'MSG " || SYSPID ||",DATA=HI' to exit program."
  fc = ASSGN('STDOUT','SYSLST')
```

Figure 24 (Part 1 of 14). Server for VSE Files and Console Commands

```
/***** Check for new events: ********************************** ******/
/*****   A new client may want to start communication with this ******/
/*****   server, or an existing client may want to sent data,   ******/
/*****   or the timeout interval has finished without any       ******/
/*****   special event.                                         ******/
  Do Forever
    sockevtlist = 'Read * Write Exception'
    parse value Socket('Select',sockevtlist,timeout) with rc socknum sellist
    If rc /= 0 Then
      Do
        Say "??? Select failed ???"
        Call Close SocketNr
        Return -1
      End
    parse upper var sellist . 'READ' orlist 'WRITE' . 'EXCEPTION' .
    If orlist /= '' Then Do
      event = 'SOCKET'
      parse var orlist orsocket .
      rest = 'READ' orsocket
    End
    Else event = 'TIME'
```

Figure 24 (Part 2 of 14). Server for VSE Files and Console Commands

```
    Select

      /* Accept connections from clients */
      When event = 'SOCKET' Then Do
        parse var rest keyword ts .

        /* Accept new connections from clients */
        if keyword = 'READ' & ts=SocketNr Then Do
          /* wait for client to connect and start handler    */
          Say "Waiting for client to connect."
          Parse Value Socket('Accept',SocketNr) ,
            with rc ClientSocket NetworkAddress
          Say "Client connected"
        End

        /* Receive Command */
        if keyword = 'READ' & ts/=SocketNr Then Do
          Command = ReceiveRequest(ts)
          If Command = 'QUIT' Then Do
            Call Close ts
            Say 'SERVSAMP: Disconnected socket' ts
          End
          Else Do
          End
        End

      End /* When */

      When event = 'TIME' Then Do
        Say "Timeout occurred"
      End

      /* Unknown event */
      Otherwise Nop
    End /* Select */
  End /* Do Forever */

  Halt:
  Call Close SocketNr
  Call Socket 'Terminate'
  Exit
```

Figure 24 (Part 3 of 14). Server for VSE Files and Console Commands

```
/*********************************************************/
/*                                                       */
/* Procedure: Close                                      */
/* Purpose:   Close the specified socket.                */
/* Arguments: Socket - active socket number              */
/* Returns:   nothing                                    */
/*                                                       */
/*********************************************************/
Close: Procedure
  Parse Arg SocketNr
  Call Socket 'Close',SocketNr
  Return

/*********************************************************/
/*                                                       */
/* Function:  ListenPort                                 */
/* Purpose:   Create a socket, bind it to a port and     */
/*            listen at the port for connecting clients.*/
/* Arguments: Port - port number                         */
/* Returns:   Socket number if successful, -1 otherwise */
/*                                                       */
/*********************************************************/
ListenPort: Procedure
  Parse Arg Port

  /* initialize socketset                               */
  parse value Socket('Initialize','SERVSAMP') ,
    with rc .
  If rc /= 0 Then
    Do
      Say "Unable to initialize socketset"
      Return -1
    End

  /* create a TCP socket                                */
  parse value Socket('Socket','AF_INET','Sock_stream','0') ,
    with rc SocketNr
  If rc /= 0 Then
    Do
      Say "Unable to create socket"
      Return -1
    End
```

Figure 24 (Part 4 of 14). Server for VSE Files and Console Commands

```
/* find out local IP address and bind socket to port  */
parse value Socket('GetHostId') with rc IpAddr
Host = "AF_INET" Port IpAddr
rc = Socket('Bind',SocketNr,Host)
if rc /= 0 Then
  Do
    Say "Unable to bind to port:" Port
    Call Close SocketNr
    Return -1
  End

/* listen at the port, allow 5 clients in queue       */
rc = Socket('Listen',SocketNr,5)
if rc /= 0 Then
  Do
    Say "Unable to listen at port:" Port
    Call Close SocketNr
    Return -1
  End

Return SocketNr
```

Figure 24 (Part 5 of 14). Server for VSE Files and Console Commands

```
/*******************************************************/
/*                                                     */
/* Function:  ReceiveRequest                           */
/* Purpose:   Wait for a command from the client and   */
/*            execute it. Return the identifier of the */
/*            command to the caller.                   */
/* Arguments: Socket - active socket number            */
/* Returns:   command identifier                       */
/*                                                     */
/*******************************************************/
ReceiveRequest: Procedure
  Parse Arg SocketNr

  /* wait for the command from the client              */
  parse value Socket('Recv',SocketNr,1024) ,
    with rc BytesRcvd CommandLine

  Say "Command line from client:" CommandLine

  CommandLine = Translate(CommandLine)
  If rc /= 0 Then
    CommandLine = "QUIT"
  Select;
  When Word(CommandLine,1) = "QUIT" Then
    Nop
  When Word(CommandLine,1) = "READLIB" Then
    Call ProcessLib SocketNr, Commandline
  When Word(CommandLine,1) = "READVSAM" Then
    Call ProcessVsam SocketNr, Commandline
  When Word(CommandLine,1) = "READPOW" Then
    Call ProcessPow  SocketNr, Commandline
  Otherwise Call ProcessCommand SocketNr, Commandline
  End; /* Select */

  /* send end of answer marker back to client */
  Call Socket 'Send',SocketNr,">>>End_of_transmission<<<"
  Return CommandLine
```

Figure 24 (Part 6 of 14). Server for VSE Files and Console Commands

```
/*******************************************************/
/*                                                     */
/* Procedure: Answer                                   */
/* Purpose:   Send one answer line back to the client  */
/*            and wait for acknowledgement from client.*/
/* Arguments: Socket - active socket number            */
/*            AnswerString - line to send to client    */
/* Returns:   nothing                                  */
/*                                                     */
/*******************************************************/
Answer: Procedure
  Parse Arg SocketNr, AnswerString
  Call Socket 'Send',SocketNr,AnswerString
  Call Socket 'Recv',SocketNr,256
  Return

/*******************************************************/
/*                                                     */
/* Procedure: AnswerQueue                              */
/* Purpose:   Send all lines contained in AnswerLines. */
/*            back to the client as the answer of the  */
/*            previous executed command.               */
/* Arguments: Socket - active socket number            */
/* Returns:   nothing                                  */
/*                                                     */
/*******************************************************/
AnswerQueue: Procedure Expose AnswerLines.
  Parse Arg SocketNr

  /* send answer lines until session queue is empty    */
  Do I = 1 to AnswerLines.0

    /* empty lines will be sent as space               */
    If AnswerLines.I = "" Then
      AnswerLines.I = " "

    Call Answer SocketNr, AnswerLines.I
  End
  Drop AnswerLines.
  Return
```

Figure 24 (Part 7 of 14). Server for VSE Files and Console Commands

```
/*******************************************************/
/*                                                     */
/* Procedure: ProcessCommand                           */
/* Purpose:   Process the console command that was     */
/*            received from the client and send back   */
/*            the result.                              */
/* Arguments: Socket - active socket number            */
/*            CommandLine - console command            */
/* Returns:   nothing                                  */
/*                                                     */
/*******************************************************/
ProcessCommand: Procedure
  Parse Arg SocketNr, CommandLine
  Call SYSVAR SYSPID
  carToken = 'SERVER' || SYSPID

  /* activate console session                         */
  ADDRESS CONSOLE 'ACTIVATE NAME' carToken 'PROFILE REXNORC'
  ADDRESS CONSOLE 'CART' carToken

  /* issue the command */
  ADDRESS CONSOLE CommandLine

  /* get the command result */
  fc = GETMSG('AnswerLines.','RESP',carToken,,15)

  /* deactivate console session                       */
  ADDRESS CONSOLE 'DEACTIVATE' carToken

  if AnswerLines.0 = 0 Then
   Do
     AnswerLines.0 = 1
     AnswerLines.1 = '??? no result ???'
   End

  Call AnswerQueue SocketNr
  Return
```

Figure 24 (Part 8 of 14). Server for VSE Files and Console Commands

```
/*******************************************************/
/*                                                     */
/* Procedure: ProcessLib                               */
/* Purpose:   Transfer the library member back to the   */
/*            client.                                  */
/* Arguments: Socket - active socket number            */
/*            CommandLine - READLIB request            */
/* Returns:   nothing                                  */
/*                                                     */
/*******************************************************/
ProcessLib: Procedure
  Parse Arg SocketNr, CommandLine

  Parse Var   CommandLine ,
        "READLIB " lib "." slib "." memname "." memtype .

  /* check for existence of the desired library member */
  Call REXXIPT 'libr_in.'
  Call OUTTRAP 'libr_out.',,'NOCONCAT'
  libr_in.0 = 2
  libr_in.1 = 'SEARCH ' || memname || '.' || memtype || ,
              ' SUBLIB=' || lib || '.' || slib
  libr_in.2 = '/*'
  ADDRESS LINK 'LIBR'
```

Figure 24 (Part 9 of 14). Server for VSE Files and Console Commands

```
If RC = 0
Then Do
  /* read the desired library member */
  libr_in.0 = 3
  libr_in.1 = 'ACCESS SUBLIB=' || lib || '.' || slib
  libr_in.2 = 'LIST ' memname || '.' || memtype
  libr_in.3 = '/*'
  ADDRESS LINK 'LIBR'
  If RC = 0
  Then Do
    Do I=1 to libr_out.0-7
      AnswerLines.I = value('libr_out.' || I+6)
    End
    AnswerLines.0 = libr_out.0-7
  End
  Else Do
    AnswerLines.0 = 1
    AnswerLines.1 = '??? Problem reading library member ???'
  End
End

Else Do
  /* desired library member has not been found */
  AnswerLines.0 = 1
  If RC = 2
  Then
    AnswerLines.1 = '??? Library member not found ???'
  Else
    AnswerLines.1 = '??? Problem accessing library member ???'
End

if AnswerLines.0 = 0 Then
 Do
   AnswerLines.0 = 1
   AnswerLines.1 = '??? no result ???'
 End

Call AnswerQueue SocketNr
Return
```

Figure 24 (Part 10 of 14). Server for VSE Files and Console Commands

```
/*********************************************************/
/*                                                       */
/* Procedure: ProcessVsam                                */
/* Purpose:   Transfer the VSAM file back to the client.*/
/* Arguments: Socket - active socket number              */
/*            CommandLine - READVSAM request             */
/* Returns:   nothing                                    */
/*                                                       */
/*********************************************************/
ProcessVsam: Procedure
  Parse Arg SocketNr, CommandLine
  Parse Var   CommandLine ,
        "READVSAM" filename catalog .
  If catalog = '' Then catalog = 'IJSYSCT'
  /* read the desired VSAM file */
  ADDRESS JCL
    "// DLBL PRINTFL,'" || filename || "',,VSAM,CAT=" || catalog
    "/*"
  Call REXXIPT 'idcams_in.'
  Call OUTTRAP 'idcams_out.',,'NOCONCAT'
  idcams_in.0 = 1
  idcams_in.1 = 'PRINT INFILE (PRINTFL) CHARACTER'
  ADDRESS LINK 'IDCAMS MARGINS(1 80)'
  If RC = 0
  Then Do
    Do I=1 to idcams_out.0-11
      AnswerLines.I = value('idcams_out.' || I+7)
    End
    AnswerLines.0 = idcams_out.0-11
  End
  Else Do
    AnswerLines.0 = 1
    AnswerLines.1 = '??? Problem reading VSAM file ???'
  End

  if AnswerLines.0 = 0 Then
   Do
     AnswerLines.0 = 1
     AnswerLines.1 = '??? no result ???'
   End

  Call AnswerQueue SocketNr
  Return
```

Figure 24 (Part 11 of 14). Server for VSE Files and Console Commands

```
/*********************************************************/
/*                                                       */
/* Procedure: ProcessPow                                 */
/* Purpose:   Transfer POWER queue entry back to the     */
/*            client.                                    */
/* Arguments: Socket - active socket number              */
/*            CommandLine - READPOW request              */
/* Returns:   nothing                                    */
/*                                                       */
/*********************************************************/
ProcessPow: Procedure
  Parse Arg SocketNr, CommandLine
  Parse Var   CommandLine ,
        "READPOW" queue entryname entrynum entryclass .
  if queue /= 'RDR' & queue /= 'LST' & queue /= 'PUN'
  Then Do
    AnswerLines.0 = 1
    AnswerLines.1 = '??? queue specification invalid ???'
    Call AnswerQueue SocketNr
    Return
  End
```

Figure 24 (Part 12 of 14). Server for VSE Files and Console Commands

```
Call OUTTRAP 'power_out.',,'NOCONCAT'
/* determine user of POWER queue entry */
ADDRESS POWER
'PDISPLAY ' || queue || ',' || entryname

Do I=2 To power_out.0
  If entrynum = ''
  Then Do
    entrynum = word(power_out.I,3)
  End
  If entryclass = ''
  Then Do
    entryclass = word(power_out.I,6)
  End
  If word(power_out.I,2) = entryname & ,
     word(power_out.I,3) = entrynum & ,
     word(power_out.I,6) = entryclass
  Then Do
    Parse Var power_out.I . 'FROM=(' from_user ')' .
    Parse Var power_out.I . 'TO=(' to_user ')' .
    Select
      When from_user /= '' Then setuid_user = from_user
      When to_user /= '' Then setuid_user = to_user
      Otherwise Do
        setuid_user = 'REXX'
        fc = SENDCMD('PALTER' queue || ',' || entryname || ,
                     ',' || entrynum || ',USER=' || setuid_user ,
                    )
      End
    End /* Select */
    Leave
  End
End
```

Figure 24 (Part 13 of 14). Server for VSE Files and Console Commands

```
/* read the desired POWER queue entry */
'SETUID' setuid_user
ADDRESS POWER 'GETQE' queue 'JOBNAME' entryname ,
                            'JOBNUM' entrynum ,
                            'CLASS' entryclass ,
                            'STEM AnswerLines.'
If RC = 0 & power_out.0 = 0
Then Do
  Nop
End
Else Do
  AnswerLines.0 = 1
  AnswerLines.1 = '??? Problem reading POWER queue entry ???'
End

if AnswerLines.0 = 0 Then
 Do
   AnswerLines.0 = 1
   AnswerLines.1 = '??? no result ???'
 End

Call AnswerQueue SocketNr
Return
```

Figure 24 (Part 14 of 14). Server for VSE Files and Console Commands

This server runs until an operator interrupts it via the console operator communication exit:

```
MSG <partition_id>,DATA=HI
```

Using the Socket SELECT function, the server determines whether a new client is starting communication or an existing client is issuing a new server request. Incoming requests that do not start with one of the keywords *QUIT*, *READLIB*, *READVSAM*, or *READPOW* are treated as console commands and given to the console router using the REXX console command environment.

To transfer a library member, the server checks first via LIBR SEARCH if the desired library member exists. If available, it moves its contents into a REXX stem variable and sends these lines to the client. To transfer a VSAM file, the server invokes IDCAMS PRINT to retrieve its data into a REXX stem variable, whose content is then sent to the client.

For POWER queue entries, the server determines existence and ownership of a requested entry using a PDISPLAY command. For entries without FROM- and TO-user assigned an artificial TO-user, REXX is defined to be authorized to access the POWER queue entry. Afterwards, the entry is retrieved via ADDRESS POWER GETQE into a REXX stem variable to be sent.

Client for VSE Files and Console Commands

A corresponding client for the previous REXX Socket sample is given here. It sends commands to the server requesting transfer of library members, VSAM files, POWER queue entries, or console command responses. For demonstration purposes, the received response is displayed at the console.

```
/*********************************************************************/
/*- CLIESAMP -- Example demonstrating the usage of REXX Sockets -----*/
/*********************************************************************/
/***** REXX program for a socket client procedure sending one   ******/
/***** of the commands                                          ******/
/*****    READLIB   -  retrieve a library member                ******/
/*****    READVSAM  -  retrieve a VSAM file                     ******/
/*****    READPOW   -  retrieve a POWER queue entry             ******/
/*****    console command  -  execute                           ******/
/***** to the server, and writing the data returned from the    ******/
/***** server to SYSLOG.                                        ******/
/*********************************************************************/

/***** initializations ************************************** ******/
/***** Server must be specified as argument.                  ******/
/***** Optionally a port can be specified in addition separated ******/
/***** by a colon. Default port is 5678                       ******/
  trace o
  Parse Arg Server
  CALL ASSGN 'STDIN','SYSLOG'              /* Input stream: SYSLOG  */
  CALL ASSGN 'STDOUT','SYSLOG'             /* Output stream: SYSLOG */

  /* check command line arguments, server is required */
  If Server = "" Then
   Do
     Say "Usage: CLIESAMP Servername"
     Say "  Servername may contain a port number separated with a colon."
     Exit 1
  End
```

Figure 25 (Part 1 of 4). Client for VSE Files and Console Commands

```
/* Connect to remote control server */
Socket = Connect(Server)
If socket = -1 Then Do
  Call Socket 'Terminate'
  Exit 1
End

/* loop until QUIT command was entered */
Do Until Command = "QUIT"
  Say "Please enter:"
  Say "     console command"
  Say "     'READLIB' lib.slib.memname.memtyp"
  Say "     'READVSAM' vsam.name catalog"
  Say "     'READPOW' RDR|LST|PUN name number class"
  Say "or   'QUIT'"
  Parse Pull CommandLine
  Parse Upper Var CommandLine Command Option
  If Length(Command) > 0 Then
    Call SendCommand Socket, CommandLine
End

/* Close connection to server */
Call Close Socket
Call Socket 'Terminate'
Exit
```

Figure 25 (Part 2 of 4). Client for VSE Files and Console Commands

```
/*******************************************************/
/*                                                     */
/* Function:  Connect                                  */
/* Purpose:   Create a socket and connect it to server. */
/* Arguments: Server - server name, may contain port no.*/
/* Returns:   Socket number if successful, -1 otherwise */
/*                                                     */
/*******************************************************/
Connect: Procedure
  Parse Arg Server

  /* if the servername has a port address specified     */
  /* then use this one, otherwise use the default port  */
  /* for the remote control server (5678)               */
  Parse Var Server Server ":" Port
  If Port = "" Then
    Port = 5678

  /* initialize a socketset                             */
  parse value Socket('Initialize','clielib') ,
    with rc .
  If rc /= 0 Then
    Do
      Say "Unable to initialize socketset"
      Return -1
    End

  /* create a TCP socket                                */
  parse value Socket('Socket','2','1','0') ,
    with rc SocketNr
  If rc /= 0 Then
    Do
      Say "Unable to create socket"
      Return -1
    End

  /* connect the new socket to the specified server     */
  Host = "AF_INET" Port Server
  rc = Socket('Connect',SocketNr,Host)
  if Words(rc) > 1 Then
    Do
      Say "Unable to connect to server:" Server
      Call Close SocketNr
      Return -1
    End

  Return SocketNr
```

Figure 25 (Part 3 of 4). Client for VSE Files and Console Commands

```
/*******************************************************/
/*                                                     */
/* Function:  SendCommand                              */
/* Purpose:   Send a command via the specified socket  */
/*            and display the full response from server */
/* Arguments: SocketNr - active socket number          */
/*            Command - command string                 */
/* Returns:   nothing                                  */
/*                                                     */
/*******************************************************/
SendCommand: Procedure
  Parse Arg SocketNr, Command

  /* send the command to the remote control server     */
  Call Socket 'Send', SocketNr, Command

  Do Forever
    parse value Socket('Recv',SocketNr,1024) ,
      with rc BytesRcvd RcvData

    /* error or end of response encountered            */
    If rc /= 0 | ,
       RcvData = ">>>End_of_transmission<<<" Then
      Leave

    /* display response and send acknowledge to server  */
    Say RcvData
    Call Socket 'Send', SocketNr, "OK!"
  End

  Say "----- end of output from command:" Command "-----"
  Return

/*******************************************************/
/*                                                     */
/* Function:  Close                                    */
/* Purpose:   Close the specified socket               */
/* Arguments: SocketNr - active socket number          */
/* Returns:   nothing                                  */
/*                                                     */
/*******************************************************/
Close: Procedure
  Parse Arg SocketNr
  Call Socket 'Close', SocketNr

  Return
/*********************************************************************/
```

Figure 25 (Part 4 of 4). Client for VSE Files and Console Commands

This client runs until command *QUIT* is entered at the console. All other console input is sent to the server, where it is either interpreted as some kind of file retrieval command (*READLIB*, *READVSAM*, or *READPOW*) or as console command.

Additional Information

Besides the complete reference material for the REXX/VSE Socket function on the VSE/ESA home page:

http://www.ibm.com/s390/vse/vsehtmls/vserxsoc.htm

you can find more information in:

- *TCP/IP for VSE/ESA User's Guide*, SC33-6601-01
- *MVS TCP/IP Sockets*, GG24-2561-00

  Even though the book is intended for MVS programmers, it contains lots of general information on socket programming valid also for VSE/ESA.
- TCP/IP for VSE/ESA home page
  http://www.ibm.com/s390/vse/vsehtmls/tcphome.htm
- REXX/VSE home page
  http://www.ibm.com/s390/vse/rexx/rexxhome.htm

If you have any questions or comments about this article or about REXX/VSE, please feel free to contact me using e-mail at **braunu@de.ibm.com**.

Security In VSE/ESA 2.4 – What's New?

Author

Helmut Hellner
VSE Development, IBM Böblingen Laboratory
hhellner@de.ibm.com

Security is an ongoing process. Therefore, we've continued to improve our security after shipping VSE/ESA 2.4. The first topic in this article, for example, shows how the Basic Security Manager's (BSM) functionality is exploited by TCP/IP for VSE/ESA. “Using Your Own Partition Startup Procedures” on page 84 describes considerations when writing your own startup procedures.

Using BSM Capabilities for TCP/IP Security Checks

TCP/IP allows various platforms to communicate with VSE. With this "openness" for VSE, new security requirements arise. This is the reason why TCP/IP for VSE/ESA provides a number of functions to protect VSE resources. (For details, see the *TCP/IP for VSE/ESA User's Guide*.)

One of these security functions is the TCP/IP security exit point. It can be used via the TCP/IP-provided code sample SECEXIT. But neither the TCP/IP internal security functions nor the sample exit code exploit the security functions of the VSE operating system, i.e. the Basic Security Manager (BSM). As a result, customers have to define and administrate the user IDs and VSE resources twice, once in TCP/IP and once in the security system of the VSE operating system.

A solution comes with APAR **DY45309**. This APAR introduces the phase **BSSTISX**, which can be used instead of the TCP/IP-provided code sample SECEXIT. Figure 26 shows the integration of BSSTISX as a link between TCP/IP and BSM.

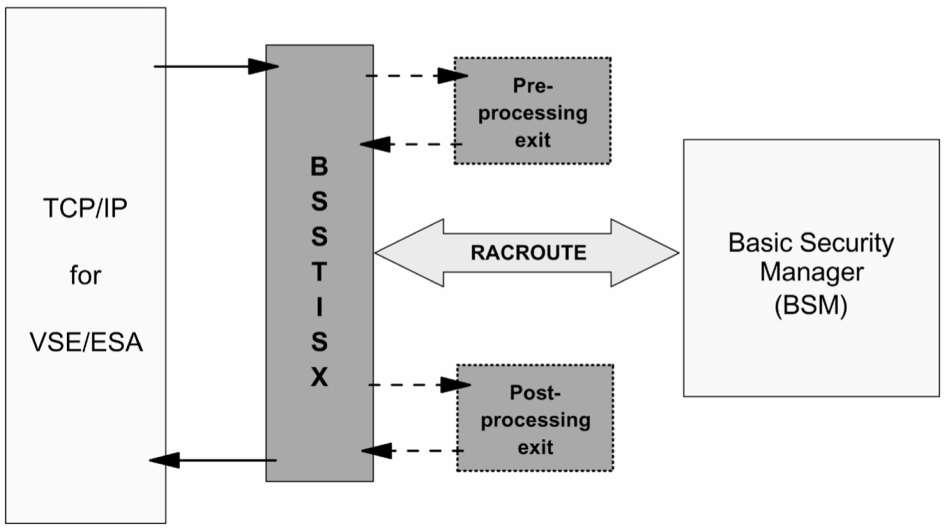


Figure 26. BSSTISX as a Link Between TCP/IP and BSM

The phase BSSTISX exploits the BSM capabilities. It issues RACROUTE requests to process user identification and user authentication, and resource access control for VSE files, libraries, and members. It also allows limited access control to POWER spool files and the SITE command.

Note: Access to POWER spool files will be allowed for administrators and users, where the user ID matches the FROM or TO user ID of the requested spool file. The SITE command can only be used by an administrator.

Certainly, there are various other checks possible via the TCP/IP exit point, which are not covered by BSSTISX. Therefore, BSSTISX provides a pre- and post-processing exit interface. Customers who need additional checks can then write their own pre-/post-processing routines for BSSTISX.

Activation of the Security Exit

To activate the security exit, you have to enter the following TCP/IP commands:

DEFINE SECURITY,DRIVER=BSSTISX[,DATA='data']
: The DEFINE SECURITY command loads the security exit BSSTISX.PHASE into the TCP/IP partition.

SET SECURITY=ON
: The SET SECURITY=ON command activates the security processing and gives control to BSSTISX for initialization. BSSTISX loads additional parts into storage and initializes its control blocks according to the parameters specified in **data**. From now on, TCP/IP passes information to the exit routine BSSTISX for verification.

The parameter **DATA=** of the DEFINE SECURITY command contains the initialization parameter for BSSTISX. The syntax is:

DATA= '[anonym_uid][,[anonym_pwd][,[preproc][,[postproc][,[mode]]]]]'

anonym_uid Here you can specify a user ID, which is defined to BSM. Each time a client logs on with user ID ANONYMOUS, your specified user ID and its access rights will be used.

anonym_pwd With this parameter, you can specify the password of the BSM defined user ID for user ANONYMOUS.

preproc If you like to use a self-written preprocessing exit, specify here the name of your preprocessing exit phase.

postproc For a self-written post-processing exit, you have to specify here the name of your post-processing exit phase.

mode The mode parameter can be used to change the processing options of the BSSTISX exit. Therefore, you have to specify the sum of the selected option codes.

For example, by default all supported checks are active. If you only want administrator user IDs to be able to access your VSE files/libaries (option code 4) and POWER spool files (option code 8), you have to specify 12 as the mode.

The following list shows all option codes and their meaning.

Option Code	Meaning
1	When this option is selected, the **validation of administrator authority for VSE files and libraries is suppressed**. By definition, an administrator can access all VSE files and libraries. Therefore, we first check whether the client user is an administrator. Only if the client is not an administrator will we call the BSM for security validations of VSE files or libraries. Using option code 1 ensures that the RACROUTE requests for VSE files and libraries will always be sent to the security manager. **Note:** When the security manager checks are also suppressed (option code 4), all access to VSE files and libraries is denied.
2	When this option is selected, the **validation of administrator authority for POWER spool files is suppressed**. Only an administrator has read/write authority to POWER spool files. If the requestor is not an administrator, read access may be allowed, which depends on the result of the POWER user ID validation. Specifying option code 2, **no** administrator validation will be done for POWER spool files. The access authority depends on the POWER user ID validation. **Note:** When the POWER user ID validation is also suppressed (option code 8), all access requests to POWER spool files will be denied.
4	With this option, **security manager checks for VSE files and libraries are suppressed**. Specifying option code 4, **no** RACROUTE calls will be issued to check the authorization for VSE files and libraries. When the administrator validation is active, only administrators can access files or libraries. Otherwise, all access requests to files or libraries will be denied. **Note:** This option will be assumed in an environment with SYS SEC=NO.
8	With this option **POWER user ID validation is suppressed**. Specifying option code 8, all access requests to POWER spool files will be denied. Only administrators can access POWER spool files, as long as the POWER administrator validation is active.

Deactivation of the Security Exit

To deactivate the security exit, you have to enter the following TCP/IP commands:

SET SECURITY=OFF

The SET SECURITY=OFF command stops the security processing and gives control to BSSTISX for cleanup and termination. BSSTISX clears its control blocks and frees the storage of its additional parts.

DELETE SECURITY

The DELETE SECURITY frees the security exit BSSTISX.PHASE.

Note: If you want to use a new version of the security exit, you should shut down TCP/IP and restart it again before you enter DEFINE SECURITY.

Using Pre- and Post-Processing Exits

The preprocessing exit gets control after the BSSTISX initialization and later on at the beginning of each request. The post-processing exit gets control at the end of each request except the termination request. Both exits get the required information from the TCP/IP-created SXBLOK.

The SXBLOK describes the interface between TCP/IP's exit point and the security exit. The mapping of the SXBLOK is shipped with TCP/IP for VSE/ESA. Be sure that you use the corresponding level of TCP/IP for BSSTISX. This is TCP/IP APAR **PQ27252** for the first version of BSSTISX.

Both the preprocessing exit and the post-processing exit have to be:

- reentrant
- AMODE(31)
- RMODE(24)

Register settings for preprocessing exit

- On entry:

  R1 Address of SXBLOK
  R13 Standard save area
  R14 Return address
  R15 Entry point of preprocessing exit phase

- On return:

  The preprocessing exit must restore registers prior to return. Register 15 shows the result:

  R15=0 BSSTISX should continue normal processing
  R15=X'E0' BSSTISX should skip all checks and terminate with R15=0 (no violation)
  R15=4 BSSTISX should skip all checks and terminate with R15=4 (security violation)

Register settings for post-processing exit

- On entry:

  R0 Current return code value of BSSTISX
  R1 Address of SXBLOK
  R13 Standard save area
  R14 Return address
  R15 Entry point of post-processing exit phase

- On return:

  The post-processing exit must restore registers prior to return. Register 15 shows the result:

  R15=0 BSSTISX should terminate with R15=0 (no violation)
  R15=n BSSTISX should terminate with R15=n
  n=4 indicates a security violation

Performance Hints

Depending on the TCP/IP usage, BSSTISX may have to issue a high number of user verifications with the same user IDs. For this condition, it is useful to activate the BSM cache via:

MSG xx,DATA=DBSTARTCACHE

where **xx** stands for the partition ID of the security server partition (default is FB).

External Security Managers

The TCP/IP security exit, BSSTISX, also can be used together with External Security Managers (ESMs), if these support the RACROUTE requests issued by BSSTISX. CA-Top Secret (distributed with VSE/ESA 2.4 and by Computer Associates International) supports these RACROUTE calls.

Using Your Own Partition Startup Procedures

If it is required to modify the IBM-provided partition startup procedures or to use your own procedures, the security-related programs have to be considered. The next section tell you more about these programs.

BSSINIT

The BSSINIT routine is the central security initialization service. As an introduction, see the article "VSE/ESA 2.4.0 Security – Implementation Details" in Issue 17 of the *VSE/ESA Newsletter*. This article is available on the Web as **vsebsts.pdf** from:

http://www.ibm.com/s390/vse/vsehtmls/vsenew17.htm

The first time BSSINIT is used is during startup in $0JCL.PROC. It ensures that the right security manager will be started. When the security manager needs a security partition, BSSINIT also will start this partition out of $0JCL.

The second time BSSINIT is called is in the security server partition (e.g. $BJCL.PROC). Here it is used to recognize the partition as the security server partition and that the security server should be started. BSSINIT indicates this via the return code 99.

When $0JCL.PROC or the startup procedure of the security server partition will be modified or replaced, the BSSINIT calls must appropriately be placed in the procedures.

BSSINIT could also be used for automated operation to get the selected security server partition ID. Therefore, BSSINIT returns a number **n**, which means n=1: F1, ..., n=11: FB.

BSTPSTS

The BSTPSTS routine is the security server of the Basic Security Manager (BSM). It has to be started in the security server partition (default is FB). In the IBM-provided procedures, BSSINIT will indicate the server partition which leads to the start of BSTPSTS.

It is recommended not to change the procedure of the security server partition for the BSM.

IESIRCVT

During processing of **$0JCL**, the procedure **USERBG** is called. The procedure USERBG starts the routine IESIRCVT. This routine provides the BSM with initialization information from the Interactive Interface.

The EXEC IESIRCVT statement should be removed when a security manager other than the BSM is used. But it is required for the BSM.

Migration Roadmap for LANRES/VSE Customers – Where to Go from Here?

Author

Anette Stolvoort
VSE Technical Marketing and Sales Support
ahaller@de.ibm.com

VSE/ESA V2.4 is the last VSE release which includes the LANRES/VSE product. This roadmap is intended to help LANRES/VSE customers to make use of the new functions of VSE/ESA V2.5 (which has been announced on February 29, 2000) by showing solutions which could be used as an alternative to their current LANRES/VSE installation.

Note: LANRES/VSE will continue to be serviced on pre-VSE/ESA V2.5 releases until the releases will no longer be supported. For a detailed overview on the support status of VSE/ESA, please refer to the home page:
http://www.ibm.com/s390/vse/vsehtmls/vsestat.htm

TCP/IP for VSE/ESA can be used to replace a majority of today's LANRES/VSE functionality. For more information about TCP/IP for VSE/ESA, please refer to the VSE home page:
http://www.ibm.com/s390/vse/

The following chapters discuss each single LANRES/VSE function. The migration from LANRES/VSE to TCP/IP for VSE/ESA might result in a reconfiguration of your network infrastructure.

Communication

LANRES/VSE supports two communication protocols, channel communication or APPC (SNA LU 6.2) communication.

TCP/IP for VSE/ESA supports a majority of the communications controllers which are also supported by the APPC protocol. The following list shows which communication controllers are supported by TCP/IP for VSE/ESA:

- Integrated Communications Adapter (ICA) of ES/9000 9221 processors for Ethernet, Token-Ring
- IBM 3172 Interconnect Controller for Ethernet, Token-Ring and FDDI
- S/390 Open Systems Adapter 2 (OSA-2) for Ethernet, Fast Ethernet, Token-Ring, FDDI and ATM (LAN Emulation mode only)
- CLAW Connection to RS/6000
- Channel to Channel Adapters (includes virtual CTCA provided by VM/ESA)
- IBM 2216 Nways Multiaccess Connector (attached via parallel or ESCON channels)
- Ethernet and Token-Ring adapter of the Multiprise® 3000

TCP/IP for VSE/ESA does not support the IBM S/390 Channel-to-LAN Connectivity for PCI adapters (9663-001 and 9663-002) nor the internal channel communication of P/390 systems.

LANRES/VSE Functions

Disk Serving

The disk serving function allows Novell NetWare users and OS/2 LAN server users to keep a Novell volume or DOS FAT formatted disk in VSAM.

No fully equivalent function is available in TCP/IP for VSE/ESA.

TCP/IP Network File System (**NFS**) might be able to replace the LANRES/VSE disk function in some installations dependent on the use of the function. NFS has the potential advantage of storing the files in a format that's accessible from both VSE host programs (although maybe not at the same time) and workstations supporting an NFS client. The downside of this is that with NFS the format on VSE is not a full Novell or OS/2 disk.

A second option might be **Samba**. Samba is a freeware package that runs on Unix platforms, including Linux and LINUX for S/390. Samba provides file sharing from Windows, DOS and OS/2 desktops. No intermediate LAN server is needed to connect to SAMBA.

More information on Samba can be found in the redbook, *S/390 File and Print Serving*, SG24-5330. Linux for S/390 can be downloaded from:
http://oss.software.ibm.com/developerworks/opensource/linux390/index.html

If you are using the LANRES/VSE Disk Serving function because you are looking for reliable, large capacity S/390-class storage, an Enterprise Storage Server (ESS or "Shark") might be a replacement option as well. The Enterprise Storage Server can be attached to S/390 as well as LAN servers and delivers reliable, large capacity S/390-class storage to both systems.

Distribution

The LANRES/VSE distribution function can be used to work with Novell NetWare and OS/2 files and directories directly from VSE. This function is mainly used to perform file transfers between a Novell NetWare server or OS/2 server and VSE. The file transfer is initiated by a REXX procedure on VSE/ESA. In addition to file transfers, the distribution function also supports to directly issue Novell NetWare commands like copy, rename, execute.

The **FTP client application** of TCP/IP for VSE/ESA offers the possibility to perform host-initiated file transfers. An FTP Daemon (Server) has to be activated on the Novell NetWare server or the OS/2 LAN server.

The TCP/IP FTP client offers different ways for invocation. One way is to invoke the FTP client from a VSE/REXX procedure, which offers similar functionality to the LANRES/VSE distribution function for performing file transfers. The TCP/IP REXX support became available with APAR **PQ27252** (PTF **UQ38659**).

A TCP/IP Telnet client can be used to connect to a Telnet Daemon (Server) on the Novell NetWare server or the OS/2 LAN server in order to issue commands like

copy, rename, delete etc. The Telnet client can also be invoked from a VSE/REXX procedure. Note that Telnet Daemons are included with the OS/2 LAN server.

Implications: The available REXX/VSE procedures have to be changed in order to invoke the TCP/IP FTP client function instead of the LANRES/VSE distribution function.

LAN-to-Host Printing

The LANRES/VSE LAN-to-Host printing function can be used to directly print from a workstation on the LAN to a VSE attached high volume printer. Different routines on VSE are available to convert LAN print output to the correct print format on VSE. This includes a print exit to translate HP PCL format to AFPDS.

TCP/IP for VSE/ESA delivers the **Line Printer Daemon** (**LPD**), which can be used to replace the LANRES/VSE LAN-to-Host printing function. A Line Printer Requester (LPR) has to be installed on the client which issues the print command. The TCP/IP for VSE/ESA Line Printer Daemon does not support translation from HP PCL format to AFPDS.

Another package which could be used to replace the LANRES/VSE LAN-to-Host print serving functionality is **Samba**. Samba provides print sharing from Windows, DOS and OS/2 desktops. No intermediate LAN server is needed for Samba.

Host-to-LAN Printing

The LANRES/VSE Host-to-LAN printing function allows VSE users to directly print on LAN-attached printers. The VSE/POWER queue is used as an interface for the spool files. Different routines on VSE are available to convert print output formatted on VSE to the correct LAN printer format. This includes a print exit to translate AFPDS to HP PCL and Postscript format.

TCP/IP for VSE/ESA delivers the **Line Printer Requester** (**LPR**), which can be used to replace the Host-to-Lan printing function. A Line Printer Daemon (LPD) has to be installed on the workstation to which the IP printer is attached.

TCP/IP LPR does not support a translation from AFPDS to HP PCL or postscript format. However, PSF/VSE supports to directly print AFP documents on TCP/IP attached printers. TCP/IP support in PSF/VSE became available with APAR **DY45247** (PTF **UD51162**).

TCP/IP for VSE offers an additional function which is not offered by LANRES/VSE, namely CICS Terminal printing. The General Print Server (GPS) function of TCP/IP for VSE can be used to do VTAM printing on IP printers without using the CICS Report Control Feature to put the output to the VSE/POWER list queue. Note that this function can be used for any VTAM 328x application.

LAN Administration

The LANRES/VSE LAN Administration function can be used to automate LAN administration by using REXX procedures on VSE.

There is no fully equivalent function offered by other program packages. However, Novell NetWare server administration can be done from any workstation through a graphical user interface.

Summary

As illustrated in the previous discussion, migration paths are available for each of today's LANRES/VSE functions. Figure 27 summarizes the available options which are a replacement with:

- TCP/IP for VSE/ESA
  http://www.ibm.com/s390/vse/vsehtmls/tcphome.htm
- Samba (e.g. on LINUX for S/390) or
- A mixture of Samba and TCP/IP for VSE/ESA, depending on the current usage of LANRES/VSE

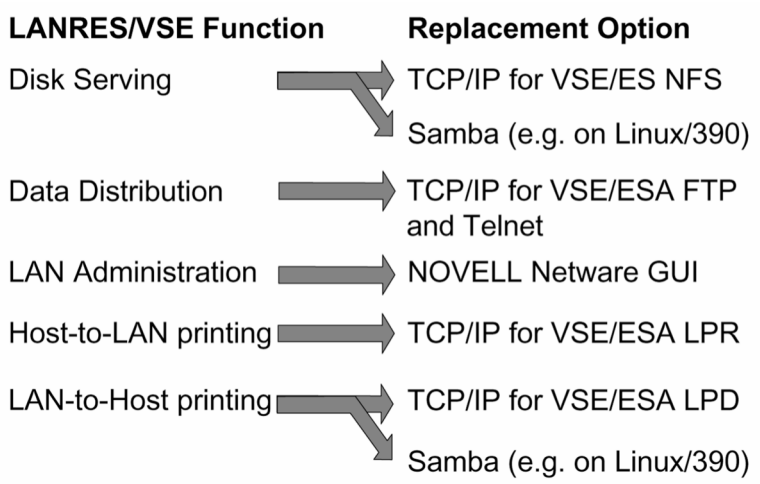


Figure 27. LANRES/VSE Replacement Options

Vendor Products

Products from Independent Software Vendors might offer similar functions to LANRES/VSE. The following is a list of vendors which might be able to assist you in replacing your LANRES/VSE installation.

IBM does not make any representations or endorsement of any of the products or services listed here which are provided by non-IBM sources. IBM has made no effort to independently verify the products or services. Users of this information are responsible for checking with the non-IBM source to confirm the specific implementation of their system. In any event, IBM shall not incur any liability by listing the following information.

- ASG Outbound Suite from Allen Systems Group, Inc.
- GENIOS from Universal Software, Inc.
- CrossAccess from Cross Access, Inc.
- ViaSQL from ViaServ, Inc.
- SPI/DSF from Software Pursuits, Inc.
- IpPack from Barnard Software, Inc.

More Articles And News

Introducing Some "Cool" Technology – Linux on S/390!

Authors

Boas Betzler
Software System Design, Linux for S/390
boas@de.ibm.com
Ed Gauthier
S/390 Marketing
gauthier@us.ibm.com

Linux is getting a lot of attention and press these days, especially since the press articles in January that all IBM server platforms will be Linux ready. This has caused some people unfamiliar with Linux to ask, “Just what is it?” And other “experts” who have heard about Linux on S/390 but never have seen it scoff and say, “What? Linux running on IBM mainframes?” Whether you fit into one of these groups or are just curious, please read on to find out about some real cool technology and an easy way to get Linux applications running on S/390!

What Is Linux?

Initially developed by Linus Torvalds, Linux is a UNIX-like operating system that supports multiple hardware platforms. It is an Open Source Software (OSS) operating system and is developed and distributed freely via the Internet under the GNU Public License (GPL).

Linux is gaining popularity because it has a reputation for reliability and performance. Due to the similarity between commercial UNIX systems and Linux, existing UNIX applications can be readily migrated to work under Linux.

Linux on S/390 is an open-source based implementation of the Linux operating system. It runs natively on S/390 hardware as well as under VM/ESA. It is a release of code to the open source community so that customers and business partners can start to understand how Linux can be used with S/390. This code is not, therefore, supported or serviced the way that other S/390 operating systems are.

Linux on S/390 adheres to the standards and interfaces found on other Linux implementations. It runs in ASCII mode and supports common applications and development tools such as GNU tools. *This is key to deployment and use of Linux on S/390, since application porting is not required.* Our experience is that most applications simply need recompilation to run on S/390 hardware.

Linux consists of *architecture independent* and *hardware architecture dependent* parts. Only the architecture dependent layer of Linux had to be changed for the port to S/390. The system structure of Linux and the de facto interface were not changed, so that all Linux extensions and applications could run unchanged on S/390.

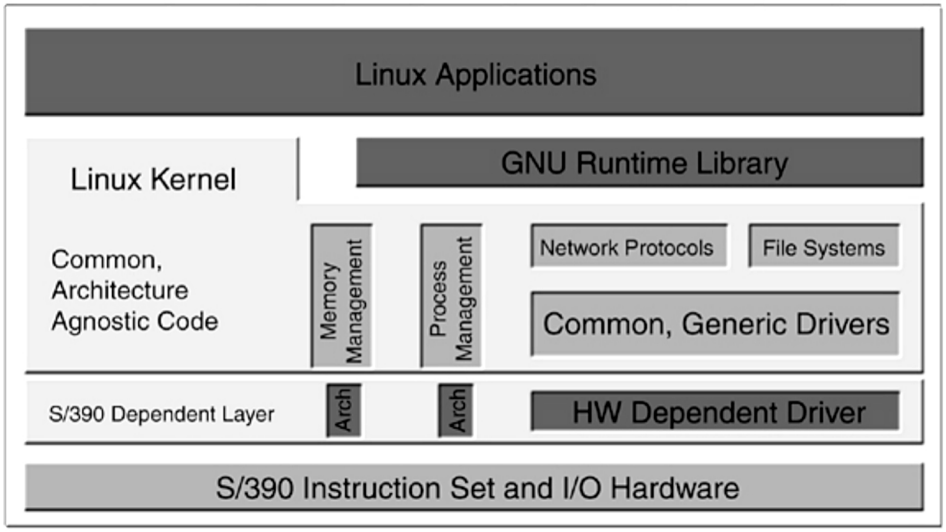


Figure 28. Architecture Layers – Linux on S/390

What Can You Download?

As shown in Figure 28, the Linux operating system kernel consists of:

- A process scheduler,
- Memory management,
- Inter-process communication, and
- I/O support for the network and file systems.

Most of these building blocks are "architecture agnostic (independent of the architecture) and written in portable C. But for some of these blocks, a thin architecture dependent layer is required and had to be provided for the S/390 hardware.

In addition to the Linux kernel itself, the GNU development environment was adopted for Linux for S/390. This includes the GNU C compiler, the binary utilities that include the GNU assembler, and the GNU C runtime library. Using these tools, other GNU and Linux applications can be built.

This code can be installed on S/390 Parallel Enterprise Servers® – Generation 3 or newer, Multiprise 2000 servers, and Multiprise 3000 servers. It is self-contained and does not require assistance from other operating systems.

Device drivers for networking, disk access, and console interaction are available. Network drivers include Ethernet and Token-Ring for the OSA-2 interface or other LCS (LAN Channel Station) devices. For communication with other S/390 operating systems, Channel-to-Channel and virtual Channel-to-Channel drivers can be included.

With a full-featured TCP/IP stack and the existence of many networking applications, it provides everything needed to build a powerful Internet server.

Data can be stored on DASD using the ECKD drivers for IBM 3390, 3380 and 3945 devices, which also provide Linux boot support (as IPL is called in the UNIX world).

Most installations use the second extended Linux file system as disk format. Under VM, the option also exists to use a minidisk as disk device.

Several applications already have been recompiled for Linux on S/390. These include the Apache Web server, Samba file and print serving, THE editor (an Xedit clone), SSH secure shell, and several XWindows applications like X3270 (a graphics-based 3270 emulator supporting the TN3270 protocol).

Additional Information

To learn more about this cool technology on S/390, please feel free to contact **linux390@de.ibm.com**. To download the code that makes up this open source contribution, use the URL shown in Ed Gauthier's report on the next page.

The Open Source and GNU Project home pages are:
http://www.opensource.org/ http://www.gnu.org/

The general IBM site for information about Linux at IBM is:
http://www.ibm.com/linux/

An On-Location Report from LinuxWorld Expo

Proving a Simple Point

It's 10 o'clock on Wednesday morning February second. Boas Betzler and I are at LinuxWorld Expo at the Jacob Javits center in New York City. We're getting our demonstration position ready. We've got Linux running on an S/390 Multiprise 3000, and it's being shown by IBM for the first time here at Linux World.

Our demonstration is a simple one, to prove a very simple point. We have a server running on Linux on our Multiprise 3000. It feeds MP3 music streams to a "jukebox installed on a client workstation also running Linux. The point we make is that what's driving the S/390 is Linux – not another operating system that appears to be Linux, but real Linux, running on the metal.

So Where's the Big Mainframe?

One of our friends from the Server Group Linux team stops by. He tells us that Linus Torvalds, the creator of Linux, mentioned S/390 in his keynote address, and we should be prepared to be very busy. Our friend from Server Group is right. It seems as if everyone at the show wants to see Linux on the S/390, and "everyone at a Linux event covers a lot of ground.

Some people know what S/390s are and spot our Multiprise 3000 right away. Some of the visitors circle the IBM booth a couple of times looking for the great big mainframe and don't find one. Eventually, they see the sign above our position and stop by. Some people, even after seeing the sign, remain quite skeptical. One fellow accuses me of standing next to a control unit!

We meet college students who use Linux at school who stop by to see what a mainframe looks like. We see elementary school students attracted by the x-pilot game we have running. We meet people who worked with S/390 processors years ago, who stop to reminisce and can't get over the compact size and power of our Multiprise 3000.

We meet members of the press, and we meet people from Fortune 500 companies who are very curious about Linux running on S/390. They tell us that they have logical partitions on their S/390s that were used for Y2K purposes until recently which are now available for something new, and they are quite curious about Linux on S/390.

The Basic Questions

- **"Is it really Linux running on S/390, or is it a Linux personality on another operating system?"**

  We tell them it's Linux running on the metal; and if they don't believe us, here's a terminal take a look for yourself. Some take us up on the offer, and are quickly convinced.

- **"Is this an IBM product?"**

  No, it's an Advanced Technology Project; and it's available to download from the Internet at no charge.

- **“Why run Linux on S/390?”**

  Where should I begin – reliability, scalability, the ability to work closely with other operating systems without network latency, large numbers of Linux servers running on VM/ESA. I could go on and on.

- **“Has any code been recompiled to Linux for S/390?”**

  Yes, quite a bit, like the Apache Web server, Open LDAP, Perl, Regina, and a lot more.

- **“Where can I get Linux for S/390?”**

  We hand out business cards with the URL of the IBM developerWorks Web site on them. It's comforting to see them going into shirt pockets and wallets instead of the bags full of t-shirts and toys that everyone seems to be carrying.

So if you can't be here to get one of our cards, here's the URL to reach the IBM developerWorks page for Linux on S/390. Pass it along!

http://oss.software.ibm.com/developerworks/opensource/linux390/

S/390 Multiprise 3000 Enterprise Server – Introduction

Editor's Note

This introductory article to the Multiprise 3000 originally appeared in Issue 26 of the *S/390 Bulletin* (December 1999). The next article by Mike Hammock, "Multiprise 3000 – Leveraging the Extra Power at No Extra Cost" on page 99, has more information for those of you interested in using this new server.

The home page for the *S/390 Bulletin* is:
http://www.ibm.com/s390/bulletin/

In September 1999, IBM announced the S/390 Multiprise 3000 Enterprise Server (Figure 29). This powerful new addition to the IBM Multiprise family provides e-business processing enhancements including collaborative computing, e-commerce, enterprise resource planning, and application development.

The Multiprise 3000 is an ideal platform for developing client/server applications, especially within coupling environments. Replacing existing S/370™ or S/390 processors with a Multiprise 3000 can be implemented because the Multiprise 3000 runs existing S/390 applications and attaches to a number of I/O devices and control units currently supported by S/390. This cost-effective member of the Multiprise family also can act as a server or a client within the LAN/WAN environment.

Figure 29. S/390 Multiprise 3000 Enterprise Server

A Cost-Effective Solution

The S/390 Multiprise 3000 Enterprise Server enables you to fully exploit the most advanced S/390 applications and expand your overall capacity while reducing operating costs – all at a lower price than previous S/390 Multiprise servers.

The Multiprise 3000 offers significant enhancements to the business enterprise. Its new features include an integrated processor and memory card, internal SSA RAID-5 DASD, industry standard PCI/ISA connections, parallel channel adapters, and ESCON channel adapters. This server provides customers with a new S/390 platform to enable their applications, replace their S/370 and S/390 processors, and/or provide application servers in a single system unit.

S/390 Multiprise 3000 Servers are supported by most of the current levels of OS/390, VM/ESA, and VSE/ESA.

Base Configurations and Features

Integrated in the base processor configuration of the Multiprise 3000 are:

- N+1 power
- One processor-memory card
  - A uni-processor, System Assist Processor, and 1GB – Model H30
  - A uni-processor, System Assist Processor, and 2GB – Model H50
  - A dyadic-processor, System Assist Processor, and 4GB – Model H70
- Four PCI adapter slots
- Three ISA adapter slots
- 12 parallel and/or ESCON adapter slots
- Differential SCSI adapter slot for attachments to 3490 Model F01 Tape Drive
- 14GB of usable emulated DASD
- 16 slots for up to 216GB of usable DASD
- Integrated 100/10 Mbps PCI Ethernet
- 4-mm DDS-3 DAT tape drive
- One CD-ROM and one diskette drive

Integrated in the first expansion frame configuration of the Multiprise 3000 are:

- 20 slots for up to 288GB of usable DASD
- 16 slots for ESCON and/or parallel channel adapters

Integrated in the second expansion frame configuration of the Multiprise 3000 are:

- 20 slots for up to 288GB of usable DASD

Features available for the Multiprise 3000 include:

- SSA RAID-5 DASD (18.2GB drives available)
- ESCON Channel Adapters and Parallel Channel Adapters (2 ports per card)
- Token Ring, Ethernet, and WAC Adapters
- Integrated Backup Battery

- Cryptographic Co-processor
- Hardware Management Console

A Great Application Development Tool!

The S/390 Multiprise 3000 Server is a cost-effective platform for developing client/server applications in the S/390 environment and is ideal for application development teams in coupling environments. The Multiprise 3000 exploits the productivity of S/390 development tools and provides immediate access to the real S/390 architecture for testing and debugging applications.

Developers can use S/390 3GL and 4GL tools and object-oriented technology to shorten the development cycle for S/390 environments. When the Multiprise 3000 becomes a dedicated resource for development and testing, it can use the same tools as the host, thereby increasing the availability of the host production system, eliminating dependencies on the host, and reducing risks to the production system.

A Great Replacement Processor!

Users of S/390 general-purpose processors can gain substantial savings by moving to this new, attractively priced technology now. Many customers with existing S/390 hardware and software are participating in the Local Area Network (LAN) environment. The Multiprise 3000 addresses business needs by providing the opportunity to grow in the future, while continuing to use S/390 applications and skills.

Instead of porting S/370 applications to a new platform, the Multiprise 3000 allows for easy upgrades to the classic S/390 strengths. Thus applications do not have to be moved to an entirely new platform that lacks the S/390's advantages. In most cases, you can continue to connect to existing tapes and printers with the parallel connection or utilize the ESCON feature to connect to new devices, as well as those which could not be connected in the past.

The Multiprise 3000 is a possible replacement for older S/370 and S/390 processors (43XX, 9371, 9373, 9375, 9377, 9221, 9121, 9021, and some 9672), depending on your requirements. Replacing S/370 systems with the Multiprise 3000 means that you can take advantage of the latest technology, including S/390 ESA software, while retaining current skills. The S/390 software provides complete Year 2000 support. Non-ESA software and S/370 mode are not supported.

Substantial savings may be achieved through lower maintenance costs on both the processor and DASD, lower energy costs, lower connectivity costs, and lower environmental costs. These potential benefits are derived from integrated disk characteristics with their high-reliability, low-power requirements, LAN connectivity, and CMOS technologies.

A Great Application Server!

As an application server, the Multiprise 3000 is ideal for a department or small business because it can run S/390 applications unchanged. Remote locations (some unattended) are possible, and remote users can access data and consistent business applications across the enterprise. In addition, Multiprise 3000 application servers solve business needs by bringing specific workloads into LAN environments. The results include improved productivity, local control, and reduced telecommunication costs.

The Multiprise 3000 can participate as a client or a server in an existing LAN with access to host data. As a client, it can seamlessly participate in any LAN environment. As a server, the Multiprise 3000 can run the appropriate LAN functions associated with S/390, depending on customer workload. This choice allows customers flexibility and availability while also satisfying the need to integrate business solutions with consistent applications.

Additional Information

For more information, refer to the article by Mike Hammock, “Multiprise 3000 – Leveraging the Extra Power at No Extra Cost” on page 99 and to the IBM Hardware Announcement Letter **199-240**. You also should visit:

http://www.ibm.com/s390/multiprise/

Multiprise 3000 – Leveraging the Extra Power at No Extra Cost

Author

C. M. (Mike) Hammock
Senior Support Representative
*Intelli*Ware Systems, Inc.
mhammock@intelliware.com

Introduction

Purpose

The IBM S/390 Multiprise 3000 Enterprise Server offers a very powerful and flexible system for many S/390 users. It is this very flexibility, however, which sometimes causes confusion and problems for some customers. With proper planning and a good knowledge of the system, the capabilities of the system can be exploited and any potential problems can easily be avoided.

As an IBM Premier Business Partner and a leading provider of P/390 technology to customers in the United States, *Intelli*Ware has extensive expertise and has installed a number of these systems for customers, and we have learned many important considerations and guidelines for the installation and use of the Multiprise 3000. This paper is our way of passing along some of this knowledge and expertise to others.

In this article, we will first introduce the S/390 Multiprise 3000 from a technical viewpoint, looking at the basic design and architecture of the system and some of its key components. We will look at considerations for planning and using components such as the disk subsystems, tapes, channel-attached devices and LAN and WAN networking. We will describe how to exploit the flexibility and capabilities of the system while avoiding possible problem areas to get the most out of your investment

What Is a Multiprise 3000?

The Multiprise 3000 is the newest member of the IBM S/390 family. It offers an outstanding combination of power, capacity, and flexibility in a surprisingly compact and economical package. These basic attributes provide an excellent platform for a wide range of enterprise computing needs at a very attractive total cost of computing. For an overview of the Multiprise 3000 system, see “S/390 Multiprise 3000 Enterprise Server – Introduction” on page 95 in this issue of the *VSE/ESA Newsletter*. While that article provides a general introduction to the system, this article delves deeper into the technical aspects.

Technical Overview

In order to understand some of the considerations and recommendations we will be discussing later, we must first understand some of the key elements of the design of the Multiprise 3000. It also will be interesting to explore the origin of some of the components.

Basic Design / Architecture

Prior to the announcement of the Multiprise, IBM had three rather distinct lines of S/390 processors.

1. The 9672 or S/390 Parallel Enterprise Server has been the top of the S/390 line, featuring the Gn (G3, G4, G5, G6) processors, with full Parallel Sysplex and the most comprehensive mix of large system capabilities.
2. The 2003 or Multiprise 2000 was the midrange system of choice for S/390 users who didn't require Parallel Sysplex or the capacity of the 9672.
3. The P/390 Technology Family – including the P/390, R/390 and most recently, the Integrated Server – has been based on the P/390 processors and its PC hardware-based host system. It provided S/390 capability to entry users and developers at a very attractive cost.

As illustrated in Figure 30 on page 101, the Multiprise 3000 combines many of the features and capabilities of each of these systems into one flexible design. For example:

- The Multiprise 3000 (IBM Model 7060) uses the processors from the 9672 G5 family, repackaged onto a different form factor with a maximum of three processors on a system, instead of twelve as on the 9672.
- The 7060 uses the System Assist Processor architecture from the 9672 and 2003. The System Assist Processor is a full-function S/390 processor that is dedicated to system control and I/O processing functions, rather than executing normal S/390 programs.
- The Multiprise 3000 uses much of the design of the Internal Disks from the Multiprise 2000, but with components (such as SSA channels and RAID-5) from the Integrated Server.
- The System Element (SE) design and Input/Output Control Program (IOCP) functions are from the 9672 and Multiprise 2000, with a few modifications to accommodate emulated I/O devices from the Integrated Server.
- The ESCON and Parallel channels in the Multiprise 3000 use the same component chip sets as the Multiprise 2000 channel cards.
- The Emulated I/O (EMIO) devices are derived from the Integrated Server and allow the use of relatively inexpensive PC type adapters and components.
- The physical frame and power/cooling components are from the Integrated Server (but were initially designed with the Multiprise 3000 in mind). This provides a compact, yet very robust, server package.

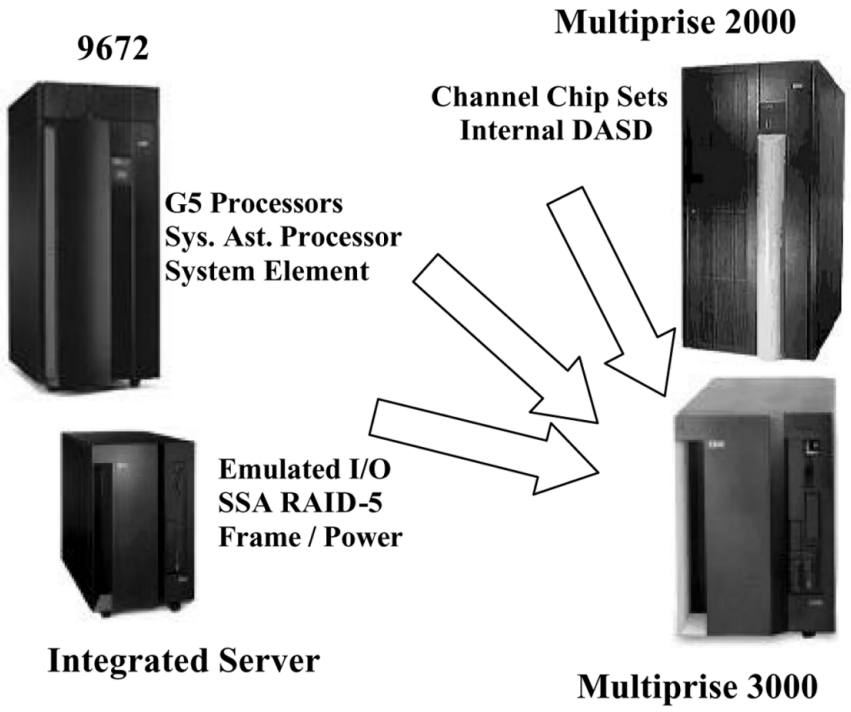


Figure 30. Where It All Came From

The Multiprise 3000 designers and developers took the best parts and features from the existing S/390 servers and combined them into a single powerful system. We now have the power of the G5 processors with the performance and capacity of a redundant Internal disk subsystem, plus the flexibility and low cost of emulating I/O devices using PC-based technology.

Sounds Great! So What's Not to Love?

Each of the components are well tested and proven to work well within their respective systems over several years of development and use.

- The G5 processor and System Assist Processor architecture is solid as a rock and provides outstanding performance and reliability.
- The ESCON and Parallel channels provide excellent I/O throughput.
- The Emulated I/O devices allow the exploitation of devices previously unavailable to large S/390 systems.

It is now possible, however, to combine the components in such a way as to cause a mismatch in capabilities or performance. For example, while the emulated I/O provides a very good function, it was initially intended to be the I/O subsystem for a small P/390 or Integrated Server of around 8 MIPS. It would be foolish to expect the EMIO to provide the primary I/O subsystem for the G5 processors that are orders of magnitude more powerful than the P/390 processor.

Let's look a little more closely at some of the more important Multiprise 3000 subsystems and components. We'll consider the IOCP functions, Internal Disk, S/390 channels, and Emulated devices. Please keep Figure 31 on page 102 in mind as we do this.

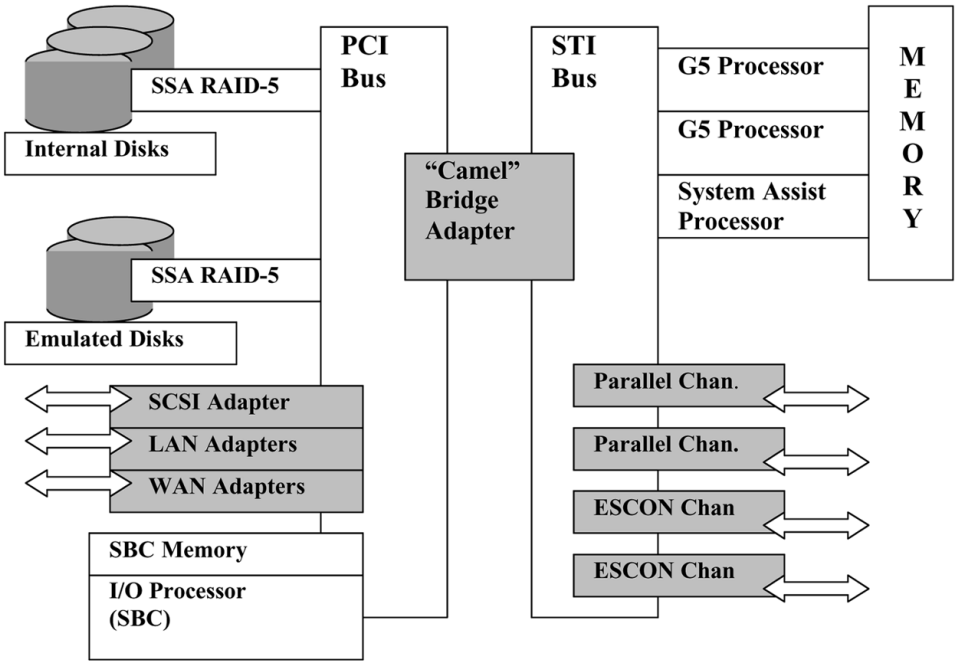


Figure 31. Functional Components

IOCP and Device Map Functions

The Input Output Configuration Program is the facility that describes the intended I/O and LPAR/device configuration. In other S/390 systems (other than the P/390 and Integrated Server family), the IOCP and IOCDS (I/O Configuration Data Set) defines all the I/O devices that are attached (or which may be added in the future) to the system. Each channel path (CHPID), control unit, and device is described in the IOCP definition.

In previous systems, the IOCDS describes the actual "real" hardware I/O devices, but the Multiprise 3000 adds the emulated device and "Device Map" concept from the P/390 family. In the P/390 (and Integrated Server) systems, the Device Map describes how PC type devices are to be used to emulate the desired S/390 devices.

With the Multiprise 3000, a Device Map describes emulated devices that must then be further described and defined via the IOCP. You could think of the Device Map as being "closer" to the hardware than the IOCP.

The Device Map function does not have a CHPID concept, as everything was logically defined on a single CHPID. Internally, the P/390 support code translated the Device Map into a single CHPID HSA (Hardware Storage Area) in the system's memory. The P/390 ancestry of the Device Map has several results that must be considered when configuring the Multiprise 3000:

1. All emulated I/O devices must be defined on a single CHPID. Specifically, they must be defined as being on CHPID **FC**. Because of this, there can be a maximum of 255 emulated devices; and we suggest some methodology be used in assigning device numbers to the emulated devices. One approach

(and some other good information) is contained in an IBM redbook, *Multiprise 3000 Basic Emulated I./O Definitions* (SG24-5669).

2. For every emulated device, a Control Unit (CNTLUNIT) and a Device (IODEVICE) macro must be coded in the IOCP. If many emulated devices are defined, this requirement causes a rather large and unwieldy IOCDS.

There are four IOCDSs available in the Multiprise 3000. For each IOCDS there is a corresponding Device Map. The Device Map is maintained via a function in the System Element.

Recommendations

This paper does not fully describe how to update and control the Device Map and IOCDS, but rather points out some items that may make this process easier and more reliable.

- Plan ahead! Lay out the emulated device configuration well in advance and review it thoroughly. Once the system is operational, the emulated device configuration only can be changed by updating the Device Map and the IOCDS, making the new IOCDS the current one, and doing a Power On Reset (POR).
- Develop a plan for addressing the emulated control units and devices AND USE IT. This will reduce the frequency of incorrectly defined devices and make the system easier to manage and operate.
- Become familiar with the System Element function. There are significant differences from other S/390 System Element programs, and much different from how the P/390 configuration was managed.

The Multiprise 3000 IOCP, Device Map, and related functions offer outstanding capabilities and flexibility, but it is different from all prior systems and needs to be understood and properly managed to be effective.

Internal Disk Subsystem

The Internal Disk subsystem of the Multiprise 3000 has a number of interesting features and considerations. (This section looks exclusively at the Internal Disk subsystem. We will consider it in conjunction with the other disk components in a later section.)

Implementation

The physical components consist of a number of 18GB disk drives connected via SSA (Serial Storage Architecture) to a PCI controller and configured as RAID-5 arrays. In addition to the RAID-5 redundancy, one hot spare drive is included in each enclosure to provide even higher data integrity. For a complete description of the internal disk subsystem see the IBM redbook, *Multiprise 3000 Technical Introduction* (SG24-5633).

While many of the components are derived from the Integrated Server, it is controlled via the System Assist Processor and features many of the same characteristics as the Multiprise 2000 Internal Disk. For example, a portion of central storage may be reserved and allocated for use as a data cache. There are significant differences, however, and these can be very important.

Considerations

- Capacity is added a RAID array at a time, and the array can provide either 72 or 108GB. Each disk increment is predefined and cannot be changed.
- The internal disk can emulate several CKD device types, and the allocation is more flexible than on the Multiprise 2000. Because of the manner in which disk space is allocated, use of 3390-1 and 3390-3 devices will work best for most users.
- Use of RAID-5 imposes the "RAID-5 Write Penalty," effectively slowing most write operations significantly. This effect, combined with the RAID data striping makes the I/O performance very dependent on the number of physical disk drives available on the system.

  An IBM white paper, *IBM S/390 Multiprise 3000 Internal Disk Performance*, covers this effect with detailed analysis of various configurations, compared to the Multiprise 2000. While the Multiprise 3000 can sustain very high I/O rates, having a sufficient number of disk devices is very important. The Write Penalty and the white paper should be considered very carefully when configuring the internal disk subsystem. It will frequently be necessary to "over configure" the disk capacity in order to include sufficient disk devices to sustain needed performance levels.
- Each PCI RAID adapter includes 32MB of battery backed-up memory that is used as a data cache. Since the memory is, in effect, nonvolatile, "Write-Back" caching can and should be used, similar to "DASD Fast Write." In a RAID-5 environment, not using Write-Back mode can cause a significant disk performance degradation. The slowdown may be as much as a factor of 5.
- All internal disk devices are defined as being on a single channel path, CHPID FD. In addition, disk space is allocated in "Storage Units", which is the size of a 3390-1 (about 1GB). If only 3390-1 devices are used, however, there will be insufficient addresses available, so we recommend that most space be allocated as 3390-3s.

When properly configured, the internal disk subsystem can provide a very cost-effective and high performance solution to today's storage intensive applications.

S/390 Channels

As stated earlier, the Multiprise 3000 uses the same channel chip set as the Multiprise 2000. Due to space and form factor requirements, the channels are packaged with two channels per adapter card. Either parallel or ESCON channels may be installed in either the system unit or the first expansion enclosure.

- While the system specifications indicate up to 56 channels can be installed, this will seldom be possible. Channel cards occupy the same physical space as the emulated I/O adapter cards. Thus there frequently will be a trade-off between channel adapters and emulated I/O adapters.
- In addition, each specific channel card slot (referred to as "BI-DI" slots) has a pair of channel IDs assigned to it. Since some channel adapter slots may not be available, this means that the CHPID numbers will not be assigned sequentially.
- Because the channel cards use the Multiprise 2000 chipsets, the channels are logically grouped in sets of 4 channels, or two cards. This means that each

group of four channels must be of the same type, parallel or ESCON. Channels 1-4 must be of the same type; 5-8 the same type; and so on.

The S/390 channels on the Multiprise 3000 can provide good performance and flexibility with only reasonable planning required.

Emulated I/O Devices

The concept of emulated I/O (EMIO) devices will be very familiar to P/390 or Integrated Server users, but somewhat strange to traditional S/390 users. The EMIO subsystem uses PC-based devices and facilities to provide what appear to be normal S/390 devices to the IOCP and S/390 operating system. (For a full description of Emulated I/O and how to define it, see the IBM manual, *Multiprise 3000 Emulated Input/Output Subsystem Users Guide and Reference* (GC38-0610). Some examples of the more common types of Emulated I/O are:

- An OS/2 file can appear to be a S/390 disk drive, such as a 3380 or 9336.
- A PC on a LAN can access the system via SNA or TCP/IP and appear to the system as a local non-SNA 3270.
- A SCSI-attached tape drive can appear to be a channel-attached 3490. The emulated I/O subsystem (as well as the System Element and some other control functions) runs in an OS/2 environment on the SBC (Single Board Computer).

How EMIO Devices Are Defined

Emulated devices are defined in the Device Map using the Emulated I/O configurator tool, then in the IOCDS and processed by the IOCP, finally being recognized by the S/390 operating system as normal S/390 devices. For every emulated device, there are at least three components:

1. The PC-based device or resource which will provide the service,
2. A "Device Manager" which runs in the OS/2 - SBC environment and provides most of the device emulation function, and
3. The desired S/390 device characteristics.

For example, the OS/2 file `F:\VSE\SYSWK1.140` is made to look like a 9336-20 with a device number of 140 by the AWSFBA device manager.

All of the definition, control, and data movement for emulated devices is handled by the OS/2 - SBC environment, not by the S/390's System Assist Processor. The Device Managers and related components perform functions similar to functions performed by the System Assist Processor, in addition to the various transformations required to accomplish the device emulation task.

EMIO Considerations

In the earlier Integrated Server and P/390 systems, all of the I/O activity was "Emulated I/O" and done via the Device Managers in the OS/2 system. In these systems, the I/O subsystem only needed to feed an eight MIP processor, and even then the I/O subsystem was frequently the constraining factor for system throughput. In the Multiprise 3000, the I/O subsystem has to feed a S/390 CPU as much as 25 times faster than the P/390, so the emulated I/O component can only provide a small part of the total I/O volume.

We will cover the various types of I/O devices individually, but most of the suggestions will be in support of the following general recommendations.

- Use System Assist Processor-controlled (non-emulated) devices whenever possible to reduce the load on the emulation processor (the Pentium running OS/2).
- Use the emulated I/O devices for testing or development work where flexibility is important, but I/O volumes are not high.

Disk I/O Subsystem

There are three means of attaching disk devices to the Multiprise 3000:

1. Channel-attached external units (such as RAMAC or ESS units) connected via ESCON or Parallel S/390 channels.
2. Internal Disks, using the SSA RAID-5 disks controlled by the System Assist Processor, as described above.
3. Emulated Disks, using the EMIO subsystem and separate SSA RAID disks with a maximum capacity of about 12GB.

Each of these has an important role to fulfill.

Channel-Attached DASD

Channel-connected DASD (such as the RAMAC or ESS units) will normally be used when they are already in place, possibly connected to a previous system, or when it is desirable to share the DASD between the Multiprise 3000 and some other system. You can expect very good performance from this configuration, similar to 9672-type systems when accessing these devices. For best performance and flexibility, ESCON channels are recommended for this connection. Traditional considerations (channel utilization, control unit caching, etc.) apply to this configuration.

Internal DASD

The Internal DASD (SSA, RAID-5 disks controlled by the System Assist Processor) usually offer the best combination of flexibility, performance, and cost effectiveness. Although this subsystem has been discussed previously, there are some additional considerations as well as some that bear restating.

- The Internal Disk can only emulate ECKD type devices (3380 and 3390) and cannot emulate FBA devices. (FBA can only be emulated by the Emulated DASD. See the next section for more information.)
- Internal disks are defined in the IOCDS as being on CHPID FD, which is type DSD and are allocated in storage units equivalent to 3390-1 volumes.
- Performance is a function of several items, including:
  - Read/Write ratio (because of the RAID-5 Write Penalty effect).
  - Number of arrays and physical drives installed and, therefore, the number of read/write heads and SSA paths across which to spread the I/O load.
  - The amount of central storage allocated to disk data cache.
  - Exploitation of "Write-Back" caching mode on the SSA RAID adapter.

- To avoid extra Power On Resets (PORs), define all installed disk space at initial install as the preferred device types, then initialize and use the volumes as needed.

Emulated DASD

The Emulated DASD uses three 9GB disks (in an SSA, RAID-5 configuration) that it shares with the OS/2 System Element and other system control functions. About 12-14GB will usually be available for emulated DASD. Since the Emulated DASD require EMIO resources (Device Managers, Pentium processor, etc.), it is best to minimize the use of this type of disk resource. There are circumstances, however, when its use is very appropriate and proper.

- If migrating from a system which used FBA architecture disks, it will probably be easiest to copy the FBA volumes onto the Emulated DASD, then migrate them onto ECKD volumes on the Internal DASD. This migration can then extend over a long period of time as long as the I/O rate to the FBA volumes is not high and the required FBA data can be accommodated in the 12-14GB.
- When utilizing a "Pre-configured CD" to load the basic operating system, it will be necessary to load initially to the Emulated DASD because the OS/2 environment (from which the CD is installed) does not have access to the Internal DASD. After the initial install, the S/390 system can be used to copy the installed volumes (using DDR, DF-DSS, etc.) to the appropriate Internal DASD volumes.

While Emulated DASD can offer some very good functions and flexibility, it will generally not perform as well as the Internal DASD or modern channel-attached DASD. High I/O rates on the Emulated DASD also can adversely affect the performance of other emulated I/O devices.

Tape Attachment and Use

As with the other devices discussed previously, there are several ways to attach, define, and use tape drives on a Multiprise 3000.

1. Standard channel-connected tape units (3480, 3490, 3490E, etc.) may be attached via parallel or ESCON S/390 channels. When attached in this manner, definition and use is conventional.
2. SCSI tape drives using 3480 or 3490 compatible media may be attached to an optional Differential SCSI adapter. Tape drives connected in this manner will be controlled by the AWS34xx Device Emulator and device driver, a part of the Emulated IO subsystem. While other tape drives may be used, the only officially IBM supported unit is the IBM 3490-F01/F11.

   Any of these drives should read and write tapes that are fully compatible with channel-connected tape units. The performance of these drives varies considerably, but the better ones will approximate the performance of a parallel channel-attached drive. The primary advantages of SCSI tape units are their compact size and generally lower maintenance costs.
3. SCSI tape drives using non-compatible media (4mm DAT and DLT are the most common) can also be defined and controlled via the AWS34xx Device Manager and Emulated I/O subsystem. Although the media is not compatible with normal S/390 devices, DLT drives are frequently used as backup devices because up to 80GB can be stored on a single tape cartridge.

Considerations and Restrictions for Tapes

As previously mentioned, the definition and use of channel (ESCON and parallel)-attached tape drives is very conventional. Normal considerations apply. On the other hand, there are some different considerations when using SCSI-attached tape units.

- The 4mm DAT tape unit (included with every Multiprise 3000) should be used only for loading products or data from tape or other incidental and convenient functions that do not involve storage of important data. *The 4mm DAT should never be used for the backup or storage of irreplaceable data.* While it provides a convenient and economical function, the 4mm DAT drive and media does not provide the level of reliability and data integrity that is associated with S/390 systems.
- SCSI-attached DLT tape drives, on the other hand, have proven to be very reliable and offer good performance. Numerous P/390, Integrated Server, and Multiprise 3000 customers use these as their primary backup / archival device.
- SCSI tape drive, either DLT or 3480/3490 compatible, can be used to backup some of the system data from the OS/2 environment using the supplied utility, XTAPE. The OS/2 System Element environment and the Emulated DASD contents (not Internal DASD) can be backed up in this manner.
- Since the OS/2 system does not have access to the Internal DASD, this data cannot be backed up via XTAPE. Therefore, S/390-based backup facilities must be used to back up the Internal DASD. In order to exploit the DLT media capacity, it would be desirable to back up multiple DASD volumes onto a single DLT tape volume.
- For systems with a lot of tape activity or with tight batch windows when tapes are used extensively, attachment via S/390 parallel or ESCON channels is recommended. This permits the use of multiple channel paths and the dynamic sharing of tape units with other systems, neither of which is possible with SCSI-attached units.

Local Area Network Connectivity

The Multiprise 3000 utilizes the P/390 or Integrated Server method of connecting to the LAN environment with PCI LAN adapters (Ethernet or Token Ring) emulating a 3172 for the S/390 system. Whereas the Multiprise 2000 and 9672 systems use the System Assist Processor-controlled Open Systems Adapter (OSA) and OSA/2, the Multiprise 3000's PCI LAN adapters are controlled by the EMIO subsystem.

Considerations for LAN Connectivity

While the use of PCI LAN adapters offers very inexpensive access to the LAN, it does impose some restrictions and considerations.

- Only Token Ring and Ethernet adapters are currently supported with no announced support for FDDI or ATM.
- The number of ports or access points to the LAN environment may be limited.
  - A maximum of four LAN adapters/controllers are supported.
  - Each adapter provides one port, and "Port Sharing" is not supported.
  - Each IP address will require a dedicated port/adapter.
  - SNA access may or may not be able to share a port with an IP port, depending on the network configuration.

The net result is that the number of adapters may sometimes be insufficient for the desired configuration, especially when using multiple LPARs or VM Guests. If there are insufficient LAN ports, several alternatives are available.

- Use one system/LPAR as the primary network interface and use IP forwarding across an ESCON or Virtual CTC connection to the other LPARs or VM guests.
- Use "outboard" LAN connections such as channel-attached 3174, 3745, 3172, 2216, or OEM devices such as the BUSTECH NetShuttle.

Some of these alternative configurations may be desirable for other reasons also.

- Because the PCI LAN Adapters are controlled by the EMIO subsystem, there are some significant throughput restrictions as well. An IBM white paper, *Multiprise 3000 Integrated LAN Adapter Feature Performance Report* (GF22-5136) describes some extensive benchmarks measuring throughput on the Multiprise 3000. This white paper found that the outbound (S/390 to network) throughput was limited to 6.3MB/sec and inbound traffic was limited to 2.5MB/sec. This is considerably below the maximum theoretical throughput for a single adapter (32MB/sec total), and the limitation is in the I/O Subsystem processor (the Pentium processor on the SBC). Thus using additional LAN Adapters will not improve total throughput.
- As frequently happens when a major subsystem is stressed by very high utilizations, some systems which make heavy use of the LAN adapters are less stable than normally expected of S/390 systems. IBM has implemented a number of fixes to correct this, but little can be done to improve the total throughput. IBM and others have made some recommendations to insure stability and to provide maximum throughput to the LAN environment.
  - Use the EMIO LAN adapters only for occasional/moderate workload, such as some subset of users doing TN3270 access. Do not do large FTP bulk data transfers across these adapters.
  - Use other LAN attachments, such as channel-connected outboard devices (3172, 2216, Bustech Netshuttle, etc.) for heavy LAN workloads, especially for FTP bulk data.
  - If migrating users from COAX attached devices to LAN-attached PCs accessing the system via the LAN adapters, migrate a few users at a time, and monitor the response times and utilization of the I/O Subsystem Processor carefully.
  - Reduce other use of the EMIO subsystem. Do not use the Emulated DASD subsystem or SCSI-attached tape drives. All of these devices are competing for the same, relatively scarce, resource.
  - Reduce other use of the I/O Subsystem Processor. For example, do not run a SAD (System Activity Display) on the system console. Instead, run this and any other System Element functions on a LAN attached Hardware Management Console (HMC).

LAN attachment via the PCI LAN adapters offers a very flexible and cost-effective means of connecting the Multiprise 3000 to the LAN, but they should not be considered as the only means of doing this. As always, a proper understanding of the system will allow us to exploit its advantages.

Wide Area Network Connectivity

The Multiprise 3000 provides some Wide Area Networking capabilities via two types of internal communications adapters.

1. Up to three Wide Area Connector (WAC) Adapters can be installed in a Multiprise 3000. Each WAC can have either one or two line interfaces, each of which can be either RS232 or V.35, providing a wide range of line speeds, although limited in number.

   On VM/ESA and VSE/ESA systems, the AWSICA Device Manager uses the WAC to emulate an Integrated Communications Adapter (ICA). OS/390 (as well as VM/ESA and VSE/ESA) can use the WAC via the WAN3172 Device Manager to emulate an XCA (External Communications Adapter).

2. Up to two MultiPort Model 2 (MPM2) adapters can be installed. Each MPM2 has eight RS232 interfaces. (The official IBM manuals state a 9600 bps maximum data rate, but many customers have run similar systems at up to 38,400 bps. However, other results may vary.) The MPM2 is supported only by the WAN3172 Device Manager.

WAN Configuration Considerations

Since the maximum number of lines is 18 (two MPM2 and one WAC) and there are rather severe line interface restrictions, many systems will include some form of channel-attached networking unit, such as a 3745, 2216, or some form of Frame Relay. Although the WAN connectivity internal facilities are somewhat limited, there are few restrictions or considerations to be aware of, beyond the number of lines and speeds supported.

- When configuring a system, be sure that the WAC and MPM2 will accommodate the required number and speed/interface of lines needed.
- If the WAN3172 Device Manager is to be used, insure that XCA nodes will provide the required functionality. There are some limitations in XCA capabilities.
- The WAN adapters, with their relatively low data rates, will not impose a significant load on the I/O Support Processor. If other Emulated I/O heavily loads the IOSP, service to the WAN adapters may be delayed enough to occasionally cause timeouts or other I/O errors.

  The solution is, again, to use external, channel connected, communications controllers. External controllers impose no load on the IOSP and are themselves immune to the effects of an overloaded IOSP.
- Don't constrain your thinking to the traditional SDLC lines via external units such as a 3745. There are many alternatives today, including Frame Relay, LAN routers and bridges, virtual private circuits, etc.

Facilities

The Multiprise 3000 is very compact and efficient in terms of power and cooling requirements. This very compactness and lack of special power and cooling needs sometimes causes users to forget some of the basics of installation planning. This section is a brief "refresher course" in physical planning, specifically for the Multiprise 3000. Although we will generally only discuss the primary "System Unit," most comments and suggestions apply equally to any attached expansion enclosures.

Physical Planning Suggestions

Like all current S/390 systems, the Multiprise 3000 will run continuously for years if it is in a "comfortable" temperature and has sufficient electrical power. Let's address the power part of the requirement first.

Electrical Power

The Multiprise has dual power feeds and power supplies. Either feed and power supply can provide all of the power requirements of the system. To the extent reasonably possible, each feed should be plugged into a different power source.

For a system like the Multiprise 3000 that uses a relatively small amount of power, it makes good sense to provide some form of Uninterruptible Power Supply (UPS). Considering the cost of a suitable UPS versus the cost of an unplanned outage, to do otherwise is foolish.

In general, there are three ways of providing UPS protected power to the Multiprise 3000:

1. Install and use the MP3000's "Internal Battery Feature" (IBF).
2. Use an external UPS dedicated to the Multiprise 3000.
3. Connect into a UPS that protects the entire computing environment.

There are specific considerations and suggestions for each alternative.

Internal Battery Feature (IBF): The IBF provides a convenient and compact way to provide power to the system unit, but only to the system unit. There are no electrical outlets to supply power to other components such as the system display or attached I/O devices. Depending on the number of IBFs installed (one or two) and the system configuration, the IBF(s) can power the system for 5 to 25 minutes. Most installations that use the IBF will also have some other form of UPS for the system display and attached I/O.

- Provide some form of "protected power" to the system display and attached I/O devices to allow continued operation of the system.
- Plug each of the two power cords into separate power circuits. Don't allow a popped circuit breaker or blown fuse to cause an outage.
- Occasionally check the power/cooling monitor (on the SE desktop) to verify proper IBF operation.

Dedicated External UPS: A dedicated UPS is frequently the simplest and best answer to power protection. If there are not a lot of attached I/O devices, a 1800 - 3000VA UPS will maintain power for a typical Multiprise 3000 configuration for 10 to 30 minutes.

- The UPS should be sized to provide time to recognize the power problem and then shut the system down cleanly. While the recognition time will depend on the operational environment, most systems can be shut down in 10 to 15 minutes.
- Of the system's two power feeds/plugs, one should be plugged into the UPS and one into the normal "house power". This protects the system against a failure of the UPS itself. Make sure the "house power" plug is on a different circuit from the UPS.
- Use of software such as APC's PowerChute is not supported or recommended by IBM, but may still be useful in some unattended operations environments.

While significant customization work is usually required, some form of automated shutdown on power failure is usually possible.

"House" UPS: Many installations will already have a UPS that protects the entire computing environment, and perhaps even the entire building or facility. While this is probably the best solution, there are still some considerations.

- Many large UPS installations were oriented only toward large computer systems and may not have standard 110V "convenience" outlets. Make sure you have access to 110V outlets from your UPS, or specify the appropriate voltage and power plugs when configuring the Multiprise 3000.
- If the house power includes a powered generator, ensure the switchover to the generator is electrically "clean". Because there are no motor-generators or other power-leveling facilities in the Multiprise 3000, it is susceptible to very short power fluctuations.

Cooling and Other Environmentals

The Multiprise 3000 does not generate large amounts of heat, so little is required in the way of cooling. Normal "office environment" conditions are sufficient for most installations. Care should be taken to provide sufficient air circulation. Do not block the bottom, rear, or front of any of the units.

Although it does not produce very much heat, we do not normally recommend placing the system unit (or expansion enclosures) in an office environment, but rather in some form of computer room. The AMDs (Air Movement Devices, or fans) and the 10,000 RPM disk drives produce enough noise of such frequencies as to be irritating to some people.

Other Installation Considerations

System Display/Keyboard Location: The system comes with display and keyboard/mouse cables sufficient to locate the display about six feet from the system unit. If a more remote location is required, some form of extension cables and/or signal amplifier will be necessary.

Space Requirements: While the system units and expansion units are very compact, they will sometimes require significant service clearance in the front, left side, and rear. While the system is on casters and can be rolled easily, once multiple channel cables are installed, it is much less portable.

Summary

As stated earlier, the Multiprise 3000 is a very powerful, compact, and cost effective computing system. The small size may be deceiving, however. It must be treated as a large, complex "mainframe," not as a "big PC." It deserves the planning, I/O facilities, and systems management of any large mainframe.

The integrated Emulated I/O system provides outstanding flexibility, but care and restraint must be exercised to avoid overusing it to the point of causing system performance and stability issues.

As is frequently the case, the best "insurance" is to have people very familiar with the Multiprise 3000 involved in the planning, configuration, installation/migration, specifically an organization that is expert on P/390 technology and facilities. It is different from all previous systems, and these differences must be understood in order to fully exploit the Multiprise 3000's capabilities.

Additional Information

For more information, refer to the publications mentioned earlier in this article.

- IBM redbook: *Multiprise 3000 Technical Introduction*, SG24-5693
- IBM redbook: *Multiprise 3000 Basic Emulated I/O Definitions*, SG24-5669
- IBM manual: *Multiprise 3000 Emulated I/O Subsystem Users Guide and Reference*, GC38-0610
- IBM white paper: *S/390 Multiprise 3000 Internal Disk Performance*
- IBM white paper: *S/390 Multiprise 3000 Integrated LAN Adapter Feature Performance Report*, GF22-5136

About the Author

Mike Hammock is based in Atlanta, Georgia, and is a Senior Support Representative for *Intelli*Ware Systems, Inc. Mike leads the P/390 technology team for *Intelli*Ware, bringing over 25 years experience from his tenure with IBM, where he last served as the head of the IBM P/390 Competency Center.

Mike can be reached at **mhammock@IntelliWare.com** or at **1-817-277-0800**.

S/390 Multiprise 3000 Enterprise Server at Tompkins County Trust Company

Editor's Note

This brief article is just one example of how the Multiprise 3000 is being used at actual customer locations. It also is another witness to the excellent support provided by IBM Business Partners to VM and VSE customers.

The article was written by T. J. Spivey-Farlow, VP Administration, T. Farlow Associates, Inc. Tompkins Country Trust Company of Ithaca, New York, is a VSE customer whose primary application – Florida Software – is from the Kirchman Corporation, a specialist in banking solutions.

For more information about this and other customer references for the Multiprise 3000, you can contact William Brogren, Programs and Offerings S/390, **bbrogrel@us.ibm.com**.

Working as Advertised

T. Farlow Associates, Inc., an IBM S/390 Business Partner, recently received unsolicited kudos from one of their customers after the installation of a Multiprise 3000 7060-H30 Server, which reduced batch processing time from eight and a half hours to about one hour.

Larry Updike, Senior Vice President of Tompkins Trust Company, said in a letter to Tim Farlow, president of TFA, *I am not in the habit of writing letters of this nature. However, seldom have I been as satisfied over a transaction as I am with this one. Of course, I am referring to your recent installation of the IBM Multiprise 3000 system here.*

Mr. Updike went on to say, *First of all, we are extremely pleased with the performance of the machine. It seems to be rare in this day and age when something actually works as advertised. This machine not only does that, it has exceeded our expectations. It has dramatically slashed our batch processing time (from over eight hours to less than one!) and provides truly instant response on our online terminals. All in a box the size of two suitcases. Amazing!*

Growth at Tompkins County Trust Company

Tompkins County Trust Company, established in 1836, is a high-performing bank which has experienced tremendous growth over the years. The bank held assets amounting to $700 million until the end of 1999, when its holding company, Tompkins Trust Company, acquired two other banks. Today, Tompkins Trust (TMP on the AMEX) holds assets totaling $1.1 billion and processes 95,000 transactions online between 7:30 a.m. and 5:30 p.m. each working day.

Tompkins County Trust Company has been an IBM mainframe shop since 1985, when an NCR system was replaced by an IBM 4361. The mainframe that the Multiprise 3000 replaced was an IBM 9221-150 with 9336 FBA DASD. The 9221 has been in use at the bank for approximately eight years.

According to Terry Barber, Assistant Vice President of Information Systems, not only has the batch update been reduced from eight and a half hours to less than one hour, online processing has improved significantly as well.

The Multiprise 3000 has been very trouble-free, Mr. Barber said. The change of hardware was accompanied by the installation of the latest VSE release and a migration from Fixed Block Architecture to Count Key Data storage. *The entire process went smoothly,* Mr. Barber said.

In closing. I would certainly recommend to any VSE user that they strongly consider upgrading to the Multiprise 3000, Mr. Updike said. *It is truly a paradigm in mainframe processing.*

The integrated features of the Multiprise 3000 upgrade and the migration project met several objectives for Tompkins County Trust, including TCP/IP connectivity (preventing the need for a communications controller) and the conversion from COBOL II Run Time to Language Environment (LE). In addition to very significant and measurable batch and online performance improvements, TFA established a PR/SM environment on the Multiprise 3000 with a testing LPAR.

TFA – Background

T. Farlow Associates, Inc. of Concord, North Carolina, provides 24 x 7 technical systems support services for small to medium-sized businesses across the United States. Founded in 1989, the company incorporated in 1993.

TFA staff perform software upgrades by pregenerating the upgrades on the company's own S/390 systems using a process that has greatly reduced the stress, effort, and inconvenience of software upgrades for TFA customers. In fact, what used to take 50 to 60 hours at a customer site has required as little as a few hours to complete, because so much of the work is done before arriving at a customer's site.

TFA also provides consulting services on hardware upgrades to new platforms and to new peripherals, including tape and disk storage solutions. TFA's partnership with IBM allows them to offer the latest technology and the most appropriate system servers to their customers.

You can contact T. Farlow Associates at **tfarlow@concordnc.com**.

e-business for VSE/ESA Customers – New Information on the VSE Home Page

Author

Anette Stolvoort
VSE Technical Marketing and Sales Support
ahaller@de.ibm.com

Customers, IBM Business Partners and IBMers are asking more and more questions about e-business and VSE/ESA and are looking for additional and recent information in this area. Therefore, we decided to create a whole set of new Web pages which are intended to exactly perform this task – give the latest information about e-business and VSE/ESA. You can find the e-business information by clicking on the link "e-business for VSE/ESA customers" from the VSE home page:

http://www.ibm.com/s390/vse/

The Web pages describe how VSE customers can change their existing business to an e-business, starting with the e-business enablement (e.g. installing TCP/IP) up to highly sophisticated e-business applications which are based on the IBM Application Framework for e-business. The focus throughout the Web pages is put on VSE – what customers can do in the area of e-business natively on VSE or how VSE can be integrated in new e-business applications which might run on an out-board Web server.

Products covered on the Web pages are from IBM and Independent Software Vendors. Products include:

- IBM TCP/IP for VSE/ESA
- MQSeries
- CICS Transaction Gateway and Transaction Server
- DB2, including DB2 Connect
- WebSphere Application Server
- IpServer from Data21, Inc.
- Web-VSE/Host from IntelliWare Systems, Inc.
- etc.

If you are interested in e-business and VSE/ESA, you should have a look at those new pages. If you have questions, comments or suggestions, please directly contact the user ID **vseesa@de.ibm.com**.

Figure 32 on page 117 shows the entry page to the "e-business for VSE/ESA customers" site.

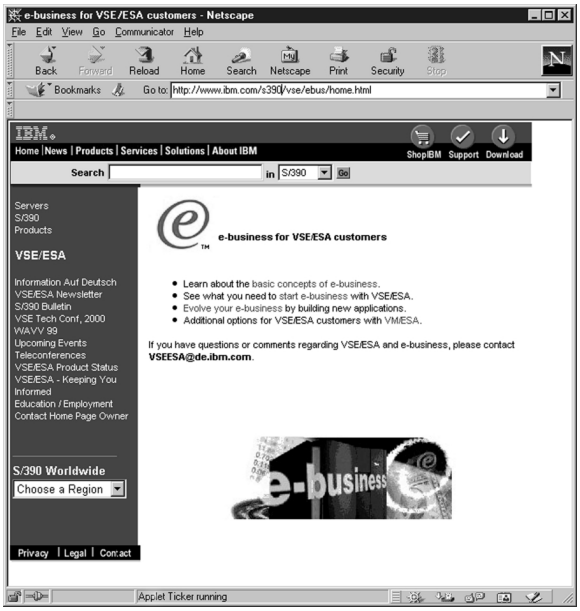


Figure 32. Entry Page to the e-business for VSE/ESA Customers Web Site

IBM Services for VM and VSE

Editor's Note

This article describes some of the offerings from IBM Global Services for customers in IBM's "Central Region" in Europe (Austria, Germany, and Switzerland). Please note that similar offerings are available to VM and VSE customers in other countries. For more information, access the IBM Global Services home page:

http://www.ibm.com/services/

As a VM and VSE customer, you've chosen dependable, solid operating systems. You expect, therefore, a competent, reliable service partner who has a comprehensive set of quality offerings and dependable support for effective exploitation of these systems.

IBM Global Services has the wide range of offerings that you require. In fact, you can select the specific services you need for your own specific requirements. And since IBM Global Services has been a service provider for many years, you get *experienced* support, the support you must have for mission-critical applications.

Keeping Your System "Up"

Offerings from IBM Global Services can increase:

- Your system's *availability* and reduce the amount of effort and time necessary to recover from problems.
- The *effectiveness* of how your system is used.
- The efficient *management* of your system through VM and VSE Remote System Service Management. This offering provides:
  - Regular, planned system service checks in order to maintain high system availability.
  - Fast problem analysis via dump transfers over remote connections.

Taking the "Pain" Out of Installation and Migration

Offerings from IBM Global Services for installation and migration involve IBM specialists who work at your site to attain optimal implementation of your software. Through this support, you can concentrate on your key processes and daily work and keep your business functioning on a normal basis.

Offerings in this area include:

- VSE migration from Release 2.x to 2.4. Transaction Server.
- Installation and migration services for all IBM software products, such as ADSM, DB2, and CICS.

Costs for these offerings are based on standard "service building blocks" which have a fixed price.

Moving into the Future with e-business

IBM Global Services also is your partner for taking advantage of business opportunities through intranets, extranets, and the Internet. Though their expertise, you can reach new markets and new customers. Sample offerings are:

- Web serving for VM systems. This involves preparing your system for a specific Business Partner solution for Web serving and optimal implementation of that solution.
- Enabling TCP/IP for VSE systems, including installation and configuration.
- Implementing the CICS Transaction Server for VSE on VSE/ESA 2.4 systems. This includes configuration and tuning of CICS TS.

Additional Information

Customers in Austria, Germany, and Switzerland can get more, detailed information about the service offerings available to them by sending an e-mail to:

VM_VSE_Services@de.ibm.com

The Global Solutions Directory from PartnerWorld for Developers

Author

James Kingman
Solution Developer Marketing
jkingman@us.ibm.com

Finally, there's a single source for thousands of business solutions. The Global Solutions Directory is a comprehensive online catalog of applications, tools and services created by some of the industry's leading developers using IBM technologies. You can search this Web site free of charge – anytime, day or night – to find the ideal solutions for your business.

With over 36,000 entries, you're bound to find what you're looking for. Each listing gives you product descriptions, company information and even direct links to company Web sites and e-mail addresses. Free-form text searches help you evaluate and compare solutions. And you can search on an incredibly wide variety of parameters – industry, technology, language, solution type, country of availability and many other criteria.

Why wait another minute to find the solution you need? Just log on to the URL shown below, select a language, and start searching the Global Solutions Directory. It's a quick click from there to the perfect solution for your business.

Figure 33. Global Solutions Directory – http://www.ibm.com/software/solutions/isv/. It's easier to find a needle in a haystack when the haystack comes with a multicriteria, searchable database!

If you have any questions, call **1 800 627-8363** in the U.S. & Canada; **+ 1 770 835-9902** worldwide.

Note: The listing of products in the Global Software Directory does not constitute an express or implied recommendation or endorsement by IBM of any particular product, service, company or technology, but is intended as an informational service.

Printed in Germany on chlorine-free bleached paper