CICS Transaction Server for VSE/ESA: CICS Web Support

e-business connector solutions for VSE/ESA

Implementation guidance for CWS, including 3270 Bridge functions

CICS Web Support samples

Erich Amrehn
Gregg Bartowitz
Hans Joachim Ebert
Kenneth Johnsen
Benno Sauer
CICS Transaction Server for VSE/ESA
Web Support and 3270 Bridge

November 2000
Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, “Special notices” on page 109.

First Edition (November 2000)

This edition applies to CICS Transaction Server for VSE/ESA 1.1.1, Program Number 5648-054, for use with VSE/ESA 2.5

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2000. All rights reserved.
Note to U.S Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.
## Contents

**Preface** ................................................................. v
The team that wrote this redbook .................................. v
Comments welcome .................................................. vi

### Chapter 1. Introduction to VSE/ESA connector solutions
1.1 Overview of available connectors ............................. 1
   1.1.1 IBM CICS connectors .................................. 1
   1.1.2 IBM MQSeries connector ............................... 2
   1.1.3 IBM DB2 connector .................................. 2
   1.1.4 IBM VSE/ESA 2.5 connectors .......................... 2
   1.1.5 IBM Host On-Demand ................................ 3
   1.1.6 ISV solutions ........................................ 3
1.2 General selection criteria for a connector .................... 3
1.3 Is CICS Web Support the appropriate connector for you ....... 4
   1.3.1 Characteristics of CICS Web Support ................. 4
   1.3.2 Advantages of CICS Web Support ..................... 4
   1.3.3 Limitations of CICS Web Support .................... 5

### Chapter 2. CICS Web Support environment overview
2.1 CICS Web Support environment prerequisites ................ 7
2.2 CICS Web Support logic and flow ................................ 7
2.3 3270 bridge solution ......................................... 9
2.4 CICS Web Support transactions ................................ 9

### Chapter 3. CWS planning, implementation, and customization
3.1 Implementing CICS Web Support ............................ 11
   3.1.1 VSE/ESA Library for CICS Web Support templates .... 11
   3.1.2 CICS Transaction Server ................................ 12
   3.1.3 TCP/IP ............................................. 14
   3.1.4 Installation verification ......................... 14
3.2 Customizing CICS Web Support ................................ 15
   3.2.1 Analyzer program .................................. 15
   3.2.2 Converter program .................................. 16
   3.2.3 Web error program: DFHWBEP ....................... 17

### Chapter 4. Writing Web applications
4.1 CICS Web Support data flow .................................... 23
4.2 New and changed APIs for CICS Web Support ............... 25
   4.2.1 EXEC CICS WEB API ................................ 25
   4.2.2 EXEC CICS EXTRACT TCP/IP API .................. 26
   4.2.3 EXEC CICS DOCUMENT API ....................... 26
   4.2.4 EXEC CICS WRITEQ/READQ/DELETQ TS ......... 27
4.3 Examples of using CWS APIs .................................. 27
   4.3.1 Example 1: Using CWS on VSE/ESA ................. 28
   4.3.2 Example 2: Starting sample programs from your browser 32
   4.3.3 Sample Web-enabling of existing 3270 program .... 38
4.4 Web page design ................................................. 39
   4.4.1 Describing and handling HTTP and HTML data ..... 39
   4.4.2 Accessing graphics and cataloging them under VSE  40
Preface

CICS Web Support (CWS) is a new function in CICS Transaction Server for VSE/ESA 1.1.1, which has the following software prerequisites:

- VSE/ESA 2.5.0
- TCP/IP for VSE/ESA 1.4.0
- LE/VSE 1.4.1

CWS first became available in OS/390 with CICS TS 1.2, and was enhanced in release 1.3. This 1.3 functionality was ported to CICS TS for VSE/ESA 1.1.1.

CICS Web Support is an effective solution for the VSE/ESA user community. It is a powerful 2-tier Web enablement solution that is easy to plan for and simple to implement. Our goal for this IBM Redbook is to provide you with the information to use this great e-business connector. CWS unites browser technology with S/390. This provides tremendous flexibility to the end-user community, while capitalizing on S/390 performance, reliability, scalability, availability, and data integrity.

This redbook discusses and positions the new CICS TS for VSE/ESA 1.1.1 CICS Web Support (CWS) and 3270 bridge. It provides a broad understanding of the new architecture, together with examples and samples to help customers in their planning and implementation of CWS. CWS employs a unique approach for using an e-business connector in the VSE/ESA environment.

This redbook also discusses planning for CWS, installation, and customization. It provides the guidance you need to design new solutions and upgrade existing solutions.

There are three new publications available with CICS TS 1.1.1 that provide detailed information about CWS:

- *CICS Transaction Server for VSE/ESA Internet Guide Release 1*, SC34-5765

Appendix C, “Related publications” on page 111, contains additional information about these publications and others that may be of interest.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Poughkeepsie Center.

**Erich Amrehn** was the Project Leader for this redbook.

**Gregg Bartowitz** is from IBM United States.

**Hans Joachim Ebert** is from IBM Germany.
Kenneth Johnsen is from IBM United States.

Benno Sauer is from IBM Germany.

Thanks to the following people for their invaluable contributions to this project:

Neville Brailsford, Hugh McGuinness, Sue White
IBM UK, Hursley Laboratories

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Please send us your comments about this or other Redbooks in one of the following ways:

- Fax the evaluation form found in “IBM Redbooks review” on page 117 to the fax number shown on the form.
- Use the online evaluation form found at ibm.com/redbooks
- Send your comments in an Internet note to redbook@us.ibm.com
Chapter 1. Introduction to VSE/ESA connector solutions

This chapter offers an introduction to e-business connector solutions for customers using VSE/ESA, and how CICS Web Support relates to them. We also explain which connectors are used to connect browsers with VSE/ESA applications and data.

1.1 Overview of available connectors

All connectors available for VSE/ESA customers are described with their basic characteristics and grouped according to the host program product. For more information, refer to the redbooks mentioned in Appendix C, “Related publications” on page 111.

1.1.1 IBM CICS connectors

Let us begin with CICS connectors. CICS is available in nearly all VSE/ESA installations and has control over most production data. Therefore, CICS is a good base for connectors.

• **CICS Transaction Gateway (CTG)**

  The CICS Transaction Gateway provides a comprehensive set of Java-based programs for access to CICS applications from a Web browser. These include Java classes and JavaBeans for writing application-specific server programs (servlets) and browser programs (applets).

  The CICS Transaction Gateway is part of CICS Transaction Server for VSE/ESA 1.1.1.

  The CTG is a 3-tier connector model. The middle tier is usually a Netfinity or RS/6000 server with WebSphere installed to provide the Web server, Java Virtual Machine (JVM) and an application server to control the CTG program.

  There are two ways to use CTG:

  – The first way to use CTG is the COMMAREA-based method. CTG receives a request from the browser and sends a back end request including data in a COMMAREA using a CALL to the CICS Universal Client, which is also part of the CTG. The CICS Universal Client transforms this into a standardized External Call Interface (ECI) CALL and sends it with the COMMAREA down an LU 6.2 connection to CICS. This is known as a Distributed Program Link (DPL). After processing the request by CICS, the COMMAREA is sent back to CTG and, from there, data are sent to the browser.

  – The second way to use CTG for a browser is in its capability to act as a converter of HTML data stream to 3270 data stream and vice versa. This mechanism is based on the External Presentation Interface (EPI).

  For more information, visit the following Web site:


• **CICS Web Support (CWS)**

  CWS is a 2-tier connector solution based on HTTP/HTML. This simple but powerful solution provides:
– COMMAREA communication
– 3270 communication

CICS Web Support is covered in more detail in the remaining chapters of this redbook.

1.1.2 IBM MQSeries connector
MQSeries uses a message-based technology to transmit data between different platforms with:

- Assured, fast, and continuous delivery
- Full compatibility between platforms

Transmission is usually asynchronous and can be transmitted using TCP/IP or LU 6.2 links from the middle tier to the host.

The browser can activate a Java Servlet, which contains the MQ-API commands to communicate, by way of the middle-tier MQSeries, with the VSE/ESA MQSeries applications.

For more information, see the Web site at:


1.1.3 IBM DB2 connector
If a customer has DB2 installed under VSE/ESA, VM/ESA, or both, another choice to connect a browser to host-based DB2 data is by using DB2 Connect on a middle tier.

The DB2 connector uses the Distributed Relational Database Architecture (DRDA) to access relational resources. The access from the requestor is by way of standard interfaces such as Java Database Connectivity (JDBC), Open Database Connectivity (ODBC), or Call Level Interfaces (CLIs).

For more information, see the Web site at:

http://www.ibm.com/software/data/db2/db2connect/

1.1.4 IBM VSE/ESA 2.5 connectors
VSE/ESA 2.5 is a “connector” release. Two new connectors are made available with this new release of VSE/ESA and are interesting possibilities to bring together Internet technology with the S/390 world. WebSphere on the middle tier is strongly recommended. For detailed information on the new VSE/ESA connectors see e-business Connectivity for VSE/ESA, SG24-5950.

- DB2 infrastructure
  This connector relies on a DB2 infrastructure and allows easy access to DB2, VSAM, and DL/I data using stored procedures. It is a TCP/IP or LU 6.2-based DRDA connection used to link from a middle tier to the host.

- VSE/ESA native
  The VSE/ESA native connector is TCP/IP-based and uses JavaBeans to transmit requests to a “connection server” within VSE/ESA 2.5, which itself reacts on behalf of the transmitted requests. It is a 3-tier connection.
For more information, see the following Web site at:

http://www.ibm.com/vse

1.1.5 IBM Host On-Demand

Host On-Demand (HOD) is an easy way to bring 3270-based applications to a
browser. HOD itself is installed on a middle tier and is downloaded to a browser
on request (on demand). It supports Secure Sockets Layer (SSL) between the
middle tier and browser.

For more information, see the following Web site at:

http://www.ibm.com/software/network/hostondemand/

1.1.6 ISV solutions

There are increasing numbers of vendor solutions that either use COMMAREA
techniques or convert 3270 data streams. Information about these solutions can
be found at the VSE/ESA home page at:

http://www.ibm.com/vse

1.2 General selection criteria for a connector

The following criteria should be considered when selecting a connector. This is
not a comprehensive list and does not cover all aspects leading to your decision.
It is supplied as a guideline to assist you in making an appropriate selection.

• 2-tier or 3-tier solution

When we talk about 2-tier or 3-tier, we mean a logical connection over 2 or 3
levels.

Two levels implies that a browser is directly served by a server program, often
called a Web server. There is no other server software between the levels,
except TCP/IP.

Three levels means that the server program previously mentioned, becomes a
client itself and accesses another server for conducting application or data
operations.

The number of physical levels may differ from the logical ones. For example, in
OS/390, it is common to run middle tier programs with the “UNIX System
Services” on the same hardware. At the time that this redbook becomes
available, middle-tier products will also be available under Linux for S/390 with
VM/ESA or in an LPAR.

• Type of access

The majority of available connectors are based on Java as the
platform-independent language. Some connectors use HTML only.

• DB2 or non-DB2 on host

If a customer has DB2 installed, it is useful to check for DB2-based
connectors. DB2 Version 7 is shipped with VSE/ESA 2.5. It is now key-enabled
for production use, but can be run in a test mode without a license.
• **Synchronous or asynchronous access**

  *Synchronous* access means that application logic and data transfer are not “buffered”. Most connectors run synchronously.

  Opposite to that, *asynchronous* means that the application logic sends data without regard to established communication links. MQSeries is a well-known example of this.

• **COMMAREA or 3270-based programs**

  The fastest way to connect a browser with VSE/ESA programs and data, is to use existing 3270 programs and have a 3270/HTML converter between VSE/ESA and the browser. Without modifications or enhancements made to the layout of the browser panel, it stays as it was: a 3270 panel.

  COMMAREA-based programs allow a free formatting of the browser panel because these programs do not rely on any 3270 presentation logic.

• **Security aspects**

  Security requirements for browser applications vary greatly, depending on the type of data that is transmitted, on the industry, and on the type of connections being made. Most intranet applications are not critical in this sense.

  On the other hand, banking applications and online shopping applications using the Internet are much more security-sensitive.

• **TCP/IP or LU 6.2 communications protocol**

  Communication from a browser to the first server is based on TCP/IP with HyperText Transfer Protocol (HTTP) and HyperText Markup Language (HTML). This is independent of the link characteristics, whether it be LAN or WAN, Ethernet or Token Ring.

  In 3-tier environments, the connection to the third level can be TCP/IP or LU 6.2, depending on the chosen connector.

### 1.3 Is CICS Web Support the appropriate connector for you

As we mentioned, there are several connectors available to VSE/ESA customers. Which solution should you use to bring application and data to a browser? CICS Web Support is a very interesting connector that needs no additional license, and can be configured in less than one day.

### 1.3.1 Characteristics of CICS Web Support

Some characteristics of the CICS Web Support connector are:

- CWS is a 2-tier model.
- CWS uses TCP/IP.
- CWS deals with an HTTP/HTML-based data stream.
- CWS is independent of other software products on the host, except TCP/IP.
- CWS is always synchronous.
- CWS supports both a COMMAREA approach and a 3270 approach for applications.

### 1.3.2 Advantages of CICS Web Support

We came to the conclusion that CWS is an excellent connector if it meets your requirements. The areas to consider when making your decision include:
• **Operational aspects**
  The entire operation related to CWS is controlled within the VSE/ESA environment. Beyond VSE/ESA, and down to the browser, there is the network only. No other software components are between them. We called it a “centralized point of control”.

• **System programming**
  There are only a few system programmer tasks required before CWS can run. These tasks may take a couple of hours, but once completed, you should see the first CWS test panels on your browser.

• **Application programming**
  As you will see, it is easy to understand and implement applications based on HTTP/HTML, provided you have some related documentation available to use for reference in case of questions. To bring 3270-based applications to a browser with CWS is even easier—try it out!

  **Note:** If it is your intent to implement a Java-based solution, CWS is *not* the connector of choice.

• **Performance**
  As a 2-tier connector, the performance of CWS is likely to be better than as a 3-tier connector.

  Our tests made with COMMAREA and 3270-based applications showed subsecond response times for all tested samples. The browser and CICS were connected by way of a LAN.

• **Debugging**
  From a debugging point of view, CWS is a handy system. What we stated for operational aspects is true for debugging too: it provides a “centralized point of control”.

1.3.3 **Limitations of CICS Web Support**

Note that CICS Web Support (CWS) has the following limitations:

• CWS has limited security for Internet connectors because the Secure Sockets Layer (SSL) is not supported by TCP/IP for VSE/ESA 1.4 and therefore also not supported by CWS. The consequence is that if you use CWS with Internet connections, a middle tier would be needed to provide full security functions.

• Long names (more than 8 characters) used by some tools for HTTP/HTML code generation are not supported by VSE/ESA and therefore also not supported by CWS.

Keep these limitations in mind when deciding if CWS is the appropriate connector for your environment.
Chapter 2. CICS Web Support environment overview

This chapter describes the VSE/ESA environment in which CWS runs and provides an overview of the CWS solution.

2.1 CICS Web Support environment prerequisites

CWS is part of CICS/Transaction Server for VSE/ESA 1.1.1. All of the required products are installed with VSE/ESA 2.5.0. The operating environment and prerequisites for CWS are:

- VSE/ESA 2.5.0
- CICS Transaction Server 1.1.1
- TCP/IP for VSE/ESA 1.4.0.
- LE/VSE 1.4.1

2.2 CICS Web Support logic and flow

To make the right decisions about which facilities to use, and how to customize them best, you need to understand the flow of CWS. This is illustrated in Figure 1.

![Diagram showing the flow of CWS](image)

Figure 1. Calling a program with CWS: Control flow for COMMAREA solution

The flow is explained in the following list:

1. An HTTP request arrives in TCP/IP for VSE/ESA from a Web browser.
2. The sockets listener monitors the TCP/IP for VSE/ESA interface for incoming HTTP requests, based on the port number specified on the TCPIPSERVICE
resource definition. The CICS TS listener transaction is CSOL. CSOL is an internally-defined transaction. It has no RDO definition. It must be defined to BSM or ESM.

3. The sockets listener task attaches Web attach transaction CWXN. CWXN, or its alias, should be specified as the TRANSACTION on the TCPIPSERVICE definition.

4. Web attach processing receives the incoming request and calls DFHCCNV to translate HTTP request headers from ASCII to EBCDIC.

5. Web attach processing links to the user’s analyzer (the analyzer URM).

6. If the analyzer requests conversion, Web attach processing calls DFHCCNV to translate the body of the HTTP request from ASCII to EBCDIC.

7. Web attach processing starts an alias transaction (CWBA or alias) to deal with all further processing of the request in CICS, and then terminates.

8. If the analyzer requests a converter, the alias calls it, requesting the Decode function. Decode can modify the communications area for the CICS program. The converter is a user-written CICS program.

9. The alias calls the CICS program that the analyzer or Decode specified. The communication area passed to the CICS program is the one set up by Decode. If no converter program was called, the communication area contains the entire request.

10. The CICS program processes the request and builds a response (optionally using EXEC CICS WEB WRITE and EXEC CICS WEB SEND commands) and optionally returns output in the communication area.

11. If the analyzer requested a converter, the alias calls the Encode function of the converter, which uses the communication area to prepare the HTTP response. If no converter program was called, the alias assumes that the CICS program has put the desired HTTP response in the communication area.

12. If the analyzer or application requested data conversion, the alias calls DFHCCNV to translate the HTTP response.

13. The alias returns the results to the sockets domain, requests that the socket be closed, and returns.

14. The sockets domain issues a call to TCP/IP for VSE/ESA to send the response.

You may access your CICS TS application from a Web browser with the following Universal Resource Locator (URL):

http://ip.address:port/converter/alias/program

In this URL, note the following explanations:

- *ip.address* is the unique Internet name or IP numerical address (dotted decimal address) of your VSE/ESA system.
- *port* is the CWS listener port defined in the RDO definition TCPIPSERVICE.
- *converter* is the name of the program used for encode and decode processing, use CICS if a converter is not required.
- *alias* is the transaction ID of the alias transaction. CWBA is the supplied alias.
- *program* is the name of the application program to be invoked.
2.3 3270 bridge solution

This feature of CWS allows access to 3270-based CICS applications without 3270 terminals!

For programs that are not BMS, HTML pages are dynamically built from the 3270 data streams. This allows dynamic, two-way translation between 3270 and HTML data. Users can access their applications from any client that supports a standard Web browser. This powerful solution requires no application changes. However, implementation of this CWS feature requires a bridge exit program. A sample bridge exit program is provided with CWS and may be sufficient in your environment.

The bridge exit program is a user-replaceable module. It is the interface between an existing 3270-based user transaction and a client application. With this support, all 3270 terminal I/O requests are intercepted and handled by a bridge exit program. It intercepts all of the 3270 input and output commands issued by the application program, and performs all necessary data transformation and routing.

For BMS programs, this solution utilizes HTML templates which can be generated from existing BMS maps. A second assembly of the BMS map is required with the parameter: TYPE=TEMPLATE added to the DFHMSD macro. The other option would be to pass: SYSPARM=TEMPLATE to the assembler. This generates the HTML code that is presented to the Web browser. The resulting HTML member may be further tailored as needed. For additional information, go to the URL at:

http://ip.address:port/cics/cwba/dfhwbttta/tran

DFHWTBTTA is the program initiating the 3270 bridge operation. It is the “switch” for CWS to utilize the 3270 bridge code.

2.4 CICS Web Support transactions

With CWS, several new programs and transactions have been introduced. Some are found in RDO group DFHWEB. There are four transactions in this group of which you should be aware. They are:

- **CWBA**: CWS alias transaction
- **CWBG**: CWS garbage collection transaction
- **CWBM**: CWS controller transaction
- **CWXXN**: CICS Web attach transaction

In addition, the CICS TS internal transaction CSOL (CWS sockets listener transactions) is very important for CWS.
Chapter 3. CWS planning, implementation, and customization

Because CWS requires planning, testing, and review, we recommend that you establish a separate CICS TS partition. A sample JCL is provided in ICCF library 59 for this purpose. This configuration provides the flexibility to recycle CICS TS when needed. Note that this CICS TS partition must be LE-enabled.

You may also need to consider hardware and software planning, depending on your environment; for example:

- What hardware connections will you use for connectivity to your intranet or to the Internet?
- Do you have a firewall?
- Do you need one?
- Do you already have an IP address for your VSE/ESA 2.5.0 system?
- Which TCP/IP for VSE/ESA ports, outside of the well-known ports, are you using today?
- Will you need to code your own analyzer program, converter program, or Web error program?
- What about security?
- Will you implement the sample security programs, DFH$WBSA, DFH$WBSC, and DFH$WBSN, which are provided with CWS?

Your planning will take more time than the CICS TS and TCP/IP for VSE/ESA “system” changes for the basic implementation of CWS in VSE/ESA.

Our environment consisted of VSE/ESA 2.5.0 running under VM. We used virtual channel-to-channel (VCTCA) connections to VM and TCP/IP. TCP/IP for VSE/ESA 1.4.0 ran in partition F4. Our CICS TS for testing CWS ran in partition F8. The partition size for this CICS TS was 50 MB of storage (yes, this was larger than we needed). From our location, we tested CWS on a system in Poughkeepsie, NY (local LAN, as well as dial-in), and in Munich, Germany. We used multiple browsers at different release levels and noticed subtle differences.

3.1 Implementing CICS Web Support

This section discusses the steps needed to implement CWS. It is easy to implement CWS, with some considerations. It requires two CICS TS modifications, some TCP/IP for VSE/ESA changes, and an understanding of the ramifications of those changes.

3.1.1 VSE/ESA Library for CICS Web Support templates

If the DOCTEMPLATE does not specify a library name, CICS searches the LIBDEF search chain for the member (with the suffix HTML) and in the order specified in the LIBDEF chain.

If the DOCTEMPLATE does specify a library name, CICS searches the sublibrary called DFHDOC in the specified library only (CICS does not search the LIBDEF string) for the member (still with the suffix HTML).
We found that generating a library of five 3390 cylinders provided enough space to handle approximately 1000 HTML templates. In our testing, a simple template was less than 4 KB of storage. Be sure to add library.sublibrary to your CICS TS LIBDEF string. The member names in the VSE library have the extension HTML. For example, one of our test members was EXTRCTF1.HTML.

To enhance performance, consider storing the templates within a VSE/ESA sublibrary on virtual disk.

### 3.1.2 CICS Transaction Server

Let us discuss the CICS Transaction Server (TS) implementation. For this discussion, we assume that this CICS TS is generated from the “C2” JCL sample found in ICCF library 59. We also assume that the “C2” tables (DFHSITC2 and others) are used.

**DFHSIT parameters**

One parameter change is required: TCPIP=NO is the default and must be changed to TCPIP=YES.

WEBDELAY=xx defines after what time, in minutes, the 3270 bridge tasks are purged if they are in a terminal wait state. This will result in a ABRQ transaction abend code.

DOCCODEPAGE=37 code page specifies the default host code page used by the Document Domain. If nothing is specified, the default of 37 will be used.

In addition, there are other parameters that you should consider. For example, another default is GRPLIST=VSELST2. Because up to four CSD lists may be concatenated, we chose to use GRPLIST=(VSELST2,MYLST2). MYLST2 contains our customizations and is processed after VSELST2.

**RDO definitions**

For RDO definitions, there is only one modification to do. The entry is in group DFH$SOT. The type is TCPIPSERVICE. The name is HTTPNSSL. TCPIPSERVICE is the resource definition used to define which TCP/IP services are to use CICS TS internal sockets support. The internal CICS TS service is CWS. By default, the TCPIP listener port number used by HTTPNSSL is 00080.

We recommend that the port number be outside the range of well-known ports, so you should select a port number that is greater than 1024. This will be the port address used for communications between CICS TS and TCP/IP.

The RDO group DFH$SOT is *not* in the default CSD lists. You must copy HTTPNSSL from DFH$SOT, modify it as needed, and reference it in your startup list. When you connect to CICS TS without the TCPIPSERVICE definitions, the following messages will be sent to the VSE console:

- IPN598W Socket request rejected due to improper foundation
- IPN373W Stalled TCP connection has been Flushed, IP:x.xxx.xxx.xxx

These messages are generated by TCP/IP (x in the above message means a valid IP adr).

Other CSD groups to be aware of are DFHWEB, TCPIP, and DFHDOC. These groups are supplied in the default lists VSELIST and VSELST2. These groups are
required for CWS support. All entries needed for CWS support are supplied in the default lists except the TCPIPSERVICE entry, HTTPNSSL. As a result, our second CSD list, MYLST2, contains only the HTTPNSSL entry and our testing entries.

**DOCTEMPLATES**
Use the doctemplates resource definition to define document templates to CICS TS. Document templates are the HTML representations of self-coded information and your BMS maps which are presented to users that access your CICS TS with a browser.

Doctemplates are CICS TS resources and are made available to CICS TS with RDO. The template has to be cataloged into a library according to what we specified in 3.1.1, “VSE/ESA Library for CICS Web Support templates” on page 11. The members in the library.sublibrary have the extension HTML.

**DFHCNV table**
DFHCNV is a user-written conversion table that is called by program DFHCCNV. It is not difficult to code this table. It is used by CWS to convert incoming requests to EBCDIC and outgoing responses back to ASCII for COMMAREA style and 3270-based programs.

The DFHWBHH template is used to translate the HTTP header information, and DFHWBUD is used by the default analyzer for user data conversion. The source code for our DFHCNV table can be found in Appendix A, “Listings” on page 69.

If DFHCNV is not available or contains incorrect definitions of code pages, you will receive ERROR 400 messages at the browser. A missing or incorrect DFHCNV can also produce an ACN1 abend in CWBA, probably followed by other transaction abends.

**Defining all TRANSIDs to BSM or ESM**
For this definition, use II (Interactive Interface) dialog 2.8 and remember to give a CEMT PERF SEC in the CWS partition after you submit job CATSEC. Be sure that transid CSOL is included.

**Virtual storage**
As a guide, CWS requires 500 KB of DSA (24-bit) storage, and 2 MB of EDSA (31-bit) storage. In addition, you need to plan for 1 MB EDSA (31-bit) storage per concurrent user.

Another consideration is temporary storage in CICS TS. Temporary storage queues are heavily used by CWS internally. This will have an impact on the size of your temporary storage dataset, the size of your CICS TS partition, or both.

These two storage considerations affect the size of your CICS TS partition, so plan accordingly.

**CICS TS startup and LIBDEFS**
If you choose to hold the templates in a separate and newly created library, this library must be added to the LIBDEF search chain of your CICS TS startup.
3.1.3 TCP/IP

This section describes the changes you must make to IPINITxx of TCP/IP for VSE/ESA as part of configuring CWS.

Reserving ports for CWS
Consistent with your overall TCP/IP strategy, reserve as many ports as you need for CWS support and ensure that CWS has exclusive use of those ports. For VSE/ESA, be careful when choosing port numbers less than 1024, the “well-known” ports (defined in RDO definition TCPIPSERVICE).

Identify the TCP/IP server
You must identify the TCP/IP for VSE/ESA server by specifying a name for its IP address:

```
DEFINE NAME,NAME=VSEESA.250.IBM.COM,IPADDR=n.n.n.n
```

Here, NAME is a meaningful host name, and IPADDR specifies the dotted decimal address of your VSE/ESA system. If you do not identify the server, CWS will not initialize successfully and the following error message appears during the CICS TS startup:

```
DFHSO0117 applid Unable to determine the TCP/IP host name. Language environment return code X'00000458', reason code X'00000000'.TCP/IP services are unavailable.
```

Specifying a name server
If you want full CICS TS function (that is, use of DFH$WBSN and DFHWBENV), CWS needs to access a name server during its operation:

```
SET DNS1=n.n.n.n
```

Here, n.n.n.n is the dotted decimal address of the name server.

If the name server lookup fails when CICS TS runs, then:

- The security sample program DFH$WBSN does not execute correctly.
- The environment variables program, DFH$WBENV, does not return a connection name in SERVER_NAME. However, it will return the dotted decimal address of the connection and a null string for the value REMOTE_HOST.

Specifying a HTTPD daemon
To specify a HTTPD daemon, as shown in Figure 36 on page 74, use the following definitions:

```
DEFINE HTTPD,ID=HTTP1,ROOT='DFHHTML.DFHDOC',SECURE=NO,CONFINE=NO
```

LIBDEF change for TCP/IP
We added DFHHTML.DFHDOC to the LIBDEF chain of our TCP/IP partition.

3.1.4 Installation verification

At this point, we need to verify the system changes made for the COMMAREA and 3270 bridge feature of CWS. We assume that TCP/IP for VSE/ESA has been recycled, if needed. We also assume that your second CICS TS subsystem is up and running with the appropriate LIBDEFS.

Note: TCP/IP must be active before CICS TS with CWS is started.
**DFH$WB1A for COMMAREA**

A sample application program, DFH$WB1A, is provided to help you test the operation of CWS. From a suitable Web browser, enter the URL that connects to the CICS TS with CWS using an absolute path /CICS/CWBA/DFH$WB1A. The response displays the following message:

DFH$WB1A on system xxxxxxxx successfully invoked through the CICS Web support.

Here, xxxxxxxx is the application ID of the system in which CWS is running.

**CEMT+INQ+TASK for 3270 bridge**

To test the 3270 bridge facility, we used the CEMT transaction. From a suitable Web browser, enter the URL that connects to the CICS TS with CWS as follows:

http://n.n.n.n/cics/cwba/dfhwbtta/CEMT+I+TAS

Note that n.n.n.n is replaced by your valid IP address. Also, to send data on the initial request, use plus signs (+) rather than blanks. CWS interprets a plus (+) sign as a blank. The first “real” blank space in a URL indicates end of data.

**Possible problems during CICS TS startup**

The following CICS TS startup message will be displayed if TCP/IP for VSE is not started or if it is started with neither DNS1 nor DEFINE NAME correctly specified.

DFHSO0117 IYB7ZA02

Unable to determine the TCP/IP host name. Language Environment return code X’00000458’, reason code X’00000000’.

TCP/IP services are unavailable.

### 3.2 Customizing CICS Web Support

CICS TS provides facilities by which you can tailor CWS. You can use the IBM-supplied code for the analyzer, converter, and Web error programs. Why would you want to customize CWS?

Let us suppose that, in your environment, you want to impose rules about which clients can use which services. To deny access, you will need to tailor your own analyzer program.

If you have a need to reformat the inbound or outbound COMMAREA, you will need to tailor your own converter program.

If you want to tailor the default HTTP error response returned by CICS TS so that the response is more meaningful to your users, you will need to tailor your own Web error program.

In the sections that follow, we briefly discuss the analyzer program, the converter program, and the Web error program. Additional information can be found in:

- **CICS Transaction Server for VSE/ESA Internet Guide Release 1, SC34-5765**
- **CICS Transaction Server for OS/390 Version 1 Release 3: Web Support and 3270 Bridge, SG24-5480**

### 3.2.1 Analyzer program

The analyzer program is a user-replaceable program for CWS. Its purpose is to interpret the incoming requests and specify the CICS TS resources that are needed to provide the requested service.
DFHWBADX is the default analyzer for CWS. The source code for the analyzer is supplied in various languages, and you can use it as the basis for your own analyzer. The source members are:

- DFHWBADX (Assembler)
- DFHWBAHX (C)
- DFHWBAOX (COBOL)
- DFHWBALX (PL/I)

If you use a different analyzer, you must specify this in the RDO TCPIPSERVICE.

The default analyzer is written for HTTP requests in which the absolute path is in one of the following four forms:

- /converter/alias/program
- /converter/alias/program?token
- /converter/alias/program/<filename>
- /converter/alias/program/<filename>?token

The default analyzer links to the CICS TS-supplied utility DFHWBUN, to analyze the user data in the communications area passed by the analyzer.

The default analyzer checks the eye-catcher, and then interprets the contents of the absolute path as follows:

- **converter** must be between one and eight characters long. It is converted to uppercase and interpreted as the name of the converter to be called by the alias, unless it has the value CICS, in which case the converter name is set to nulls to show that no converter is to be used.

- **alias** must be between one and eight characters long. It is converted to uppercase and interpreted as the transaction ID of the alias transaction to be used to service the request.

- **program** is the name of your application program and must be between one and eight characters long. It is converted to uppercase and interpreted as the name of the CICS TS program that is to be used to service the request.

- **filename** is an optional parameter when PROGNAME is regarded as a directory. It can be any length, but it must not begin with a slash (/) or contain a question mark (?). It must be made up of the characters allowed in URLs. It is ignored by the analyzer, but is available to the converter or the CICS TS program.

- **token** is an optional 8-character field to be passed in the USER_TOKEN field to the decode function of the converter program.

- If **program** is DFHWBTTA, the filename is treated as the ID of the transaction to be run using the 3270 bridge facility.

Note that CICS TS supports only the HTTP 1.0 Keep-Alive implementation of the persistent connections, not the HTTP 1.1 implementation.

### 3.2.2 Converter program

You can have many converter programs in a CICS TS system to support the operation of CWS. Each converter must provide two functions and the converter program must supply two sections:

- **Decode** is used *before* the CICS TS program is called. It can:
– Use the data from the Web browser to build the communication area in the format expected by the CICS TS program.

– Supply the lengths of the input and output data in the CICS TS program communication area.

– Perform administrative tasks related to the response.

**Encode** is used after the CICS TS program has been called. It can:

– Use the data from the CICS TS program to build the HTTP response and HTTP response headers.

– Perform administrative tasks related to the response.

You might not need to write any converters. If the analyzer or the caller of the CICS TS business logic interface indicates that a converter is not required, the first 32 KB of storage of the request is passed to the CICS TS program in its communication area.

The names of parameters and constants in the communication area are passed to the converter, translated into appropriate forms for the different programming languages supported, and defined in copybooks supplied as part of CWS. The copybook names are listed in Table 1.

<table>
<thead>
<tr>
<th>Language</th>
<th>Parameters copybook</th>
<th>Constants copybook</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembler</td>
<td>DFHWBCDD</td>
<td>DFHWBUCD</td>
</tr>
<tr>
<td>C</td>
<td>DFHWBCHD</td>
<td>DFHWBUCH</td>
</tr>
<tr>
<td>COBOL</td>
<td>DFHWBCDO</td>
<td>DFHWBUCO</td>
</tr>
<tr>
<td>PL/I</td>
<td>DFHWBCDL</td>
<td>DFHWBUCL</td>
</tr>
</tbody>
</table>

These copybooks give language-specific information about the data types of the fields in the communication area. If you use these, you must specify XOPTS(NOLINKAGE) on the translator step; failure to do this causes the compile to fail. If **CICS** is entered as the converter name, no encode or decode occurs.

### 3.2.3 Web error program: DFHWBEP

The Web error program, DFHWBEP, is a user-replaceable module driven by CWS when there is a failure in the processing of a Web request. DFHWBEP allows you to modify the HTTP response issued by CICS TS, or to put out an alternative message.

The parameter list passed to the Web error program contains a pointer to a buffer containing the default HTTP response returned by CICS TS for the error detected, and the length of the response. The Web error program can:

– Leave the response unchanged

– Modify the response returned, and update the length in **WEBEP_RESPONSE_LEN** accordingly

– Use the GETMAIN function to create a new buffer, build a new HTTP response, and pass back the address of the new buffer using the **WEB_RESPONSE_PTR** parameter
The EXEC CICS WEB application programming interface is not available from the Web error program. The data to be returned to the client must be in the buffer addressed by WBEP_RESPONSE_PTR.

The default HTTP response is passed to the Web error program in EBCDIC form. CICS TS assumes that the HTTP response addressed by WBEP_RESPONSE_PTR on exit from the Web error program is in EBCDIC, and performs code page conversion on the response to convert it to ASCII before returning it to the client. The key used for this conversion is that which was selected by the analyzer user-replaceable module. If none was selected, or if the analyzer was not selected before the error occurred, the response is assumed to be in the ISO-8859-1 code page (Latin -1).

On input, DFHWBEP receives:

- **WBEP_LENGTH**: The length of the DFHWBEP DSECT.
- **WBEP_EYCATCHER**: An eye catcher which is greater than “wbepca”.
- **WBEP_VERSION**: Version of DFHWBEP DSECT being passed by CICS. Should be X'001'.
- **WBEP_ERROR_CODE**: The binary number indicating the cause of the original error. Constants which this field may contain can be found in the copybook DFHWBUCD.
- **WBEP_ABEND_CODE**: The CICS abend code associated with the error.
- **WBEP_MESSAGE_NUMBER**: The message number, a pointer to the text and the length of the CICS WB message associated with the error. The message is also written on a TD queue CWBO.
- **WBEP_MESSAGE_PTR**: See the WBEP_MESSAGE_NUMBER description.
- **WBEP_MESSAGE_LEN**: See the WBEP_MESSAGE_NUMBER description.
- **WBEP_CLIENT_ADDRESS**: The 15-byte field containing the TCP/IP address of the client and the length of the address contained in this field.
- **WBEP_CLIENT_ADDRESS_LEN**: See the WBEP_CLIENT_ADDRESS description.
- **WBEP_SERVER_ADDRESS**: The 15-byte field containing the TCP/IP address of the server and the length of the address contained in this field.
- **WBEP_SERVER_ADDRESS_LEN**: See the WBEP_SERVER_ADDRESS description.
- **WBEP_TCPIService_NAME**: Name of the TCPIPSERVICE associated with the failing request.
- **WBEP_CONVERTER_PROGRAM**: Name of the converter program, if one is used, associated with the failing request.
- **WBEP_TARGET_PROGRAM**: Name of the target program associated with the failing request.
- **WBEP_FAILING_PROGRAM**: The program in which the error occurred.
- **WBEP_HTTP_RESPONSE_CODE**: The HTTP error response code CICS is returning for this error. This response code is also contained in the buffer containing the HTTP response that is passed to DFHWBEP, where it can be overridden by DFHWBEP.
• **WBEP_ANALYZER_RESPONSE**: Response and reason code returned by the analyzer program.

• **WBEP_ANALYZER_REASON**: See the WBEP_ANALYZER_RESPONSE description.

• **WBEP_CONVERTER_RESPONSE**: Response and reason code returned by the converter program.

• **WBEP_CONVERTER_REASON**: See the WBEP_CONVERTER_RESPONSE description.

Parameters common to input and output are:

• **WBEP_RESPONSE_PTR**: On input, a pointer to and length of the default HTTP error response to be returned to the HTTP client. The HTTP response is contained in a 32 K buffer. DFHWBEP can change or append the HTTP response message. If the buffer is too small, DFHWBEP can GETMAIN a new buffer.

  On output, a pointer to and the length of the changed HTTP error response. If no new buffer is GETMAINed, the buffer pointer can stay unchanged, and only the length field has to be updated.

• **WBEP_RESPONSE_LEN**: See the WBEP_RESPONSE_PTR description.

The DSECT or copybooks to describe the fields in the COMMAREA are provided by CICS and have the following names:

• **DFHWBEPD**: Assembler

• **DFHWBEPH**: C

• **DFHWBEPO**: COBOL

• **DFHWBEPL**: PL/I

Table 2 provides the DSECT layout of the DFHWBEP parameters.

**Table 2. DFHWBEP DSECT layout**

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Field name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000</td>
<td>D204</td>
<td>DFHWBEPC DSECT</td>
<td></td>
</tr>
<tr>
<td>000000</td>
<td>D12,X’C’</td>
<td>WBEP_PREFIX</td>
<td>Input</td>
</tr>
<tr>
<td>000000</td>
<td>2</td>
<td>WBEP_LENGTH</td>
<td>Input</td>
</tr>
<tr>
<td>000002</td>
<td>8</td>
<td>WBEP_EYECATCHER</td>
<td>Constant</td>
</tr>
<tr>
<td>00000A</td>
<td>2</td>
<td>WBEP_VERSION</td>
<td>Input</td>
</tr>
<tr>
<td>00000C</td>
<td>D112</td>
<td>WBEP_DATA</td>
<td></td>
</tr>
<tr>
<td>00000C</td>
<td>2</td>
<td>WBEP_ERROR_CODE</td>
<td>Input</td>
</tr>
<tr>
<td>00000E</td>
<td>2</td>
<td>Not currently used</td>
<td></td>
</tr>
<tr>
<td>000010</td>
<td>4</td>
<td>WBEP_ABEND_CODE</td>
<td>Input</td>
</tr>
<tr>
<td>000014</td>
<td>4</td>
<td>WBEP_MESSAGE_NUMBER</td>
<td>Input</td>
</tr>
<tr>
<td>000018</td>
<td>4</td>
<td>WPEP_MESSAGE_PTR</td>
<td>Input</td>
</tr>
<tr>
<td>00001C</td>
<td>4</td>
<td>WPEP_MESSAGE_LEN</td>
<td>Input</td>
</tr>
</tbody>
</table>
Section A.7, “Source listing DFHWBEP sample program” on page 99, contains the source code we used. The program first checks to see if the HTML tag </body> is present within the length of the response that is given as input. If yes, it overwrites this tag with the information that will be added to the buffer, and writes a new one at the end. If the tag is not present, it starts writing immediately after the response in the buffer.

There is no checking for length of the returned error message, because 32 KB should be large enough for what we add. The program checks all DFHWBEP DSECT input fields, and if one is present, it adds the information from the DSECT to the response buffer.

Figure 2 shows the output of DFHWBEP for a Program Not Found error.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Field name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>000020</td>
<td>4</td>
<td>WBEP_RESPONSE_PTR</td>
<td>Input/Output</td>
</tr>
<tr>
<td>000024</td>
<td>4</td>
<td>WBEP_RESPONSE_LEN</td>
<td>Input/Output</td>
</tr>
<tr>
<td>000028</td>
<td>1</td>
<td>WBEP_CLIENT_ADDRESS_LEN</td>
<td>Input</td>
</tr>
<tr>
<td>000029</td>
<td>D15/X'F'</td>
<td>WBEP_CLIENT_ADDRESS</td>
<td>Input</td>
</tr>
<tr>
<td>000038</td>
<td>1</td>
<td>WBEP_SERVER_ADDRESS_LEN</td>
<td>Input</td>
</tr>
<tr>
<td>000039</td>
<td>D15/X'F'</td>
<td>WBEP_SERVER_ADDRESS</td>
<td>Input</td>
</tr>
<tr>
<td>000048</td>
<td>8</td>
<td>WBEP_TCPIPSERVICE_NAME</td>
<td>Input</td>
</tr>
<tr>
<td>000050</td>
<td>8</td>
<td>WBEP_CONVERTER_PROGRAM</td>
<td>Input</td>
</tr>
<tr>
<td>000058</td>
<td>8</td>
<td>WBEP_TARGET_PROGRAM</td>
<td>Input</td>
</tr>
<tr>
<td>000060</td>
<td>8</td>
<td>WBEP_FAILING_PROGRAM</td>
<td>Input</td>
</tr>
<tr>
<td>000068</td>
<td>4</td>
<td>WBEP_HTTP_RESPONSE_CODE</td>
<td>Input</td>
</tr>
<tr>
<td>00006C</td>
<td>4</td>
<td>WBEP_ANALYZER_RESPONSE</td>
<td>Input</td>
</tr>
<tr>
<td>000070</td>
<td>4</td>
<td>WBEP_ANALYZER_REASON</td>
<td>Input</td>
</tr>
<tr>
<td>000074</td>
<td>4</td>
<td>WBEP_CONVERTER_RESPONSE</td>
<td>Input</td>
</tr>
<tr>
<td>000078</td>
<td>4</td>
<td>WBEP_CONVERTER_REASON</td>
<td>Input</td>
</tr>
</tbody>
</table>
404 Program Not Found

MEANINGFUL FIELDS FROM DFHWBEP COMMAREA:

WBEP_ERROR_CODE: dfhwbbli_link_failed_parameters

WBEP_MESSAGE: DFHWB0101 10/20/00 14:13:21 A0005C22.CWBA The CICS Web Support alias program DFHWBA detected a failure in program DFHWBBLI. Host IP address: 9.10.21.200. Client IP address: 9.82.23.49. TCPIPSERVICE: HTTPSSL

WBEP_CLIENT_ADDRESS: 9.82.23.49

WBEP_SERVER_ADDRESS: 9.10.21.200

WBEP_TCPIPSERVICE_NAME: HTTPSSL

WBEP_TARGET_PROGRAM: EXTRCT5

WBEP_FAILING_PROGRAM: EXTRCT5

*Figure 2. DFHWBEP added text to the standard error response*
Chapter 4. Writing Web applications

This chapter describes the data flow of CICS Web Support programs and gives some basic information about HTTP and HTML. In addition, new and changed APIs are explained through examples using sample code and listings. Refer to Appendix A, “Listings” on page 69, for a complete collection of code and listing samples.

4.1 CICS Web Support data flow

A design principle of CICS applications should be to have the business logic separated from the presentation logic. Use the COMMAREA command, EXEC CICS LINK, to link to the business logic, as shown in Figure 3. If the business logic is separated, you can use different presentation methods like 3270 or CWS. The recommended method for CICS Web applications is to use the new Web API, which is described in 4.2, “New and changed APIs for CICS Web Support” on page 25. Examples of using the new Web API are given in 4.3, “Examples of using CWS APIs” on page 27.

In order to make decisions about which facilities to use and how to customize them, you need to understand how data is passed within CICS Web Support. Figure 4 on page 24 shows the data flow from client through CICS and back, with the communications area used. Be aware that Figure 4 on page 24 is a different view than what is described in 2.2, “CICS Web Support logic and flow” on page 7. The following figure concentrates on communications area handling.
The flow shown in Figure 4 is explained here:

1. A request arrives from a client, and the CICS sockets listener transaction, CSOL, starts the Web attach transaction, CWXN, and then reads the request into CICS temporary storage.

2. The DFHCCNV program translates the HTTP headers from ASCII into EBCDIC.

3. The DFHCCNV program translates the HTTP user data from the client code page into EBCDIC.

4. The Decode function of the converter constructs the communication area for the CICS program. This communication area can be constructed in-place in the buffer provided by CICS. Decode can create a new buffer, or it can use the EXEC CICS WEB application programming interface to retrieve the parts of the incoming request.

5. The CICS program updates the communication area.

6. The Encode function of the converter constructs the HTTP response to be sent to the client. The response can be constructed in-place in the communication area.

   Encode can free the communication area and get a new buffer for the response, or it can use the new Web application programming interface to construct an HTTP response. The response consists of headers and user data. You can make your response longer than 32 K, as described in CICS Transaction Server for VSE/ESA Internet Guide Release 1, SC34-5765.
7. The DFHCCNV program translates the headers from EBCDIC to ASCII.
8. The DFHCCNV program translates the user data from EBCDIC to the client code page.
9. The alias sends the response to the client and frees the storage.

### 4.2 New and changed APIs for CICS Web Support

The following Application Programming Interface verbs have been changed or introduced with the new level of CICS Transaction Server for VSE/ESA 1.1.1. A short explanation about the following new and changed APIs is given in this chapter:

- EXEC CICS WEB
- EXEC CICS TCPIP
- EXEC CICS DOCUMENT
- EXEC CICS WRITEQ TS, READQ TS and DELETEQ TS


It is important to note that sending data to a task, and receiving data from a task, is a totally new approach for CICS systems.

Prior to CWS, a task that was started received data in two ways:

- Using a terminal input/output area for a terminal-attached task.
- Using mechanisms like COMMAREAs, READQ TD, and RETRIEVE. This input was provided by CICS programs which run prior to the attached task.

With CWS, all tasks are non-terminal tasks, but data is exchanged with a “terminal” from a logical point of view, called a “browser”. To cover this new situation, EXEC CICS WEB was introduced with parameters such as RECEIVE and SEND, and so on. This means that certain application programming interface commands cannot be used. Those commands include:

- Terminal control commands that refer to the principal facility
- Options of EXEC CICS ASSIGN that return terminal attributes
- BMS commands
- Signon and signoff commands

Prior to CICS TS, it was not possible to debug non-terminal tasks with CEDF. To allow CWS tasks to be debugged interactively, the new CICS with CEDX transaction has been developed. More information about CEDX can be found in Chapter 7, “Problem determination and application debugging” on page 67.

#### 4.2.1 EXEC CICS WEB API

The EXEC CICS WEB API should read and send HTTP responses, built with EXEC CICS DOCUMENT commands. The request and response APIs are:

- **WEB SEND**: Selects a document for delivery by CICS Web Support.
- **WEB RECEIVE**: Receives the body of an HTTP request from CICS Web Support into an application supplied buffer.
- **WEB WRITE**: Allows the application to add HTTP header information to the response.

- **WEB READ FORMFIELD**: Retrieves the value of a specific field from an HTML form, the name of which is given on the request of the FORMFIELD parameter. The form is part of the body of an HTTP request being processed by the current CICS task.

- **WEB READ HTTPHEADER**: Extracts the HTTP header information.

- **WEB READNEXT FORMFIELD**: Retrieves the next name-value pair in an HTML form.

- **WEB READNEXT HTTPHEADER**: Retrieves the next HTTP header in the list of headers.

- **WEB ENDBROWSE FORMFIELD**: Terminates the browse of a set of name-value pairs in an HTML form. The form is part the body of an HTTP request being processed by the current CICS task. No information is returned on the ENDBROWSE.

- **WEB ENDBROWSE HTTPHEADER**: Terminates the browse of HTTP headers. No information is returned on the ENDBROWSE.

- **WEB EXTRACT**: Allows the application to obtain additional information about an inbound request from a client.

- **WEB RETRIEVE**: Command retrieves the DOCTOKEN of the document which was sent using an earlier WEB SEND command.

- **WEB STARTBROWSE FORMFIELD**: Signals the start of a browse of a set of name-value pairs in an HTML form that is part of the body of an HTTP request being processed by the current CICS task. Pay close attention to the data conversion and code pages used.

- **WEB STARTBROWSE HTTPHEADER**: Signals the start of a browse of the HTTP header information.

### 4.2.2 EXEC CICS EXTRACT TCPIP API

The EXEC CICS EXTRACT TCPIP API allows Web applications to retrieve TCP/IP-related information about the client and the server. While using this API, a INVREQ condition may occur. To handle the INVREQ condition, we recommend you use EXEC CICS HANDLE CONDITION INVREQ for COMMAREA-based programs.

### 4.2.3 EXEC CICS DOCUMENT API

The EXEC CICS DOCUMENT API allows you to build up output data areas, which are stored in the temporary storage area, and send the data to the browser using the EXEC CICS WEB API. This API was introduced to put together static and dynamic data, including graphics, in one data stream.

- **DOCUMENT CREATE**: Signals the start of the document creation process. The document being created can be an empty document, or it can be based on an existing document, a template, or data contained in an application buffer.

- **DOCUMENT INSERT**: Allows the application to insert document objects at insertion points within the document. The insertion points (bookmarks) define relative positions within the document. Bookmarks must be defined before
being referenced. Data is always inserted after the position identified by the bookmark.

- **DOCUMENT RETRIEVE**: Allows the application to obtain a copy of the document in its own buffer, which it can then manipulate directly. The document is managed by CICS, and the application does not have direct access to the buffer containing the contents of the document.

  The document exists only for the duration of the current transaction, so the application must retrieve the document and store it if the document is to exist over transaction boundaries. The retrieved document can be used as a basis for a new document by using the FROM option of the DOCUMENT CREATE command.

- **DOCUMENT SET**: Allows the application to add symbols and their associated values to the symbol table. If the symbol being added already exists in the table, it is replaced by the new definition.

### 4.2.4 EXEC CICS WRITEQ/READQ/DELETQ TS

Because of the internal usage of temporary storage by CWS, the length of the TSQUEUE names has been increased to 16 characters. Therefore, a new option (QNAME) has been introduced. QNAME(name) is an alternative to QUEUE. QNAME specifies the symbolic name (1 to 16 characters) of the queue. This affects EXEC CICS READQ TS, WRITEQ TS, and DELETQ TS.

### 4.3 Examples of using CWS APIs

The sample program DFHSWB1A is using the COMMAREA technique for its output. The sample programs (EXTRCT1 through EXTRCT4) are using EXEC CICS WEB and EXEC CICS DOCUMENT, utilizing temporary storage instead of the communication area (COMMAREA).

This section provides two CWS COMMAREA examples and explains how they work. Table 3 describes the sample (HTML) documents and programs used throughout this chapter.

All CICS assembler sample programs and the associated HTML documents can be found in A.4, “Listing of program samples including JCL” on page 75.

Table 3. Our supplied HTML documents and CICS sample programs

<table>
<thead>
<tr>
<th>Documents/Programs</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX.HTML</td>
<td>sent by TCP/IP</td>
</tr>
<tr>
<td>DEMO.HTML</td>
<td>sent by TCP/IP</td>
</tr>
<tr>
<td>EXTRCTH1.HTML</td>
<td>header for sample EXTRCT1</td>
</tr>
<tr>
<td>EXTRCTF1.HTML</td>
<td>footer for sample EXTRCT1</td>
</tr>
<tr>
<td>EXTRCTH2.HTML</td>
<td>header for sample EXTRCT2</td>
</tr>
<tr>
<td>EXTRCTF2.HTML</td>
<td>footer for sample EXTRCT2</td>
</tr>
<tr>
<td>EXTRCTH3.HTML</td>
<td>header for sample EXTRCT3</td>
</tr>
<tr>
<td>EXTRCTF3.HTML</td>
<td>footer for sample EXTRCT3</td>
</tr>
<tr>
<td>EXTRCTH4.HTML</td>
<td>header for sample EXTRCT4</td>
</tr>
</tbody>
</table>
Before using this example, the following actions are required:

1. Compile EXTRCT1 as an online program and catalog it to the previously defined DFHHTML.DFHDOC sublibrary (use II, option 8 for job generation).

2. Catalog the header and footer documents (EXTRCTH1 and EXTRCTF1) in sublibrary DFHHTML.DFHDOC as `name. HTML`. For a sample catalog job, see Figure 58 on page 90. Change the `n.nn.nn.nn` to your defined IPADDR.

3. Define the DOCTEMPLATE (EXTRCTH1 and EXTRCTF1) with CEDA DEFINE DOCTEMPLATE (see Figure 5 for a sample of the RDO definition).

4. Install the group where the DOCTEMPLATES are in. There is no need to define the EXTRCT1 program in CICS, if AUTOINSTALLATION for programs is ON.

5. Catalog and link all COMMAREA programs with AMODE(31) and RMODE(ANY) because CWBA (transaction where user programs are running) has TASKDATLOC(ANY) defined.

6. Check EIBCALEN in your program, whether the length of the COMMAREA is as expected (not a length of zero).

**Note:** The screen layout presented by the browser could vary from one browser to another or even from one browser release to another.

**Example 1 uses the following sample files:**

- HTML header document, EXTRCTH1 (see Figure 40 on page 76)
- HTML footer document, EXTRCTF1 (see Figure 41 on page 77)
- CICS assembler source program, EXTRCT1 (see Figure 6 and Figure 7)
Figure 5. Sample RDO definition for DOCTEMPLATE EXTRCTH1
Figure 6. CICS assembler source of EXTRCT1 (Part 1 of 2)
The numbers appearing down the right-hand side of Figure 6 on page 30 and Figure 7 correspond to the items in the following list:

1. Creates document with the first part of the HTML template (header template ‘EXTRCTH1’).

2. Retrieves information from TCP/IP related control block.

3. Inserts the HTML tag “<p>Client Name :” into the document. This tag generates a new paragraph and inserts “Client Name :” as text.

4. Inserts the client name that we received from TCP/IP EXTRACT into the document. The client name is inserted right after the text “Client Name :”.

5. Inserts the HTML tag “<p>Server Name :” into the document. This tag generates a new paragraph and inserts “Server Name :” as text.

6. Inserts the server name that we received from TCP/IP EXTRACT into the document. The server name is inserted right after the text “Server Name :”.

7. Inserts the HTML tag “<p>TCPIPSERVICE name: ” into the document. The tag generates a new paragraph and inserts “TCPIPSERVICE name:” as text.

8. Inserts the TCP/IP service name that we received from TCP/IP EXTRACT into the document. The TCP/IP service name is inserted right after the text “TCPIPSERVICE name:”.

9. Inserts the HTML tag “<p>Server Name :” into the document. This tag generates a new paragraph and inserts “Server Name :” as text.

10. Inserts the server name that we received from TCP/IP EXTRACT into the document. The server name is inserted right after the text “Server Name :”.

11. Inserts the HTML tag “<p>TCPIPSERVICE name: ” into the document. The tag generates a new paragraph and inserts “TCPIPSERVICE name:” as text.

12. Inserts the TCP/IP service name that we received from TCP/IP EXTRACT into the document. The TCP/IP service name is inserted right after the text “TCPIPSERVICE name:”.

* Insert TCPIPSERVICE being used

EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) TEXT(TCPSINFO) LENGTH(TCPSINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) TEXT(PORTINFO) LENGTH(PORTINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) TEXT(PRINUMB) LENGTH(L’PRINUMB)

EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) TEMPLATE('EXTRCTF1')

* Send complete document

EXEC CICS WEB SEND DOCTOKEN(DOCTOKEN)
CLIENTCODEPAGE('iso-8859-1')
EXEC CICS RETURN

CLNTINFO DC C’<p>Client Name : ’
CLNTINFL DC F’17’
SRVRINFO DC C’<p>Server Name : ’
SRVRINFL DC F’17’
ADDRINFO DC C’<p>Client Address : ’
ADDRINFL DC F’20’
ADDRINFO DC C’<p>Server Address : ’
ADDRINFL DC F’20’
TCPINFO DC C’<p>TCPIPSERVICE name: ’
TCPSINFL DC F’22’
PORTINFO DC C’ - on Port Number: ’
PORTINFL DC F’19’

END EXTRCT1

/*
Inserts the HTML footer ‘EXTRCTF1’ to the document. This completes our document with all required tags.

Next, the document is sent to the browser and translated with the specified code page.

Now we are ready to use the EXTRCT1 program. To start EXTRCT1, enter a URL like the following example on your browser:

http://n.nn.nn.nnn:1080/cics/CWBA/EXTRCT1

For this URL, note the following explanations:

- **http** is the protocol you want the browser to use.
- **n.nn.nn.nnn** is the TCP/IP address for the VSE/ESA system, where CICS is running.
- **1080** is the port number you specified in the TCPIPSERVICE definition.
- **cics** means you do not want to use a converter.
- **CWBA** is the CICS Web transaction.
- **EXTRCT1** is the name of your program.

The image you should see on your browser is shown in Figure 8.

---

**EXTRACT - Running CICS Web Support on VSE/ESA**

with no image help at all

Client Name: TST1
Client Address: 9.37.3.16
Server Name: SJMV/ESA
Server Address: 9.12.13.29

TCPIPSERVICE name: HTTPSSSL - on Port Number: 01080

---

**Figure 8. HTML sent by the EXTRCT1 program**

4.3.2 Example 2: Starting sample programs from your browser

Example 2 shows an easier way to start our sample programs. Before using this example, you must do the following:

1. Catalog all sample HTML documents into the DFHHTML.DFHDOC sublibrary.
2. Compile, link, and catalog all CICS assembler programs into that same sublibrary.

3. Define documents EXTRCTH1, EXTRCTH2, EXTRCTH3, and EXTRCTH4 as DOCTEMPLATE in CICS TS.

4. Define documents EXTRCTF1, EXTRCTF2, EXTRCTF3, and EXTRCTF4 as DOCTEMPLATE in CICS TS.

5. Catalog DEMO.HTML and INDEX.HTML documents into the sublibrary, which is defined in the “TCP/IP HTTP daemon definition” as ROOT sublibrary. In our environment, it is also “DFHHTML.DFHDOC”. The HTTP daemon looks for the member INDEX.HTML as the default document unless the Web user enters a different file name in the URL.

As mentioned earlier, an easier way to start our sample programs is to type the URL http://n.nn.nn.nnn on the browser. This forces the HTTP daemon to look for the member INDEX.HTML in the ROOT sublibrary, and send it to the browser (see Figure 9).

![Figure 9. INDEX.HTML sent by HTTP daemon](image)

At this INDEX page, you can click here to receive the DEMO.HTML page (see Figure 10 on page 34).
ES4250 - Running CICS Web Support on VSE/ESA

Press DFH$WB1A to see the CICS Web Support sample application in action.

What is actually passed to the Web Browser and thus to CICS Transaction Server for VSE/ESA is:

"http://nn.nn.nn:1080/cics/cwsbar/DFH$WB1A"

Press EXTRCT1 to see the CICS Web Support running an application that will call the EXEC CICS EXTRACT command to determine exactly what CICS environment the program is executing in.

What is actually passed to the Web Browser and thus to CICS Transaction Server for VSE/ESA is:

"http://nn.nn.nn:1080/cics/cwsbar/EXTRCT1"

This page was updated September 2000.

Figure 10. DEMO.HTML sent by HTTP daemon

At the DEMO page, you can choose to run the DFH$WB1A sample or the EXTRCT1 sample. If you click EXTRCT1, you are presented a page with two additional options as shown in Figure 11.

EXTRACT - Running CICS Web Support on VSE/ESA

with no image help at all

Client Name: TST1
Client Address: 9.37.3.36
Server Name: SIVM/VSE
Server Address: 9.12.12.29
TCPBPSERVICE name: HTTPNSL - on Port Number: 01080

Click here for a MastHead
Click here for a MastHead and Backdrop
Return

Figure 11. EXTRCT1.HTML sent by sample program EXTRCT1

The first of the additional options is Click here for a Masthead, which starts the EXTRCT2 program and displays the page shown in Figure 12 on page 35.
The second option, *Click here for a MastHead and Backdrop*, starts the EXTRACT3 program and display the page shown in Figure 13.

![CICS Web Interface](image)

**EXTRACT - Running CICS Web Support on VSE/ESA**

with a masthead

- Client Name: TST1
- Client Address: 9.37.3.55
- Server Name: SJMVSE
- Server Address: 9.12.13.29
- TCP/IP SERVICE name: HTTPSSL - on Port Number: 01080

![CICS Web Interface](image)

**EXTRACT - Running CICS Web Support on VSE/ESA**

with a masthead and a background

- Client Name: TST1
- Client Address: 9.37.3.86
- Server Name: SJMVSE
- Server Address: 9.12.13.29
- TCP/IP SERVICE name: HTTPSSL - on Port Number: 01080

The page shown in Figure 13 also allows you to select one of six backdrop boxes that will start sample program EXTRACT4.
Figure 14 shows parts of EXTRCT4.

```
* EXTRCT4 CSECT
EXTRCT4 AMODE 31
EXTRCT4 RMODE ANY
   EXEC CICS HANDLE CONDITION INVREQ(DOCINSRT) 1
   EXEC CICS DOCUMENT CREATE DOCTOKEN(doctoken) * 1
      TEMPLATE('EXTRCTH4')
      ENDBODY
* DOCINSRT EXEC CICS WEB READ FORMFIELD(TEXTURE) NAMELENGTH(L'TEXTURE') * 2
     VALUE(TEXT) VALUELENGTH(TEXTL) NOHANDLE
     CLI TEXT,'C'1'
     BE TEXT1
     CLI TEXT,C'6'
     BE TEXT6
     ENDBODY
TEXT1 EQU *
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 4
      TEXT(TEXT1NFO) LENGTH(TEXT1NFL)
      B ENDBODY
TEXT6 EQU *
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 4
      TEXT(TEXT6NFO) LENGTH(TEXT6NFL)
      B ENDBODY
* Close Body
ENDBODY EQU *
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 5
      TEXT(BODYINFO) LENGTH(BODYINFL)
* Insert image name
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 6
      FROM(IMAGINFO) LENGTH(IMAGINFL)
* Insert header
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 7
      FROM(HEADER) LENGTH(HEADERL)
* Insert client name
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 8
* Insert footer for document
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 9
      TEMPLATE('EXTRCTF4')
* Send complete document
   EXEC CICS DOCUMENT INSERT DOCTOKEN(doctoken) * 10
      CLNTCODEPAGE('iso-8859-1')
   EXEC CICS RETURN

   IMAGINFO DC C'&lt;img src="/DFHWBIMG/masthead.gif"&gt;'
   IMAGINFL DC F'41'
   HEADER DC C'&lt;h1&gt;EXTRACT - Running CICS Web Support on VSE/ESA&lt;/h1&gt;'
   DC C'&lt;p&gt;with a masthead and a background'
   DC C'&lt;p&gt;&lt;hr&gt;'
   HEADERL DC A(*-HEADER)
   TEXTINFO DC C' background="/DFHWBIMG/texture1.jpeg"'
   TEXTINFL DC A(*-TEXTINFO)
   TEXT6INFO DC C' background="/DFHWBIMG/texture6.jpeg"'
   TEXT6NFL DC A(*-TEXT6INFO)
   BODYINFO DC C'&gt;'
   BODYINFL DC F'1'
   TEXTURE DC C'TEXTURE'
   END EXTRCT4
```

Figure 14. Parts of sample EXTRCT4 program
If you click one of the boxes in Figure 13 on page 35 to change the backdrop, program EXTRCT4 follows this process:

1. Creates a document with the first part of the HTML template (header template EXTRCTH4).
2. Reads in the FORMFIELD(TEXTURE) and VALUE(TEXT) at the “CLICK”.
3. Analyzes the “CLICK”.
4. Inserts the background you selected into the document.
5. Completes the body by adding the greater than (>) sign and inserts it into the document.
6. Inserts the HTML img tag into the document.
7. Inserts additional HTML tags into the document.
8. Inserts client name, and so on, into the document.
9. Completes the document by inserting the footer template (EXTRCTF4) into the document.
10. Sends the complete document to the browser with the selected backdrop and masthead image.

Figure 15 shows you the HTML sent by sample program EXTRCT4, when box 1 was selected as the background. See Figure 54 on page 87 to view the EXTRCT4 source code with the background choices.
Figure 16 shows the HTML sent by sample program EXTRCT4 when box 2 was selected as the background.

By the way, many browsers give you the ability to look at the source code of the displayed document. In Netscape, select View->Page Source to access the HTML of the page being displayed.

4.3.3 Sample Web-enabling of existing 3270 program

One of the last tasks we started during this project was the modification of the 3270 sample program VSAMTEST, described in Chapter 5, “Accessing existing 3270 transactions” on page 43, to a Web API program (VSAMHTML) with EXEC CICS WEB and EXEC CICS DOCUMENT.

This VSAMHTML program shows the operation principles of a possible conversion of a 3270 program to a Web-enabled program for browser output. We tested the following functions:

- Read (read a file)
- Browse (browse through a file)
- Long Browse (browse through a file with 10 records and display the last record)

We did not test the additional record functions Add, Update and Delete. These additional functions are disabled in the test program VSAMHTML. If you need this functionality, you must change the source code appropriately and test the results.

Figure 17 on page 39 shows a sample browser output generated by the VSAMHTML program. The difference between this VSAMHTML and the EXTRCT1 to EXTRCT4 programs is that in VSAMHTML, we insert variable data into the static templates; this is accomplished by using EXEC CICS DOCUMENT SET SYMBOL statements.
For example, in the statement EXEC CICS DOCUMENT SET SYMBOL(‘KEY’) VALUE(MOVEKEY) in the program VSAMHTML, the SYMBOL(‘KEY’) refers to the variable &KEY; in the template VSAMBODY, and the VALUE(MOVEKEY) is the key field within program VSAMHTML. Refer to Appendix A, “Listings” on page 69, for a complete collection of code and listing samples.

Figure 17. Sample browser output of program VSAMHTML

This sample merely gives an easily understandable example of the principles we discussed; it does not demonstrate program or Web design.

4.4 Web page design

CICS Web Support applications should take advantage of the richness of the user interface provided by Web browsers. A newly created Web page should not look like a 3270 screen. Feel free to use background colors, logos, and pictures when you design a Web page. But keep in mind that each picture or graphic that is imbedded in your Web page must be sent through the network. Extensive use of graphics and pictures may negatively impact the performance of your network, so try to find a good balance between esthetics and performance.

4.4.1 Describing and handling HTTP and HTML data

The HyperText Transfer Protocol (HTTP) defines requests and responses between a client and a server.

The HyperText Markup Language (HTML) specification defines user interface elements for text with various fonts and colors, lists, tables, images, and forms.
(text fields, buttons, check boxes, radio buttons). It is similar to other markup languages like IBM BookMaster. A good introduction to HTML can be found at the Web site:

http://www.w3.org/MarkUp/

CWS supports HTTP level 1.0 and all levels of HTML.

4.4.1.1 Creating HTML output manually
HTML documents are created as standard text files with imbedded HTML tags. They must be stored in EBCDIC format in a VSE sublibrary. Tags are used to denote various elements in an HTML document. HTML tags are recognizable as text strings enclosed in less than (<) and greater than (>) symbols. Tags are usually paired: for example, <head> and </head> to start and end the tag instruction. The end tag looks like the start tag except that a slash (/) precedes the text within the brackets.

Every HTML document should contain certain standard HTML tags. Each document consists of a head and body section. The head section contains the title, and the body section contains the actual text that is made up of paragraphs, lists and other elements.

A short HTML sample is shown here:

```html
<html>
<head>
  <title>A simple HTML example</title>
</head>
<body>
  <h1>HTML is very easy</h1>
  <p>This is the first step in HTML</p>
</body>
</html>
```

The required elements are the <html>, <head>, <title>, and <body> tags (and their corresponding end tags). For validation of your HTML document, go to the URL:

http://validator.w3.org/

4.4.1.2 Creating HTML output using more sophisticated tools
There are several PC tools that can be used to create and validate HTML documents. These tools allow you to write text on a “what you see is what you get” basis and to generate HTML code. In some cases it might be worthwhile to check the generated code and remove lines that are not needed. Some PC tools for HTML are:

- WordPro from Lotus
- Word from Microsoft
- NetObjects Fusion
- Netscape Composer
- Microsoft FrontPage Express

4.4.2 Accessing graphics and cataloging them under VSE
Graphic and multimedia files can only be created on PCs or workstations. There are several tools available to create GIF or JPEG files, for example.
Once a LOGO or PICTURE is created on your PC, it can be downloaded to your
VSE/ESA library using FTP. Graphic and multimedia files must be transferred in
binary format. For performance reasons, we recommend that you either
download the files to the TCP/IP ROOT directory, or store them on a separate
server.

Here is a sample of how to access a GIF file from the CICS home page (Figure 18
on page 41) and transfer it to your VSE/ESA:

2. On the CICS home page, right-click the CICS-IMG.gif file, and save it as
   CICSTST.gif on your PC.

   ![CICS home page](image)

3. On your PC, open a command prompt, and use the following commands to
   transfer the file CICSTST.gif to your VSE/ESA to sublibrary
   DFHHTML.DFHDOC:

   ```
   c:\>ftp n.nn.n.nnn
   Connect to n.nn.n.nnn.
   220-TCP/IP for VSE -- Version 01.03.00 -- FTP Daemon
   Copyright (c) 1995,1999 Connectivity Systems Incorporated
   220 Service ready for new user.
   User (n.nn.n.nnn:(none)): XXXX
   331 User name okay, need password.
   Password: 
   230 User logged in, proceed.
   ftp> cd dfhhtml.dfhdoc
   250 Requested file action okay, completed.
   ftp> bin
   200 Command okay.
   ftp> put cicstst.gif
   150-File: DFHHTML.DFHDOC.CICSTST.GIF
   Type: Binary Recfm: S Lrecl: 4096
   CC:ON UNIX=OFF RECLF=OFF TRCC=OFF CRLF=ON
   150 File status okay; about to open data connection
   226-Bytes sent: 5,823
   ```
4. To test the GIF file you just stored, enter `http://n.nn.nn.nnn/cicstst.gif` on your browser. Figure 19 shows the result.

![Figure 19. Display of CICSTST.GIF](http://n.nn.nn.nnn/cicstst.gif)

To include an inline image (for example, CICSTST.gif) in your HTML document, enter the full pathname (URL) in the `img` statement as follows:

```
<img src="http://n.nn.nn.nnn/CICSTST.gif">
```

Each newly created Web page should be tested with different browsers and different PCs to verify that the output (Web page) meets your requirements. The Web page layout presented by the browser may vary depending on the browser.
Chapter 5. Accessing existing 3270 transactions

The 3270 bridge facility provides the ability to run existing 3270-based transactions with a browser, without the need to change the associated application.

There are two kinds of 3270-based programs:
- BMS-based programs that use the MAP parameter in a terminal command
- Non-BMS based programs that use Terminal Control 3270 data streams (that is, CEMT)

5.1 3270 bridge logic and flow

The 3270 bridge is an integral part of CICS Web Support. It follows the same control flow, up to the alias transaction, as previously described for COMMAREA-based programs as shown in Figure 1 on page 7.

The switch to the bridge facility takes place when DFHWBTTA has been detected in the incoming HTTP data stream as the program name followed by the name of the transaction to be run; see Figure 20.

The process shown in Figure 20 is explained here:

1. If the analyzer requests a converter, the alias links to it and requests the decode function. Decode sets up the COMMAREA for DFHWBTTA.
2. The alias links to DFHWBTTA and passes the COMMAREA setup by decode, or, if no converter program was called, the COMMAREA contains the entire request.
3. DFHWBTTA finds the transaction ID for the terminal-oriented transaction from the HTTP request. DFHWBTTA then starts the 3270 user transaction immediately and assigns the CICS Web bridge exit DFHWBLT.
4. For non-BMS programs, HTML conversion to 3270 data stream and back, is done automatically. For a BMS-based program, HTML conversion is done by using an HTML template.

5. When the application program attempts to write data, the data is intercepted by CICS and given to the 3270 bridge exit, which passes it to the alias. If the caller requested a converter, the alias calls the encode function of the converter, which uses the COMMAREA to prepare the response. If no converter program was called, the alias assumes that the communication area contains the desired response.

5.2 DFHWBTTA

DFHWBTTA, the terminal translation program, is a callable CICS-supplied program that provides an interface between Web browsers and user 3270 CICS transactions using the 3270 bridge facility. DFHWBTTA, and its associated programs, performs the translation between HTML and 3270 data streams or BMS maps. DFHWBTTA supports non-conversational, conversational, and pseudo-conversational transactions.

**Note:** Before you can run a BMS transaction, you must provide HTML templates that correspond to the maps you are using in the 3270 transaction. HTML templates are generated from existing BMS mapset definitions, as explained in 5.3.1, “Creating HTML templates from BMS definitions” on page 45.

5.2.1 Input to DFHWBTTA

If you want to run a 3270 transaction, you must specify DFHWBTTA as the program to be called. DFHWBTTA picks up the transaction ID (transid) of the application, and optional data, to execute under the 3270 bridge environment from the URL.

The required URL format is:

```
http://machine.name:port/converter/alias/DFHWBTTA/transid
```

If no converter is required use “CICS” in its place. The CICS-supplied alias is “CWBA”. Also note that the transid of the application to execute under the 3270 bridge facility must follow the last slash (/), for example:

```
http://.../cics/cwba/DFHWBTTA/cemt
```

To pass optional data with the request, use plus signs (+) rather than blanks. CWS interprets a (+) sign as a blank. For example, to invoke a CEMT with INQ TAS, use the following URL:

```
http://testcics/cics/cwba/dfhwbtta/cemt+inq+tas
```

The input to the user transaction ends with the first “real” blank.

**Note:** Keep in mind that once you have received your response, for example, to the above URL, you need to use the buttons provided in the browser output to simulate the various 3270 keys, not the keys on your keyboard. Also note that it is not a good practice to use the backward or forward buttons of your browser to move through your application’s output.
5.2.2 Output from DFHWBTTA

DFHWBTTA presents an HTTP response to the encode function of the converter (if any). This response is contained in a buffer that begins with a 32-bit unsigned number that specifies the length of the buffer followed by the actual HTTP response. The HTML in the response corresponds to the output BMS map or 3270 data stream from the transaction program. This output may have been customized, as described 5.3.2, “Customizing the generated HTML output” on page 57.

5.3 Programs with BMS support

In releases earlier than CICS TS 1.1.1, BMS provided three assembler macros for defining maps:

- **DFHMSD** defines a mapset
- **DFHMDI** defines a single map as a collection of fields
- **DFHMDF** defines an individual field within the map

Before you can start writing a program that uses a map, you must assemble these macros twice.

When you define TYPE=MAP on the DFHMSD macro, you can assemble and link edit a load module called the physical mapset. The physical mapset is loaded by CICS at execution time and is used to transform the application data to a 3270 data stream or vice versa.

When you specify TYPE=DSECT on the DFHMSD macro, you assemble the symbolic mapset. This is a series of data structures in the language specified in the LANG option. A symbolic map is copied into the program and enables you to refer to the fields in the maps by name, without having to know details about the physical position on the screen.

Various modifications to existing BMS macros and new tools enable generation of HTML templates from BMS maps and customization of the BMS maps to make the output more suitable to the Web browser environment.

5.3.1 Creating HTML templates from BMS definitions

This section describes how to create HTML templates from existing BMS mapset definitions.

For BMS programs that want to use the 3270 bridge facility, their BMS maps must be reassembled specifying TYPE=TEMPLATE on the DFHMSD macro, or by specifying SYSPARM=TEMPLATE in the parameters passed to the assembler. This, in turn, generates an HTML template to be used during the BMS mapping operation. Note that the label on the DFHMSD macro is used to name the HTML templates produced for each map in the mapset being processed.

There are no changes required to the application program, nor is there a need to generate a DOCTEMPLATE RDO definition, unless you want to specify a VSE/ESA library other than the IBM default library, DFHHTML, in which to store your templates.

**Note:** Installations without access to the original source code for BMS mapsets can re-create the BMS macro statements, with some limitations, using a BMS
utility called DFHBMSUP. See CICS Transaction Server for VSE/ESA Enhancement Guide Release 1, SC34-5763, for details.

5.3.1.1 Generating HTML templates from existing maps

Of the two methods for generating an HTML template, the easiest method is to specify SYSPARM='TEMPLATE' in the options being passed to the assembler when generating the map. The alternative method is to update the source for each mapset definition and add TYPE=TEMPLATE on the DFHMSD macro.

Unfortunately, in this release of VSE/ESA, there is no option on the Intelligent User Interface (IUI) COMPILE JOB GENERATION screen to allow you to generate a template instead of a regular map definition. Therefore, you will need to modify the generated output of the map definition and specify TYPE=TEMPLATE in the options being passed to the assembler. One way to solve this problem is:

- When compiling your mapset using the "COMPILE JOB GENERATION" IUI screen, specify that you want to catalog the mapset definition. You should also specify an output member name to save the generated job.

- Next, update the saved job. Figure 21 on page 47 takes you through this process.
Figure 21. CSSASMAP mapset JCL generation skeleton

The numbers in the following list correspond to the numbers printed on the right-hand side in the figure.

1 The templates must be cataloged into a library according to what we described in 3.1.1, “VSE/ESA Library for CICS Web Support templates” on page 11, which we reinforce here again. If the DOCTEMPLATE does not define a library name, CICS searches the LIBDEF search chain for the member (still with suffix HTML) and in the order specified in the LIBDEF chain.

2 Remove this catalog statement. This allows the proper catalog statement to be generated for the template name. As mentioned earlier, the HTML template names consist of the label from the DFHMSD macro, plus one character starting from A for each map in the mapset. This allows the bridge exit to match the HTML template with the BMS map when a BMS SEND or RECEIVE is issued by a program. The library member type is always HTML.

3 Change the SYSparm='DSECT' to SYSParm='TEMPLATE'.
Another method is to modify C$$nnMAP in ICCF library2 and make the changes directly to the skeleton.

To demonstrate the 3270 bridge facility, we used an assembler program generating BMS output to allow a user to access a VSAM data set for read, browse, delete and update functions. The program name is VSAMTEST. The mapset name is VSAMSET and the transid to invoke this program is VSAM.

The source for the BMS mapset defined for our program is shown in Figure 22 on page 49. This is followed by the normal 3270 screen output when transid VSAM is entered on the 3270 CICS screen. Note that we added additional BMS macros, DFHMSX and DFHMDX, to customize the output sent to a browser. This customization is ignored with a regular 3270 data stream generation. These macros are discussed further in 5.3.2, “Customizing the generated HTML output” on page 57.

**Note**: The source for our assembler program and JCL used to define and load the VSAM file is listed in Appendix A, “Listings” on page 69.
Figure 22. Defining BMS mapset VSAMSET
5.3.1.2 Map output on a 3270 screen

Using the information supplied in 5.3.1, “Creating HTML templates from BMS definitions” on page 45, we generated our HTML template for which the source is listed on the following pages. Note that if you intend to modify the HTML source output, it is case sensitive. BMS generates buttons to represent 24 PF keys, three PA keys, and an Enter key as shown in Figure 24 on page 57.

When you generate HTML templates from BMS maps, templates can be larger than 32k. If this is the case, they cannot be used by the 3270 Web bridge. This problem is not apparent until a transaction using the map is run using the 3270 bridge. When this happens, message DFHWB0133 is issued and **500 Internal Server Error** is displayed on the browser.

The following example shows the generated HTML for the VSAMSET mapset:

```html
<!doctype html public "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>CICS Web Support BMS screen emulation</title>
<meta name="generator" content="CICS Transaction Server/1.1.1">
<script language="JavaScript">
<!--
function dfhsetcursor(n)
 {for (var i=0;i<document.VSAMMAP.elements.length;i++)
  {if (document.VSAMMAP.elements[i].name == n)
   {document.VSAMMAP.elements[i].focus();
    document.VSAMMAP.DFH_CURSOR.value=n;
    break}}}
function dfhinqcursor(n)
 {document.VSAMMAP.DFH_CURSOR.value=n}
// -->
</script>
```
Chapter 5. Accessing existing 3270 transactions

51
<table>
<thead>
<tr>
<th>Function</th>
<th>Record Key</th>
<th>Record Data</th>
<th>Response</th>
<th>Timeout</th>
<th>Milliseconds Runtime</th>
<th>Function Out</th>
<th>Term Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>B BROWSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L LONG BROWSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<input type="submit" name="DFH_PF1" value="PF01" />
<input type="submit" name="DFH_PF2" value="PF02" />
<input type="submit" name="DFH_PF3" value="PF03" />
<input type="submit" name="DFH_PF4" value="PF04" />
<input type="submit" name="DFH_PF5" value="PF05" />
<input type="submit" name="DFH_PF6" value="PF06" />
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRAN</td>
<td>CICS transaction id</td>
</tr>
<tr>
<td>FUNC</td>
<td></td>
</tr>
<tr>
<td>KEY</td>
<td></td>
</tr>
<tr>
<td>RECORD</td>
<td></td>
</tr>
</tbody>
</table>

VSAM TEST MAP

R READ

U READ FOR UPDATE + REWRITE

A ADD
### Browse Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Input Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>B BROWSE</td>
<td>&lt;input type=&quot;text&quot; name=&quot;FUNC&quot; value=&quot;&amp;FUNC;&quot; onFocus=&quot;dflhinqcursor('FUNC')&quot; size=&quot;1&quot; maxlength=&quot;1&quot;&gt;</td>
</tr>
<tr>
<td>LONG BROWSE</td>
<td>&lt;input type=&quot;text&quot; name=&quot;KEY&quot; value=&quot;&amp;KEY;&quot; onFocus=&quot;dflhinqcursor('KEY')&quot; size=&quot;6&quot; maxlength=&quot;6&quot;&gt;</td>
</tr>
<tr>
<td>RECORDKEY</td>
<td>&lt;input type=&quot;text&quot; name=&quot;RECORD&quot; value=&quot;&amp;RECORD;&quot; onFocus=&quot;dflhinqcursor('RECORD')&quot; size=&quot;80&quot; maxlength=&quot;80&quot;&gt;</td>
</tr>
<tr>
<td>RECORDDATA</td>
<td></td>
</tr>
<tr>
<td>&amp;RESPONSE</td>
<td></td>
</tr>
<tr>
<td>&amp;TIMEOUT</td>
<td></td>
</tr>
<tr>
<td>MILLISSECONDS</td>
<td></td>
</tr>
<tr>
<td>RUNTIME</td>
<td></td>
</tr>
<tr>
<td>&amp;FUNCOUT</td>
<td></td>
</tr>
<tr>
<td>&amp;TERMOUT</td>
<td></td>
</tr>
</tbody>
</table>

**Additional Buttons:**

- [DFH_PF1] (PF01)
- [DFH_PF2] (PF02)
- [DFH_PF3] (PF03)
- [DFH_PF4] (PF04)
Chapter 5. Accessing existing 3270 transactions

<table>
<thead>
<tr>
<th>Function</th>
<th>Record Key</th>
<th>Record Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF05</td>
<td>PF06</td>
<td>PF07</td>
</tr>
<tr>
<td>PF08</td>
<td>PF09</td>
<td>PF10</td>
</tr>
<tr>
<td>PF11</td>
<td>PF12</td>
<td>PF13</td>
</tr>
<tr>
<td>PF14</td>
<td>PF15</td>
<td>PF16</td>
</tr>
<tr>
<td>PF17</td>
<td>PF18</td>
<td>PF19</td>
</tr>
<tr>
<td>PF20</td>
<td>PF21</td>
<td>PF22</td>
</tr>
<tr>
<td>PF23</td>
<td>PF24</td>
<td>PF25</td>
</tr>
</tbody>
</table>

<!-- The following variables are the names of the fields that could contain the next CICS transaction id -->
<input type="hidden" name="DFH_NEXTTRANSID.1" value="TRAN" />
<input type="hidden" name="DFH_NEXTTRANSID.2" value="FUNC" />
<input type="hidden" name="DFH_NEXTTRANSID.3" value="KEY" />
<input type="hidden" name="DFH_NEXTTRANSID.4" value="RECORD" />
</form>
</body></html>
<table>
<thead>
<tr>
<th>RESPONSE</th>
<th>TIMEOUT</th>
<th>MILLISECONDS RUNTIME</th>
<th>FUNCOUT</th>
<th>TERMOUT</th>
</tr>
</thead>
</table>

**CICS Transaction Server for VSE/ESA**

---

**Command Line Interface:**

The following variables are the names of the fields that could contain the next CICS transaction id:

- **TRAN**
- **FNC**
- **KEY**

---

**Additional Command Line Options:**

- **PF1**
- **PF2**
- **PF3**
- **PF4**
- **PF5**
- **PF6**
- **PF7**
- **PF8**
- **PF9**
- **PF10**
- **PF11**
- **PF12**
- **PF13**
- **PF14**
- **PF15**
- **PF16**
- **PF17**
- **PF18**
- **PF19**
- **PF20**
- **PF21**
- **PF22**
- **PF23**
- **PF24**
- **PF25**
- **PF26**
- **PF27**
- **PF28**
- **PF29**
- **PF30**
- **PF31**
- **PA1**
- **PA2**
- **PA3**
- **CLEAR**
- **Enter**
5.3.2 Customizing the generated HTML output

To improve the layout of 3270-based output for a browser, modifications can be made easily using two new BMS customization macros:

- DFHMDX
- DFHWBOUT

Figure 24. 3270 bridge generated HTML page displayed by a browser
These macros are inserted into the BMS map source, but have an effect only on
the generated HTML. The definition of a customizing macro must be written
according to the rules for assembler macro definitions and must also follow the
rules for assembler language macro statements. A customizing macro definition
contains the following elements as described in Figure 25:

- A MACRO statement to begin the definition.
- The name of the macro.
- Any number of invocations of the DFHMDX macro. DFHMDX is invoked from
  within macro DFHMSX.
- A MEND statement to end the definition.

For more information on these customizing macros, refer to CICS Transaction

<table>
<thead>
<tr>
<th>MACRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFHMSX</td>
</tr>
</tbody>
</table>
| DFHMDX MAPSET=*,MAP=*, *
  BGCOLOR=yellow, *
  TEXT=blue, *
  RESET=NO
| MEND   |

Figure 25. Customizing a macro definition from VSAMMAP

5.3.2.1 DFHMSX and DFHMDX
You can use the DFHMSX macro to define your own customization macro that is
used when the templates are being created from the BMS map definitions. With
the DFHMDX macro, HTML templates can be customized to:

- Support the application’s use of keys that are not in the standard output
- Suppress the HTML reset function, which does not correspond to any 3270
  function
- Change the appearance of the keys, or the text associated with them
- Provide an HTML title for the HTML page
- Provide a masthead graphic for the HTML page
- Change the color of the background or specify a special background
- Modify the BMS colors
- Suppress parts of the BMS maps
- Add Web browser control functions, JavaScript functions for example, to the
  HTML page

When CICS creates the templates for each of your BMS map definitions, it
invokes the DFHMSX customizing macro. Each DFHMDX macro is processed in
sequence, and if applicable, the parameter values are stored. Where a duplicate
parameter is specified for a particular map or mapset, the new value replaces the
previous value for that map or mapset. The first invocation of DFHMDX sets
defaults for the values to be applied to subsequent invocations of DFHMDX by
specifying an asterisk (*) for the mapset name and map name.
In our example, we have set as defaults, the background color to yellow, the lettering as blue, and suppressed the HTML reset function. The definitions, which are shown in Figure 25 on page 58 and in Figure 22 on page 49, are at the top of the source code for our BMS mapset.

5.3.2.2 DFHWBOUT

The DFHWBOUT macro is used to add text to the HTML page generated from a BMS map only if PARM=TEMPLATE. You can:

- Add Web browser control functions; for example, JavaScript functions, to the HTML page
- Add text that appears only on the HTML page, but is not part of the BMS map
- Add HTML header information to the HTML page

If the macro is used before the first occurrence of DFHMDF in a macro, the text is placed in the <head> section of the HTML page. If the macro is used elsewhere in the map, the text is placed inline in the HTML page, immediately following the text generated by the preceding DFHMDF macro. Figure 26 shows how the DFHWBOUT macro is defined.

![DFHWBOUT customization macro](image)

For more details about the new BMS macros, see *CICS Transaction Server for VSE/ESA Internet Guide Release 1, SC34-5765*.

5.4 Customizing the 3270 to HTML conversion for non-BMS applications

When a 3270 transaction is run under a bridge transaction and the application sends 3270 data streams, the HTML conversion is done dynamically. This is a standard conversion, which can be customized in two ways.

The first way is to use a converter program where the encode function customizes the output after the 3270 conversion.

The second way is to use header and footer templates. These templates are described by an RDO DOCTEMPLATE definition and they are copied in front of, and appended after, the converted screen, as shown in Figure 27 on page 60.
Header and footer templates can be allocated as follows:

- Header and footer template per transaction
- Header and footer template per CICS region

You supply one or more of the following templates, whose names are defined in the \texttt{TEMPLATENAME} fields of \texttt{DOCTEMPLATE} definitions:

- \texttt{tranHEAD} is a template that is inserted at the head of the HTML page being output for transaction \texttt{tran}, if it is installed.
- \texttt{CICSHEAD} is a template that is inserted at the head of the HTML page being output for transactions that do not have a corresponding \texttt{tranHEAD} template installed.
- \texttt{tranFOOT} is a template that is inserted at the foot of the HTML page being output for transaction \texttt{tran}, if it is installed.
- \texttt{CICSFOOT} is a template that is inserted at the foot of the HTML page being output for transactions that do not have a corresponding \texttt{tranFOOT} template installed.

Figure 28 on page 61 shows an example of a non-customized CEMT panel on a Web browser. The default header generated by CICS for a non-BMS template is as follows:

```html
<!doctype html public "//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>CICS Web support screen emulation</title>
<script language="JavaScript">
</script>
<meta name="generator" content="CICS Transaction Server/1.1.1">
</head>
<body>
The default footer generated by CICS for a non-BMS template is as follows:

```html
</pre>
</body>
```
5.5 Bridge exits

An important part of the new 3270 bridge facility is the bridge exit program. The bridge exit program is a user-replacable module that essentially emulates a 3270 terminal transparently to the 3270 transaction being executed. It intercepts all the 3270 input and output data commands issued by the application program and does the necessary data transformation and routing.

Two sample 3270 bridge exit programs that use the CICS Web Support are supplied with CICS:

- DFH0CBRE is a bridge exit program supplied in COBOL source that uses CICS temporary storage (TS) or transient data (TD) queues to pass input and output from and to the user application (another CICS application). This exit can be modified. DFH0CBRE is the most general of the supplied exits. To run a transaction using this exit, you simply issue the following command:

```
START TRANSID() BRDATA() BREXIT(DFH0CBRE)
```
The bridge environment is established when CICS receives a START BREXIT TRANSID call that applies only to non-terminal starts. The CICS translator rejects START BREXIT requests when TERMID is specified. The transaction specified by the TRANSID is associated with the bridge exit specified by BREXIT.

If BREXIT is not specified in START BREXIT, then the user transaction RDO definition's Brexit value is used. If no bridge exit name is specified, then the default 3270 bridge exit, DFHWBLT, is invoked.

Figure 29 shows the Brexit parameter in our RDO definition for our transaction, “VSAM”. As you can see, we did not specify a parameter, which defaults to the bridge exit named DFHWBLT.

Also supplied is DFH0CBRF, which is a COBOL bridge exit formatter designed to work with DFH0CBRE.

- DFHWBLT is the default bridge exit provided by the Web 3270 support. It allows you to access a CICS transaction from the Internet through a Web browser. It is provided in object code only. This exit handles conversion of 3270 data streams to and from HTML and supports both BMS and 3270 terminal control. Support is also provided for customization of the standard HTML output produced.

If nothing is entered for BREXIT, the default bridge exit name DFHWBLT is used.

You must ensure that the LOCAL CBRF terminal is defined for default bridge facility, otherwise you will get an ABRJ abend or even a DFHTF0002 system dump error code 1715. For example:

```
DFHTF0002 BRDXC CICS A severe error (code X'1715') has occurred in module DFHTFRF
This terminal is defined in RDO groups DFHTERM and VSETERM. Please do not delete.
```

Additional information on the supplied bridge exits, and how to write your own, is documented in the following publications:

- **CICS Transaction Server for VSE/ESA Internet Guide Release 1**, SC34-5765
Chapter 6. Security and performance

This chapter covers the topics of security and performance. Although it does not go into great detail on either topic, consider the following questions while planning your system requirements for an e-business solution:

- How much security do you need?
- Do you need a 3-tier solution?
- Do you need a firewall?
- Do you need a Proxy server?
- How much tuning is enough?

The answers to these questions depend on the type of data being accessed, and on individual company standards. We feel that in this environment, more security is better than less. With this in mind, we chose to use this section to point you in the right direction. The ultimate selections are yours. This is an overview of the non-SSL facilities available.

6.1 CICS Web Support security

CICS TS provides three sample security programs and two sample authentication programs that you may use for security and authentication. All five programs are written in assembler language.

Sample security programs are:
- DFH$WBSA: Security analyzer
- DFH$WBSC: Security converter
- DFH$WBSN: Sign-on program

To use the security analyzer program, you must specify its name as the Analyzer Program name in the CICS TS RDO definition TCPIPSERVICE. Multiple TCPIPSERVICE definitions may be used to allow access through multiple ports. Each port may have different security criteria established allowing and denying service.

In this case, DFH$WBSA is the analyzer name in the RDO definition TCPIPSERVICE. When the user enters the URL, DFH$WBSA analyzes the request. DFH$WBSA sets the converter to DFH$WBSC. This converter saves the original URL using the state management program, DFH$WBST, and sets the user program to DFH$WBSN. The sign-on program builds an HTML form to be sent to the browser requesting a user ID and password. The encode function of DFH$WBSC creates the HTTP response. The user receives the HTML form and fills it in.

The security analyzer detects the input and calls the security converter, DFH$WBSC, again. DFH$WBSC ensures that DFH$WBSN is called. DFH$WBSN extracts the user ID and password and issues a EXEC CICS VERIFY PASSWORD command to validate the user ID. The encode function of DFH$WBSC builds the HTTP response and adds a redirection to it, specifying the original URL. The browser receives the redirected URL and sends a request for the original program. Through all of this, the state management program has maintained a token which must be carried as a query string through the remainder of the conversation for it to execute under the desired user ID.
This method of prompting for user ID and password has the disadvantage that the user ID and password are transmitted in readable text over the network. This should only be used in an intranet, and only with care.

For additional information about how to manage this type of access, refer to CICS Transaction Server for VSE/ESA Internet Guide Release 1, SC34-5765.

Sample authentication programs are:

- DFH$WBAU: Basic authentication analyzer
- DFH$WBSB: Basic authentication converter

These sample programs show how to use the HTTP basic authentication. The first time the browser attempts to access CICS TS, it prompts for a user ID and password, which will be encoded, but not encrypted, for transmission. This user ID and password will be supplied with every request and must be validated by CICS TS for every HTTP request. The second and subsequent requests are not sent as prompts to the user at the browser.

The basic authentication analyzer, DFH$WBSB, looks for an incoming Authorization HTTP header and extracts the user ID. DFH$WBSB always schedules the basic authentication converter, DFH$WBAU.

DFH$WBAU decodes the user ID and password and issues an EXEC CICS VERIFY PASSWORD command. If the user ID/password combination is not valid, or there is no HTTP Authorization header, then an HTTP 401 response is sent back to the browser, and the result is that the user is prompted for a password. If the user ID/password combination is valid, the alias transaction that runs the application program will run under this user ID.

To use this method, DFH$WBSB must be named as the analyzer in your CICS TS TCPIPSERVICE definition.

In this case, the user ID and password are not encrypted, but are encoded using a commonly known translation algorithm. The decode algorithm can be found in the sources of both sample authentication programs.

In addition to the CWS programs, you may also employ a CICS TS security package or VSE/ESA security package to assist in securing your data.

### 6.1.1 VSE/ESA Web Server security

TCP/IP for VSE/ESA provides VSE/ESA the ability to participate in the Internet and intranet environments. This is the interface through which your communications and data will flow. Therefore, security is a major point to consider. A firewall can be implemented to stop unauthorized access from the Internet or from your intranet. The VSE/ESA Web server itself cannot act as a firewall. If you need the type of security that a firewall provides, it must be implemented outside of the VSE/ESA system.

TCP/IP for VSE/ESA offers a sample security exit that can be found in PRD1.BASE called SECEXIT.A. This program offers the possibility to establish your own security exit to allow or deny access to your VSE/ESA system based on the hardware or IP address. You may also modify the code to check for valid user IDs and passwords. FTP access to POWER queues can be restricted by
customizing the sample member, SECPOWER.A, which is also provided in PRD1.BASE.

TCP/IP for VSE/ESA also provides a function called basic security. This is done by specifying SET SECURITY=ON in the IPINITXX.L startup member and the DEFINE USER, ID=XXXX, PASSWORD=XXXX. This forces the clients to log on with a valid user ID and password before they can use the TCP/IP functions. You may also want to use the SET SECURITY_ARP and SET SECURITY_IP commands to provide further security.

When using the HTTP server, the parameter SECURE=YES is available for your use. Review the TCP/IP manuals for the appropriate use of this security feature. Another parameter, CONFINE=YES, may be used to limit sublibrary access to the sublibrary specified by the ROOT parameter. In addition, the HTTP server executes in read-only mode.

TCP/IP for VSE/ESA provides a wealth of functionality. It also provides the ability to force sign-ons, limit access, limit searches, restrict FTP, and more.

### 6.1.2 Security summary

CICS TS and TCP/IP for VSE/ESA do provide some security. In a limited intranet, this may satisfy your company's security policy. These products, together with an external security manager for VSE/ESA and CICS TS, or perhaps a firewall or proxy server, may be needed. You must determine the levels of security required to protect your company's assets.

### 6.2 CICS Web Support performance

Not too long ago, performance considerations consisted of CPU processing power, central storage use, and minimizing I/O or having a robust enough I/O subsystem. These concerns still exist.

However, you now have more to consider. Discussing performance issues are not a primary goal of this redbook. It is important, however, to understand the possible performance issues related to CWS.

In general, the following performance tuning issues exist:

- **Operating system tuning**
  Considerations in this area include issues such as having sufficient storage, processing power, and efficient I/O subsystem. You may also consider items such as utilizing VDISK for your DFHDOC sublibrary.

- **CICS TS system tuning**
  CICS TS is a more robust subsystem than its predecessors. As a result, more storage is needed for these functions. For CWS, 500 KB of storage for DSA (24-bit) and 2 MB of storage for EDSA (31-bit) are required. In addition, 1 MB of storage for EDSA (31-bit) is required for each concurrent user.

  Keep in mind that CWS can heavily use temporary storage. Monitor this closely prior to going into production. If you can use the shared virtual area (SVA) to load CICS TS phases, do it.
• CICS TS application tuning
Issues here would be topics such as efficient coding, correct LE parameters (confirming to site standards), proper dataset blocking and other issues that most programmers should be aware of, as documented in application programming manuals.

The expected overhead of CWS functions in addition to resources consumed by applications is, in our opinion, small. We were not able to make any measurements, but for some cases we used the CICS TS auxiliary trace, which showed us only milliseconds of CPU consumptions as possible overhead.

As stated previously, use care when designing Web pages. Graphics and pictures can severely impact the response time of the network and of end users.

• TCP/IP system tuning
TCP/IP performance can be affected by partition size and partition priority within your VSE/ESA system. The TCP/IP partition should have a higher priority than CICS TS. In addition, window size, transfer buffers, and MTU/MSS used can have performance implications. More specific TCP/IP for VSE/ESA performance information and performance results are available on the CD-ROM TCP/IP for VSE/ESA, SK2T-1336, and at the VSE/ESA Internet home page found at:

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

• TCP/IP application tuning
TCP/IP application considerations should include planning the number of bytes of data being transferred in and out. Obviously, the fewer bytes of data being passed have a greater possibility of performing well.

• Network considerations
Network considerations include items such as the type of communications adapter being used. What is the LAN/WAN available throughput? Are there bottlenecks at the routers or servers?

Summary
In summary, there is a multitude of performance issues to be concerned with. However, proper planning in advance of implementation will minimize these issues. Once you have implemented CWS, monitoring ensures that you can take appropriate steps in advance of a growing workload.

For more information about performance hints and tips, visit the VSE/ESA home page at the following address. There you can find many performance-related articles and sources of information:

http://www.ibm.com/vse
Chapter 7. Problem determination and application debugging

This chapter helps you debug problems in CICS Web Support. For more information, refer to:

- CICS Transaction Server for VSE/ESA Internet Guide Release 1, SC34-5765
- CICS Problem Determination Guide, GC33-1663

7.1 Documentation about the problem

To investigate a problem, you must review logs provided by VSE/ESA, TCP/IP and CICS. In addition, you should use dumps and traces when identifying the problem. More detailed “Web support information” is written to the trace table if you set the trace level for Socket (SO) and Web (WB) domain to ALL at the “CETR Component Trace Options”. With SO=ALL you can get up to 4000 bytes of data, SO=1 does not show data.

Messages may be written to:

- Console log
- TCP/IP job log
- CICS log (use Interactive User Interface (IUI) fast path 4.2 to display the logs)
- OLPD log (use IUI fast path 4.1 to display the logs)
- Browser screen (if DFHWBEP is used, see A.7, “Source listing DFHWBEP sample program” on page 99)

Dumps and traces may be written to:

- Transaction dump data sets A or B, if defined
- CICS auxiliary trace data set, if enabled
- VSE/ESA system dump library: This also contains the CICS internal trace, if enabled

7.1.1 Using messages and codes

For messages found on any of the previously mentioned logs, use the VSE/ESA, TCP/IP, or CICS Messages and Codes manuals to find an explanation of the message. Another way to find an explanation of the message is by using the EXPLAIN MESSAGE function in VSE/ESA. On the console, type the message (for example, DFHWB1008), and press PF9 for EXPLAIN. An explanation of the message is displayed.

7.1.2 Using dumps and traces

DFHDU410 is used to print transaction dumps, and to format the trace table. Different options can be used which are described in CICS Transaction Server for VSE/ESA Operations and Utilities Guide, Release 1, SC33-1654.

To print the auxiliary trace data set, DFHTU410 is used. The trace can be printed in abbreviated and full format. A detailed description of the parameters can also be found in CICS Transaction Server for VSE/ESA Operations and Utilities Guide, Release 1, SC33-1654.
A CICS TS dump is analyzed and formatted with DFHPD410. The job with all INFOANA control statements can be generated in IUI via the dialog STORAGE DUMP MANAGEMENT (fast path 4.3). To select the level of dump formatting printed for CICS Web Support, you can change the control statement SO and WB. For a full description of the control statement, see CICS Transaction Server for VSE/ESA Internet Guide Release 1, SC34-5765.

### 7.2 Debugging with CEDX

Because CWS tasks run as nonterminal transactions within CICS TS, you should use CEDX tranid for debugging CWS tasks. The transaction CEDX is provided to assist in monitoring and debugging. CEDX is defined in the RDO group DFHEDF.

The command syntax for CEDX is as follows:

CEDX tranid,ON

or

CEDX tranid,OFF

**Note:** For COMMAREA programs, use CWBA for the tranid. For 3270-based programs, use xxxx, where xxxx is your tranid.

CICS TS intercepts the transaction specified on the CEDX tranid command, and displays the EDF diagnostic panels at the terminal on which the EDF command is issued.

CEDX provides the same function and diagnostic display panels as CEDF, and the basic rules for CEDF also apply to CEDX.
Appendix A. Listings

This appendix contains all the code and listing samples we used during this redbook project in Poughkeepsie.

A.1 Listing of DFHSIT for CWS

Following is a listing of DFHSIT with the CWS parameters highlighted.

```
<<......1......2......3......4......5......6......7......8......9......0......
MEM=DFHSITC2>>.

TITLE 'DFHSITC2 -- SIT FOR CICS TS - APPLID PRODCICS'
PUNCH 'CATALOG DFHSITC2.OBJ REP=YES'

DFHSIT TYPE=CSECT, *

AIEXIT=IESZATDX, AUTO INSTALL TERMINALS *
AIDLDELAY=200, AUTO INSTALL DEL TERM PQ03810 *
AIHMAX=100, AUTO INSTALL CONC TERMINALS *
AIRDDELAY=700, AUTO INSTALL ELAPSE TIME *
AIKPFRQ=200, ACTIVITY KEYPOINTING FREQUENCY *
APPFD=PRODCICS, <= CICS APPLICATION NAME *
AUXTR=OFF, AUXTRACE OFF *
BMS-PULL, FULL BASIC MAPPING SUPPORT *
CLEXEDT=MOTIFY, *
CMDPROT=YES, VALIDATE START ADDRESSES *
CMSEC-ASIS, CMSEC WILL BE HONORED *
CONFDATA-SHOW, SHOW USER DATA IN TRACE *
CONFXT=NO, VTAM SHOW USER DATA *
CSDACC=READWRITE, CSD MAY BE UPDATED *
CSDLSRNO=1, CSD LOCAL SHARED RESOURCE *
CSDSTRNO=4, CSD SIMULTANEOUS ACCESS *
DATFORM=MMDDYY, EXTERNAL DATE DISPLAY *
DBP=1$, DYN. BACKOUT (NO LOCAL DLI I/F)*
DBUFSZ=2000, DYN. ADJUSTED BY CICS *
DCT=C2, FOR SECOND CICS *
DFORMAT=MMDDYY, EXTERNAL DATE DISPLAY *
DBUFSZ=2000, DYN. ADJUSTED BY CICS *

GMTEXT='VSE/ESA CICS/TS', GMM MSG TEXT *
GMTRAN=ICKM, LOGON TRANSACTION ID *
GNTRAN=IEGT, TIME OUT TRANSACTION *
GRPLIST=(VSELST2,MYLST2), AUTOINST, TERMS, & MRO *
ICP=COLD, INTERVAL CONTROL PGM *
ICV=1000, INTERVAL CONTROL EXIT TIME-MS *
ICVR=20000, RUNAWAY TASK TIME *
ICVSD=200, TERMINAL SCAN DELAY *
INTR=DT, INTERNAL TRACE *
IRCSTRT=NO, START IRC DURING INITIALIZATION *
ISC=YES, INTERSYSTEM COMMUNICATION *
JCT=NO, NO JOURNALING *
JCVS=NO, SUPPORT LE ON THIS CICS *
KRNLOG=YES, VTAM LOGON DATA *
MCT=NO, NO MONITOR CONTROL TABLE *
MN=OFF, MONITORING OFF *
MNCONV=NO, NO MONITORING OF CONVERSATIONAL *
MNEXC=OFF, MONITORING EXCEPTION CLASS *
MNIPF=0, MONITORING FREQUENCY *
MNPER=OFF, MONITORING PERFORMANCE CLASS *
MNSYN=NO, MONITORING SYNCPNT *
MNTIME=LOCAL, MONITORING TIME GMT *
MROBUTCH=1, MRO BATCHING EVENTS *
MROLONG=NO, MRO LONG RUNNING MIRROR TASK *
MSOVR=1, MESSAGES ON BOTH SYSLST/SYSLOG *
MXT=50, MAX NO. OF ALL CONCURRENT TASKS*

Figure 30. DFHSITC2 (Part 1 of 2)
A.2 Listing of RDO definitions: CWS parameters highlighted

The following figures show RDO definitions for DOCTEMPLATEs and for TCPIPSERVICE.

A.2.1 Listing defined DOCTEMPLATEs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Group</th>
<th>Date</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTRCTF1</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.32.50</td>
</tr>
<tr>
<td>EXTRCTF2</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.33.16</td>
</tr>
<tr>
<td>EXTRCTF3</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.33.28</td>
</tr>
<tr>
<td>EXTRCTF4</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.33.41</td>
</tr>
<tr>
<td>EXTRCTH1</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.33.56</td>
</tr>
<tr>
<td>EXTRCTH2</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.14.11</td>
</tr>
<tr>
<td>EXTRCTH3</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.14.20</td>
</tr>
<tr>
<td>EXTRCTH4</td>
<td>DOCTEMPLATE</td>
<td>EXTRACT</td>
<td>00.248</td>
<td>18.14.29</td>
</tr>
</tbody>
</table>

Figure 32. CEDA V DOCTEMPLATE(EXTRACT*) GROUP(*)

Figure 33. CEDA V definitions of DOCTEMPLATE EXTRACTF1
### A.2.2 Listing of RDO TCPIPSERVICE

<table>
<thead>
<tr>
<th>OBJECT CHARACTERISTICS</th>
<th>CICS RELEASE = 0411</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEDA View TCpipservice{ HTTPNSSL }</td>
<td></td>
</tr>
<tr>
<td>TCpipservice : HTTPNSSL</td>
<td></td>
</tr>
<tr>
<td>Group : F8CWS</td>
<td></td>
</tr>
<tr>
<td>Description : CICS Web TCPIPSERVICE</td>
<td></td>
</tr>
<tr>
<td>Urm : DFRRNBDX</td>
<td></td>
</tr>
<tr>
<td><strong>Portnumber</strong> : 01080</td>
<td>1-65535</td>
</tr>
<tr>
<td><strong>Status</strong> : Open</td>
<td>Open</td>
</tr>
<tr>
<td><strong>Transaction</strong> : CWXN</td>
<td></td>
</tr>
<tr>
<td><strong>Backlog</strong> : 00005</td>
<td>0-32767</td>
</tr>
<tr>
<td><strong>Ipaddress</strong> :</td>
<td></td>
</tr>
<tr>
<td><strong>Socketclose</strong> : No</td>
<td>No</td>
</tr>
</tbody>
</table>

SYSID=CIC1 APPLID=DBDCCICS

Figure 34. RDO TCPIPSERVICE definition
// EXEC LIBR,PARM="MSHP;A S=PRD2..CONFIG"
CATALOG IPINIT00.L REPLACE=YES

* Define the constants
*------------------------------------------*
SET IPADDR = N.NN.NN.NN <---------
SET MASK = 255.255.255.000
* SET ALL_BOUND = 30000
SET WINDOW = 4096
SET TRANSFER_BUFFERS = 20
SET TELNETD_BUFFERS = 20
SET RETRANSMIT = 100
SET DISPATCH_TIME = 30
SET REDISPATCH = 10
* SET SECURITY = ON
SET ADDITIONAL_WINDOW = 70000
SET PING_MESSAGE = ON
SET MAX_SEGMENT = 32684
* GATEWAY ON
------------------------------------------*
** Wait for VTAM Startup **
------------------------------------------*
WAIT VTAM
------------------------------------------*
** Define Routine Information **
------------------------------------------*
* DEFINE ROUTE,ID=VMESA,LINKID=VM_TCPIP,IPADDR=0.0.0.0, - <---------
GATEWAY=N.NN.NN.NN
* DEFINE ROUTE,ID=VMASA,LINKID=VM_TCPIP,IPADDR=0.0.0.0, - <---------
GATEWAY=N.NN.NN.NN
* Define Names
*------------------------------------------*

Figure 35. IPINIT00 (Part 1 of 3)
DEFINE NAME, NAME=MYVMVSE, IPADDR=N.NN.NN.NN
DEFINE NAME, NAME=GREG, IPADDR=N.NN.NN.NN
DEFINE NAME, NAME=HANS, IPADDR=N.NN.NN.NN
DEFINE NAME, NAME=BENN, IPADDR=N.NN.NN.NN
DEFINE NAME, NAME=KENN, IPADDR=N.NN.NN.NN
*------------------------------------------*
** Define Name Server Support **
*------------------------------------------*
SET DNS1=N.NN.NN.NN
*------------------------------------------*
** Define HTTPD **
*------------------------------------------*
DEFINE HTTPD, ID=HTTP1, ROOT='DFHHTML.DFHDOC', SECURE=NO, CONFINE=NO
*------------------------------------------*
** Define Telnet Daemons **
*------------------------------------------*
DEFINE TELNETD, ID=LU, TERMNAME=TELNLJ, TARGET=DBDCCICS, PORT=23, COUNT=2
*------------------------------------------*
** Define FTP Daemons **
*------------------------------------------*
DEFINE FTPD, ID=FTP, PORT=21, COUNT=2
*------------------------------------------*
** Automated Line Printer Client **
*------------------------------------------*
DEFINE EVENT, ID=LST_LISTEN, TYPE=POWER, CLASS=X, QUEUE=LST, ACTION=LPR
*------------------------------------------*
** Setup the File System **
*------------------------------------------*
DEFINE FILESYS, LOCATION=SYSTEM, TYPE=PERM
  * DEFINE FILE, PUBLIC='IJSYSRS', DLBL=IJSYSRS, TYPE=LIBRARY
  * DEFINE FILE, PUBLIC='PRD1', DLBL=PRD1, TYPE=LIBRARY
  * DEFINE FILE, PUBLIC='PRD2', DLBL=PRD2, TYPE=LIBRARY
  * DEFINE FILE, PUBLIC='DFHHTML', DLBL=DFHHTML, TYPE=LIBRARY
  * DEFINE FILE, PUBLIC='POWER', DLBL=IJQFILE, TYPE=POWER

Figure 36. IPINIT00 (Part 2 of 3)
A.4 Listing of program samples including JCL

This section contains all HTML documents and CICS assembler sample programs for the CWS COMMAREA approach. In addition, the catalog and compile jobs are listed.

A.4.1 All HTML documents

*n.nn.nn.nnn* in HTML documents refers to the TCP/IP address.

```
CATALOG INDEX.HTML REPLACE=YES
<html>
<head>
<title>ESA250 - Running TCP/IP for VSE/ESA 1.4</title>
</head>
<body>
<h2>ESA250 - Running TCP/IP for VSE/ESA 1.4</h2>
<p><hr></p>
<h2>CICS Web Support</h2>
<p>Pressing <a href="DEMO.HTML">here</a> will take you to a page demonstrating CICS Web Support on CICS Transaction Server for VSE/ESA 1.1.1.</p>
<hr>
<h3>Just try it out</h3>
<p><hr></p>
This Home page has been updated on 11th September 2000.
</body>
</html>
```

*Figure 38. INDEX.HTML document*
Figure 39. DEMO.HTML document

<catalog demo.html replace=yes>
<html>
<head>
<title>ESA250 - CICS Web Support on VSE/ESA</title>
</head>
<body>
<h1>ESA250 - Running CICS Web Support on VSE/ESA</h1>
<p><hr></p>
<p>Press <a href="http://n.nn.nn.nnn:1080/cics/cwba/DFH$WB1A">DFH$WB1A</a> to see the CICS Web Support sample application in action. What is actually passed to the Web Browser and thus to CICS Transaction Server for VSE/ESA is: "http://n.nn.nn.nnn:1080/cics/cwba/DFH$WB1A"
</p>
<hr>
<p>Press <a href="http://n.nn.nn.nnn:1080/cics/cwba/EXTRCT1">EXTRCT1</a> to see the CICS Web Support running an application that will call the EXEC CICS EXTRACT command to determine exactly what CICS environment the program is executing in. What is actually passed to the Web Browser and thus to CICS Transaction Server for VSE/ESA is: "http://n.nn.nn.nnn:1080/cics/cwba/EXTRCT1"
</p>
<hr>
</body>
</html>
</catalog>

Figure 40. Header document EXTRCTH1.HTML for sample EXTRCT1

<catalog extrcth1.html replace=yes>
<html>
<head>
<title>EXTRACT - CICS Web Support on VSE/ESA</title>
</head>
<body>
<h1>EXTRACT - Running CICS Web Support on VSE/ESA</h1>
<p>With no image help at all</p>
</body>
</html>
</catalog>
Figure 41. Footer document EXTRCTF1.HTML for sample EXTRCT1

Figure 42. Header document EXTRCTH2.HTML for sample EXTRCT2

Figure 43. Footer document EXTRCTF2.HTML for sample EXTRCT2
Figure 44. Header document EXTRCTH3.HTML for sample EXTRCT3

Figure 45. Footer document EXTRCTF3.HTML for sample EXTRCT3

Figure 46. Header document EXTRCTH4.HTML for sample EXTRCT4
A.4.2 All HTML documents for VSAMHTML sample program

The sample code below is the VSAMHTML program.

```
* VSAMHTML -- PROGRAM FOR ACCESSING FILEA WITH CMS  *
* Please NOTE: The functionalty of this program is equivalent  *
* to the VSAMTEST 3270 sample program.  *
* It should just demonstrate how a 3270 program could be  *
* converted to a web enabled program with WEB and DOCUMENT  *
* API.  *
* Due to shortage of time READ, BROWSE and L-BROWSE have been  *
* tested, not ADD, DELETE and UPDATE. Therefore these three  *
* functions are disabled. A plain document is sent instead.  *
* Used DOCTEMPLATES:  *
* VSAMHEAD, VSAMBODY and VSAMFOOT  *
* have to be defined via RDO.  *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
PRINT GEN
***********************************************************************
DFHREISITG DSECT
WORK DS D
VSAMAREA DS CL80
SCRATCH DS CL80 SCRATCH INPUT AREA USED FOR UPDATE
DOCTOKEN DS CL16
FUNCTION DS CL8
FUNCTIONL DS F
VSAMKEY DS CL16
VSAMKEYL DS F
ADDRREC DS CL80
ADDRRECL DS F
MOVEKEY DS CL6
MOVEDATA DS CL73
R8 EQU 8
* COPY VSAMSET
***********************************************************************
PROGRAM START
***********************************************************************
PRINT GEN
VSAMHTML CSRCT
VSAMHTML AMODE 31 55200000
VSAMHTML RMODE ANY 56000000
B MVC
DC CL8 'VSAMHTML'
MVC DS 0H
MVC FUNCTIONL,=F'6'
MVC ADDRECL,=F'73'
EXEC CICS HANDLE CONDITION NOTFOUND(SENDHTML)
EXEC CICS HANDLE CONDITION ENDFILE(SENDHTML)
* Address COMMAREA for viewing of input data stream
```
EXEC CICS ADDRESS COMAREA(R8)

* analyse function
  EXEC CICS WEB READ FORMFIELD(TEXTURE) NAMELENGTH(L'TEXTURE) *
  VALUE(FUNCTION) VALUELENGTH(FUNCTIOL) NOHANDLE

* analyse key
  EXEC CICS WEB READ FORMFIELD(KEY) NAMELENGTH(L'KEY) *
  VALUE(VSAMKEY) VALUELENGTH(VSAMKEY) NOHANDLE
  CLI FUNCTION,C'R' IF FUNCTION IS READ
  BE READ
  CLI FUNCTION,C'U' IF FUNCTION IS UPDATE
  BE SENDHTTP(UPDATE DISABLED)
  CLI FUNCTION,C'A' IF FUNCTION IS ADD
  BE SENDHTTP(ADD DISABLED)
  CLI FUNCTION,C'D' IF FUNCTION IS DELETE
  BE SENDHTTP(DELETE DISABLED)
  CLI FUNCTION,C'B' IF FUNCTION IS BROWSE
  BE BROWSE
  CLI FUNCTION,C'L' IF FUNCTION IS LONG BROWSE (10 RECORDS)
  BE BROWSE10

* Create document with header template
  SENDHTTP EXEC CICS DOCUMENT CREATE DOCTOKEN(DOCTOKEN) *
  TEMPLATE('VSAMHEAD')

* Insert key
  EXEC CICS DOCUMENT SET SYMBOL('KEY') VALUE(MOVEKEY) *
  DOCTOKEN(DOCTOKEN) LENGTH(VSAMKEYL)

* Insert data
  EXEC CICS DOCUMENT SET SYMBOL('RECORD') VALUE(MOVEDATA) *
  DOCTOKEN(DOCTOKEN) LENGTH(ADDRECL)

* Insert body template
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
  TEMPLATE('VSAMBODY')

* Insert footer template
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
  TEMPLATE('VSAMFOOT')

* Send complete document
  SENDHTTP EXEC CICS WEB SEND DOCTOKEN(DOCTOKEN) *
  CLNTCODEPAGE('iso-8859-1')
  B RETURN

READ BQU *
  EXEC CICS READ DATASET('FILEA') RIDFLD(VSAMKEY) *
  INTO(VSAMAREA)
  MVC MOVEKEY(6),VSAMKEY
  MVC MOVEDATA(73),VSAMAREA+7
  B SENDHTTP

UPDATE DS OH DO READ/UPDATE OF RECORD
  EXEC CICS READ DATASET('FILEA') RIDFLD(VSAMKEY) *
  INTO(VSAMAREA) UPDATE
  MVC MOVEKEY(6),VSAMKEY
  MVC MOVEDATA(73),VSAMAREA+7
  B SENDHTTP

REWITE DS OH DO ONE REWRITE OF UPDATED RECORD
  EXEC CICS RERITE DATASET('FILEA') FROM(VSAMAREA)
  B RETURN

ADD DS OH ADD ONE RECORD TO THE FILE
  EXEC CICS WEB READ FORMFIELD(RECORD) NAMELENGTH(L'RECORD) *
  VALUE(ADDREC) VALUELENGTH(ADDRECL) NOHANDLE
  MVC VSAMAREA+1(6),VSAMKEY
  MVC VSAMAREA+7(73),ADDREC
  EXEC CICS WRITE DATASET('FILEA') RIDFLD(VSAMKEY) *
  FROM(VSAMAREA)
  B SENDHTTP

DELETE DS OH DELETE ONE RECORD FROM THE FILE
  EXEC CICS READ DATASET('FILEA') RIDFLD(VSAMKEY)
  INTO(VSAMAREA) UPDATE
  EXEC CICS DELETE DATASET('FILEA')
  B SENDHTTP

BROWSE DS OH
  EXEC CICS STARTBR DATASET('FILEA') RIDFLD(VSAMKEY)
  EXEC CICS READNEXT DATASET('FILEA') INTO(VSAMAREA) X
  RIDFLD(VSAMKEY)
  EXEC CICS ENDWR DATASET('FILEA')
  MVC MOVEKEY(6),VSAMKEY
  MVC MOVEDATA(73),VSAMAREA+7
  B SENDHTTP

BROWSE10 DS OH DO 'LIMIT / 10' CYCLES OF ...
  MVC KEYB,VSAMKEY ...
  EXEC CICS STARTBR DATASET('FILEA') RIDFLD(KEYB)
  LA 8,10 LOAD LOOP COUNTER ...

80 CICS Transaction Server for VSE/ESA
The sample code below is VSAMBODY.HTML for the VSAMHTML program.

```html
  <p>RECORDKEY</p>
  <input type=text name="KEY" size=6 maxlength=6 value="&KEY;"> 
  <p>RECORDDATA</p>
  <input type=text name="RECORD" size=73 maxlength=73 value="&RECORD;"> 
  <p>FUNCTION</p>
  <input type=submit name="texture" value="Read "> 
  <input type=submit name="texture" value="Add ">
  <input type=submit name="texture" value="Update ">
  <input type=submit name="texture" value="Delete ">
  <input type=submit name="texture" value="Browse ">
  <input type=submit name="texture" value="L-Browse">
</form>
/*
*/
```

The sample code below is VSAMHEAD.HTML for the VSAMHTML program.

```html
<html>
<head>
<title>CICS Web Support: accessing FILEA with CWS</title>
</head>
<body>
<h1>CICS Web Support: accessing FILEA with CWS</h1>
<p>
<h1>VSAM HTML Test Map</h1>
<p/>
/*
*/
```
The sample code below is VSAMFOOT.HTML for the VSAMHTML program.

```
<catalog VSAMFOOT.HTML REPLACE=YES>
<br>
<form method=post>
<input type="submit" name="DFH_CLEAR" value="Clear">
<input type="submit" name="DFH_ENTER" value="Enter">
<input type="reset" value="Reset">
</form>
</body></html>
/*
*/
A.4.3 CICS assembler samples

```assembly
DFHEISTG DSECT
DOCTOKEN DS CL16
CLNTNAML DS F
CLNTNAME DS CL80
SRVRNAML DS F
SRVRNAME DS CL80
CLNTADDR DS CL15
DS 0F
SERVADDR DS F
SERVNAME DS CL8
SSLTYPE DS F
TCPIPSER DS CL8
PRTNUMB DS CL5
DS 0F
EXTRCT1 CSBCT
EXTRCT1 AMODE 31 55200000
EXTRCT1 RMODE ANY 56000000
EXEC CICS HANDLE CONDITION INVREQ(DOCSNSRT)
EXEC CICS DOCUMENT CREATE DOCTOKEN(DOCTOKEN) *
  TEMPLATE('EXTRCTH1')
*
  MVC CLNTNAML,=F'80'
  MVC SRVRNAML,=F'80'
  MVC CLNTADDR,=F'15'
  MVC SERVADDR,=F'15'
* Retrieve some TCPIP related information
  EXEC CICS EXTRACT TCPIP *
    SERVERNAME(SRVRNAME) SNAMELENGTH(SRVRNAML)
  EXEC CICS EXTRACT TCPIP *
    PORTNUMBER(PRTNUMB)
  EXEC CICS EXTRACT TCPIP *
    TCPIPSERVICE(TCPIPSER)
  EXEC CICS EXTRACT TCPIP *
    CLIENTADDR(CLNTADDR) CADDRLENGTH(CLNTADRL)
  EXEC CICS EXTRACT TCPIP *
    CLIENTNAME(CLNTNAME) CNAMELENGTH(CLNTNAML)
  EXEC CICS EXTRACT TCPIP *
    SERVERADDR(SERVADDR) SADDRLENGTH(SERVADRL)
  EXEC CICS EXTRACT TCPIP *
    TCPIPSERVICE(TCPIPSER)
* Insert client name
  DOCINSRT EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(CLNTINFO) LENGTH(CLNTINFL)
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(CLNTNAME) LENGTH(CLNTNAML)
* Insert client address
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(ADDRINFO) LENGTH(ADDRINFL)
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(CLNTADDR) LENGTH(CLNTADRL)
* Insert server name
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(SRVRNAME) LENGTH(SRVRNAML)
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(SRVRADDR) LENGTH(SERVADRL)
* Insert server address
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(SRVRADDR) LENGTH(SERVADRL)
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(SRVRNAME) LENGTH(SRVRNAML)
* Insert TCPIPSERVICE being used
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TCPIPSER) LENGTH(L'TCPIPSER)
EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
  TEXT(TCPIPSER) LENGTH(L'TCPIPSER)
```

Figure 48. Source of CICS assembler program EXTRCT1 (Part 1 of 2)
Figure 49. Source of CICS assembler program EXTRCT1 (Part 2 of 2)

```assembly
* Insert Port number being used
EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(PORTINFO) LENGTH(PORTINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(PRTNUMB) LENGTH(L'PRTNUMB)
* Insert footer for document
EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEMPLATE('EXTRCTF1')
* Send complete document
EXEC CICS WEB SEND DOCTOKEN(DOCTOKEN) *
    CLNTCODEPAGE('iso-8859-1')
EXEC CICS RETURN
CLNTINFO DC C'<p>Client Name : '
CLNTNAML DS F
CLNTNAME DS CL80
SRVRINFO DC C'<p>Server Name : '
SRVRNAML DS F
SRVRNAME DS CL80
CLNTADDR DS F
CLNTADRL DS CL15
DS OF
SERVADDR DS F
SERVADDR DS CL15
DS OF
SSLTYPE DS F
TCPIPSER DS CL8
PRTNUMB DS CL5
DS OF
EXTRCT2 CSECT
EXTRCT2 AMODE 31 55200000
EXTRCT2 RMODE ANY 56000000
EXEC CICS HANDLE CONDITION INVREQ(DOCINSRT)
EXEC CICS DOCUMENT CREATE DOCTOKEN(DOCTOKEN) *
    TEMPLATE('EXTRCTH2')
* MVC CLNTNAML, F'80'
MVC SRVRNAML, F'80'
MVC CLNTADRL, F'15'
MVC SERVADDR, F'15'
*/
```

Figure 50. Source of CICS assembler program EXTRCT2 (Part 1 of 2)
Figure 51. Source of CICS assembler program EXTRCT2 (Part 2 of 2)
Figure 52. Source of CICS assembler program EXTRCT3 (Part 1 of 2)
Figure 53. Source of CICS assembler program EXTRCT3 (Part 2 of 2)

```assembly
* Insert PORT number being used
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
  TEXT(PORTINFO) LENGTH(PORTINFL)
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
  TEXT(PORTNUMB) LENGTH(L'PRTNUMB)
* Insert footer for document
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
  TEMPLATE('EXTRCTF3')
* Send complete document
  EXEC CICS WEB SEND DOCTOKEN(DOCTOKEN) *
  CLNTCODEPAGE('iso-8859-1')
  EXEC CICS RETURN
CLNTINFO DC C'\p>Client Name : '  
CLNTINFL DC F'17'
SRVRINFO DC C'\p>Server Name : '  
SRVRINFL DC F'17'
ADDRINFO DC C'\p>Client Address : '  
ADDRINFL DC F'15'
SERVADDR DS F'15'
SSLTYPE DS F  
TCPIPSER DS CL8
PRTNUMB DS CL5
TEXTL ds CL1
TEXTL DS F  
EXTRCT4 CSECT
EXTRCT4 AMODE 31  55200000
EXTRCT4 RMODE ANY  56000000
  EXEC CICS HANDLE CONDITION INVREQ(DOCINSRT)  
  EXEC CICS DOCUMENT CREATE DOCTOKEN(DOCTOKEN) *
  TEMPLATE('EXTRCTH4')
  MVC CLNTNAML,=F'80'
  MVC SRVRNAML,=F'80'
  MVC CLNTADDR,=F'15'
  MVC SERVADDR,=F'15'
  MVC TEXTL,=F'1'
```

Figure 54. Source of CICS assembler program EXTRCT4 (Part 1 of 3)
* Retrieve some TCPIP related information
EXEC CICS EXTRACT TCPIP *
  SERVERNAME(SRVNamE) SNAMELENGTH(SRVNAML)
EXEC CICS EXTRACT TCPIP *
  SERVERADDR(SRVADDR) SADDRLENGTH(SRVADRL)
EXEC CICS EXTRACT TCPIP *
  TCPPIPSERVICE(TCPIPSER)
EXEC CICS EXTRACT TCPIP *
  PORTNUMBER(PRTNMB)
EXEC CICS EXTRACT TCPIP *
  CLIENTADDR(CLNTADDR) CADDRLENGTH(CLNTADRL)
EXEC CICS EXTRACT TCPIP *
  CLIENTNAME(CLNTNAME) CNAMELENGTH(CLNTNAML)

* Analyse input.
DOINSERT EXEC CICS WEB READ FORMFIELD(TEXTURE) NAMELENGTH(L'TEXTURE) *
  VALUE(TEXT) VALUELENGTH(TEXTL) NOHANDLE
  CL1 TEXT,C'1'
  BE TEXT1
  CL1 TEXT,C'2'
  BE TEXT2
  CL1 TEXT,C'3'
  BE TEXT3
  CL1 TEXT,C'4'
  BE TEXT4
  CL1 TEXT,C'5'
  BE TEXT5
  CL1 TEXT,C'6'
  BE TEXT6
  B ENDBODY

  TEXT1 EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TEXT1INFO) LENGTH(TEXT1NFL)
  B ENDBODY

  TEXT2 EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TEXT2INFO) LENGTH(TEXT2NFL)
  B ENDBODY

  TEXT3 EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TEXT3INFO) LENGTH(TEXT3NFL)
  B ENDBODY

  TEXT4 EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TEXT4INFO) LENGTH(TEXT4NFL)
  B ENDBODY

  TEXT5 EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TEXT5INFO) LENGTH(TEXT5NFL)
  B ENDBODY

  TEXT6 EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(TEXT6INFO) LENGTH(TEXT6NFL)
  B ENDBODY

  * Close Body
  ENDBODY EQU *
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(BODYINFO) LENGTH(BODYINFL)

  * Insert image name
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(IMGINFO) LENGTH(IMGINFL)

  * Insert header
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(HEADER) LENGTH(HEADERL)

  * Insert client name
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(CLNTINFO) LENGTH(CLNTINFL)
  EXEC CICS DOCUMENT INSERT DOCTOKEN(DOCTOKEN) *
    TEXT(CLNTNAME) LENGTH(CLNTNAML)

Figure 55. Source of CICS assembler program EXTRCT4 (Part 2 of 3)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (ADDRINFO) LENGTH (ADDRINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (CLNTADDR) LENGTH (CLNTADRL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (SRVRINFO) LENGTH (SRVRINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (SRVRNAME) LENGTH (SRVRNAML)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (SDDRINFO) LENGTH (SDDRINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (SERVADDR) LENGTH (SERVADRL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (TCPSINFO) LENGTH (TCPSINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (TCPIPSER) LENGTH (L'TCPIPSER)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (PORTINFO) LENGTH (PORTINFL)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEXT (PRTNUMB) LENGTH (L'PRTNUMB)
EXEC CICS DOCUMENT INSERT DOCTOKEN (DOCTOKEN) TEMPLATE ('EXTRCTF4')
EXEC CICS WEB SEND DOCTOKEN (DOCTOKEN) CLNTCODEPAGE ('iso-8859-1')
EXEC CICS RETURN
CLNTINFO DC C' Client Name : '
CLNTINFL DC F' 17'
SRVRINFO DC C' Client Address : '
SRVRINFL DC F' 17'
ADDRINFO DC C' Server Name : '
ADDRINFL DC F' 20'
SDDRINFO DC C' Server Address : '
SDDRINFL DC F' 20'
TCPSINFO DC C' TCPIPSERVICE name: '
TCPSINFL DC F' 22'
PORTINFO DC C' on Port Number : '
PORTINFL DC F' 19'
IMAGINFO DC C' image alt="" src="/DFHWBIMG/masthead.gif"'
IMAGINFL DC F' 41'
HEADER DC C' "EXTRACT - Running CICS Web Support on VSE/ESA"'
DC C' with a masthead and a background'
DC C' <hr>
HEADERL DC A(*-HEADER)
TEXTINFO DC C' background="/DFHWBIMG/texture1.jpeg"
TEXTINFO DC A(*-TEXTINFO)
TEXTINFO DC C' background="/DFHWBIMG/texture2.jpeg"
TEXTINFO DC A(*-TEXTINFO)
TEXTINFO DC C' background="/DFHWBIMG/texture3.jpeg"
TEXTINFO DC A(*-TEXTINFO)
TEXTINFO DC C' background="/DFHWBIMG/texture4.jpeg"
TEXTINFO DC A(*-TEXTINFO)
TEXTINFO DC C' background="/DFHWBIMG/texture5.jpeg"
TEXTINFO DC A(*-TEXTINFO)
TEXTINFO DC C' background="/DFHWBIMG/texture6.jpeg"
TEXTINFO DC A(*-TEXTINFO)
BODYINFO DC C' '
BODYINFL DC F' 1'
TEXTURE DC C' TEXTURE'
END EXTRCT4
/*

Figure 56. Source of CICS assembler program EXTRCT4 (Part 3 of 3)
A.4.4 Catalog jobs

Figure 57. Sample job to compile, link and catalog CICS assembler programs

```assembler
* $$ JOB JNM=&JOBNAME,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
* $$ FEN DISP=I,DEST=*,PRI=9,CLASS=A
// JOB &JOBNAME TRANSLATE PROGRAM &PROGNAME
// ASSGN SYSIPT, SYSRDR
// EXEC EISINSRT
$ $$ LST DISP=D,CLASS=Q,PRI=3
// JOB &JOBNAME COMPIL PROGRAM &PROGNAME
// SETPARM CATALOG=&CATALOG
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION ERRS, SXREF, SYM, LIST, NODECK
// GOTO ENDCAT
/. CAT
// LIBDEF PHASE, CATALOG=DFHHTML.DFHDOC
// OPTION ERRS, SXREF, SYM, CATAL, NODECK
// PHASE &PROGNAME,*
// INCLUDE DFHEAI
/. ENDCAT
// EXEC ASMA90, SIZE=(ASMA90, 64K), PARM='EXIT(LIBEXIT(EDECKXIT)), SIZE(MAXC -200K, ABOVE)'
* $$ END
// ON $CANCEL OR $ABEND GOTO ENDJ2
// OPTION NOLIST, NODUMP, DECK
// EXEC DFHEAP1$, SIZE=512K
*ASM XOPTS(CICS)
* $$ SLI ICCF=(&PROGNAME, &PASSWORD), LIB=(&LIBNO)
/.*
/. ENDJ2
// EXEC EISINSRT
/.*
// IF CATALOG NH 1 OR $MRC GT 4 THEN
// GOTO NOLNK
// EXEC LNKD,T, SIZE=256K
/. NOLNK
#4
* $$ EOJ
* $$ END
/.*
* $$ EOJ
```

Figure 58. Sample job to catalog the HTML documents into sublib DFHHTML.DFHDOC

```assembler
* $$ JOB JNM=CATMEMB,DISP=D,CLASS=0
// JOB CATMEMB CATALOG MEMBER INTO AF-LIBRARY
// EXEC LIBR, PARM='MSHP'
ACCESS S-DFHHTML.DFHDOC
* $$ SLI ICCF=(xxxxx), LIB=(xx)
/.*
/.*
* $$ EOJ
```
### A.5 Sample listings used with the 3270 bridge support

```plaintext
// JOB FILEACLU DEFINE FILE FILEA FOR TESTING 3270 BRIDGE
// EXEC IDCAMS,SIZE=AUTO
DELETE (CICSTS11.SAMPLE.FILEA) CL NOERASE PURGE -
CATALOG (VSESP.USER.CATALOG )
DEFINE CLUSTER ( -
  NAME (CICSTS11.SAMPLE.FILEA ) -
  CYLINDERS(1 1 ) -
  SHAREOPTIONS (2) -
  RECORDSIZE (80 80 ) -
  VOLUMES (DOSRES ) -
  NOREUSE -
  INDEXED -
  FREESPACE (15 7) -
  KEYS (6 1 ) -
  NOCOMPRESSED -
  TO (99366)) -
  DATA (NAME (CICSTS11.SAMPLE.FILEA.@D@ ) -
  CONTROLINTERVALSIZE (4096 )) -
  INDEX (NAME (CICSTS11.SAMPLE.FILEA.@I@ )) -
  CATALOG (VSESP.USER.CATALOG )
IF LASTCC NE 0 THEN CANCEL JOB
/*
// OPTION STDLABEL=ADD
// DLBL FILEA,'CICSTS11.SAMPLE.FILEA',,VSAM, X
CAT=VSESPUC
*/
// EXEC IESVCLUP,SIZE=AUTO
A CICSTS11.SAMPLE.FILEA FILEA VSESPUC
*/
/*

Figure 59. IDCAMS definitions for user VSAM file to test 3270 bridge
```
Figure 60. IDCAMS repro job to load VSAM file for testing 3270 bridge facility
Figure 61. Source listing for program VSAMTEST for testing 3270 bridge (Part 1 of 4)

* VSAMTEST -- PROGRAM FOR VSAMTEST TEST SUITE. *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

PRINT GEN

***********************************************************************
DFHEISTG DSECT
COPY VSAMSET
STARTCLK DS D
STOPCLK DS D
WORK DS D
VSAMDATA DS CL80 I/O AREA FOR VSAMDATA-DATASET
SCRATCH DS CL80 SCRATCH INPUT AREA USED FOR UPDATE
***********************************************************************

* PROGRAM START *
***********************************************************************
PRINT NOGEN

VSAMTEST CSECT
B MVC
DC CL8'VSAMTEST'
MVC DS 0H
STCK STARTCLK RECORD START TIME
EXEC CICS HANDLE CONDITION NOTFND(NOTFOUND)
EXEC CICS HANDLE CONDITION DSIDERR(DSIDERR)
EXEC CICS HANDLE CONDITION EOC(REWRITE)
EXEC CICS HANDLE CONDITION ERROR(ERROR)
EXEC CICS HANDLE CONDITION MAPFAIL(MAPFAIL)

RECEIVE DS 0H
EXEC CICS RECEIVE MAP('VSAMMAP') MAPSET('VSAMSET')
MVC RESPONSE(29),BLANK
OI FUNCI,C' ' OR A BLANK TO FORCE UPPERCASE
CLI FUNCI,C'A' IF FUNCTION IS NOT ADD, CLEAR INPUT
BE NOBLANK
MVC RECORDI(244),BLANK
NOBLANK DS 0H
CLI FUNCI,C' R' IF FUNCTION IS READ
BE READ
CLI FUNCI,C'U' IF FUNCTION IS UPDATE
BE UPDATE
CLI FUNCI,C'A' IF FUNCTION IS ADD
BE ADD
CLI FUNCI,C'D' IF FUNCTION IS DELETE
BE DELETE
CLI FUNCI,C'B' IF FUNCTION IS BROWSE
BE BROWSE
CLI FUNCI,C'L' IF FUNCTION IS LONG BROWSE (10 RECORDS)
BE BROWSE10
CLI FUNCI,C'E' IF FUNCTION IS END BROWSE
BE ENDBR
MVC INVINPUT OTHERWISE, SAY INVALID INPUT
MAPFAIL DS 0H SEND BLANK MAP ON FIRST TIME ENTRY
EXEC CICS SEND MAP('VSAMMAP') MAPSET('VSAMSET') MAPONLY WAIT X
FREEEEKS
B RETURN

READ DS 0H DO READ OF SPECIFIED RECORD
EXEC CICS READ DATASET('FILEA') RIDFLD(KEYI) INTO(VSAMDATA)
MVC KEYI(6),VSAMDATA+1
MVC RECORDI(73),VSAMDATA+7
B SENDNORM WHEN FINISHED, SEND NORMAL RESPONSE

UPDATE DS 0H DO READ/UPDATE OF RECORD
EXEC CICS READ DATASET('FILEA') RIDFLD(KEYI) INTO(VSAMDATA)
MVC KEYI(6),VSAMDATA+1
MVC RECORDI(73),VSAMDATA+7
MVC RESPONSE,SENDREN
EXEC CICS SEND MAP('VSAMMAP') MAPSET('VSAMSET')
FREEEEKS
EXEC CICS RECEIVE MAP('VSAMMAP') MAPSET('VSAMSET')
MVC VSAMDATA+1(6),KEYI
MVC VSAMDATA+7(74),RECORDI

REWRITE DS 0H DO ONE REWRITE OF UPDATED RECORD
EXEC CICS REWRITE DATASET('FILEA') FROM(VSAMDATA)
B SENDNORM WHEN FINISHED, SEND NORMAL RESPONSE
SENDNORM DS 0H SEND NORMAL RESPONSE
MVC RESPONSE,NORMRESP
Figure 62. Source listing for program VSAMTEST for testing 3270 bridge (Part 2 of 4)
Figure 63. Source listing for program VSAMTEST for testing 3270 bridge (Part 3 of 4)
Figure 64. Source listing for program VSAMTEST for testing 3270 bridge (Part 4 of 4)

The source for mapset VSAMSET is listed in Figure 22 on page 49.

We defined both the RDO definitions for file FILEA and transaction VSAM in Group FILEA.

```
ZERO DC PL6'0' PACKED ZERO
ONE DC PL6'1' PACKED ONE
TEN DC PL6'10' PACKED TEN
KEYB DS CL6 FIELD FOR L10 BROWSE
   SPACE 2
   END VSAMTEST
/*
```

Figure 65. CEDA display of group FILEA

```
DI G(FILEA)
ENTER COMMANDS
NAME  TYPE  GROUP  DATE   TIME
FILEA FILE  FILEA  00.250 15.57.16
VSAM  TRANSACTION FILEA  00.250 16.09.11
```
OBJECT CHARACTERISTICS
CEDA View File( FILEA )

File : FILEA
Group : FILEA
DESCRIPTION : FILE FOR TESTING 3270 BRIDGE

VSAM PARAMETERS
DSName : CICSTS11.SAMPLE.FILEA
Password : PASSWORD NOT SPECIFIED
Lsrpoolid : 01 1-15 | None
Catname : VSESPUC
DSNSharing : Noreqs Noreqs | Allreqs | Modifyreqs
STRings : 001 1-255
Nsrgrp :
SRkaccess : Key Key | Rba

REMOTE ATTRIBUTES

REMOTE System : 
REMTENName : 

RECORDSsize : 00080 1-32767
* Keylength : 008 1-255

* INITIAL STATUS
STAtus : Enabled Enabled | Disabled | Unenabled
Opentime : Firstref Firstref | Startup

BUFFERS
Databuffers : 00002 2-32767
Indexbuffers : 00001 1-32767

DATATABLE PARAMETERS
Table : No No | Cics | User
Maxnumrecs : 16-16777215

DATA FORMAT
RECORDFormat : F V | F

OPERATIONS
Add : Yes No | Yes
Browse : Yes No | Yes
DELETE : Yes No | Yes
READ : Yes Yes | No
* Update : Yes No | Yes

* AUTO JOURNALING
Journal : No No | 1-99
JNLRead : None None | Updateonly | Readonly | All
JNLSyncRead : No No | Yes
JNLUpdate : No No | Yes
JNLAdd : None None | Before | After | All
JNLSyncWrite : Yes Yes | No

RECOVERY PARAMETERS
RECOvery : None None | Backoutonly | All
PWDRecovlog : No No | 1-99

Figure 66. RDO file definition for our test VSAM file FILEA
<table>
<thead>
<tr>
<th>OBJECT CHARACTERISTICS</th>
<th>CICS R</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEDA View TRANSACTION( VSAM )</td>
<td></td>
</tr>
<tr>
<td>TRANSACTION : VSAM</td>
<td></td>
</tr>
<tr>
<td>Group : FILEA</td>
<td></td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td></td>
</tr>
<tr>
<td>PROGram : VSAMTEST</td>
<td></td>
</tr>
<tr>
<td>TMAsize : 00000 0-32767</td>
<td></td>
</tr>
<tr>
<td>PROFile : DFHCICST</td>
<td></td>
</tr>
<tr>
<td>PARTitionset</td>
<td></td>
</tr>
<tr>
<td>STATUS : Enabled Enabled</td>
<td>Disabled</td>
</tr>
<tr>
<td>PRIMedsize : 00000 0-65520</td>
<td></td>
</tr>
<tr>
<td>TASKDATALoc : Below Below</td>
<td>Any</td>
</tr>
<tr>
<td>TASKDATAKey : User User</td>
<td>Cics</td>
</tr>
<tr>
<td>STOREageclear : No No</td>
<td>Yes</td>
</tr>
<tr>
<td>Runaway : System System</td>
<td>0 500-270000</td>
</tr>
<tr>
<td>Shutdown : Disabled Disabled</td>
<td>Enabled</td>
</tr>
<tr>
<td>BRexit</td>
<td></td>
</tr>
<tr>
<td>REMOTE ATTRIBUTES</td>
<td></td>
</tr>
<tr>
<td>* Dynamic : No No Yes</td>
<td></td>
</tr>
<tr>
<td>CEDA View TRANSACTION( VSAM )</td>
<td></td>
</tr>
<tr>
<td>* REMOTESystem :</td>
<td></td>
</tr>
<tr>
<td>REMOTEName :</td>
<td></td>
</tr>
<tr>
<td>TNProf :</td>
<td></td>
</tr>
<tr>
<td>Localg : No Yes</td>
<td></td>
</tr>
<tr>
<td>SCHEDULING</td>
<td></td>
</tr>
<tr>
<td>PRIORity : 001 0-255</td>
<td></td>
</tr>
<tr>
<td>TClass : No No 1-10</td>
<td></td>
</tr>
<tr>
<td>TRANClass : DFHTCL00</td>
<td></td>
</tr>
<tr>
<td>ALIASES</td>
<td></td>
</tr>
<tr>
<td>Alias :</td>
<td></td>
</tr>
<tr>
<td>TASKReq :</td>
<td></td>
</tr>
<tr>
<td>XTRanid :</td>
<td></td>
</tr>
<tr>
<td>TRName :</td>
<td></td>
</tr>
<tr>
<td>XTPname :</td>
<td></td>
</tr>
<tr>
<td>* RECOVERY</td>
<td></td>
</tr>
<tr>
<td>DTimout : No No 1-6800</td>
<td></td>
</tr>
<tr>
<td>Indoubt : Backout Backout</td>
<td>Commit Wait</td>
</tr>
<tr>
<td>RESTart : No No Yes</td>
<td></td>
</tr>
<tr>
<td>SPurge : No No Yes</td>
<td></td>
</tr>
<tr>
<td>TFFlgge : No No Yes</td>
<td></td>
</tr>
<tr>
<td>DUMP : Yes Yes No</td>
<td></td>
</tr>
<tr>
<td>TRACe : Yes Yes No</td>
<td></td>
</tr>
<tr>
<td>CONFdata : No No Yes</td>
<td></td>
</tr>
<tr>
<td>SECURITY</td>
<td></td>
</tr>
<tr>
<td>RESSec : No No Yes</td>
<td></td>
</tr>
<tr>
<td>CMdsec : No No Yes</td>
<td></td>
</tr>
<tr>
<td>Extsec : No No Yes</td>
<td></td>
</tr>
<tr>
<td>TRANSec : 01 1-64</td>
<td></td>
</tr>
<tr>
<td>RES : 00 0-24 Public</td>
<td></td>
</tr>
<tr>
<td>* CONFdata : No No Yes</td>
<td></td>
</tr>
<tr>
<td>SECURITY</td>
<td></td>
</tr>
<tr>
<td>RESSec : No No Yes</td>
<td></td>
</tr>
<tr>
<td>CMdsec : No No Yes</td>
<td></td>
</tr>
<tr>
<td>Extsec : No No Yes</td>
<td></td>
</tr>
<tr>
<td>TRANSec : 01 1-64</td>
<td></td>
</tr>
<tr>
<td>RES : 00 0-24 Public</td>
<td></td>
</tr>
</tbody>
</table>

Figure 67. RDO transaction definition for our test transaction VSAM
A.6 DFHCNV source code

```
* $S JOB JNM=DFHCNV,CLASS=0,DISP=D
// JOB DFHCNV CONVERSION TABLE FOR CICS CLIENT CONNECTIONS
// LIBDEF *,CATALOG=PRD2.CONFIG
// OPTION CATAL,NOXREF,LOG
* THIS DFHCNV INCLUDES THE DEFINITION FOR CICS WEB SUPPORT
* DFHWBHH IS FOR CONVERTING HTTP HEADER DATA.
* DFHWBUD IS FOR CONVERTING HTTP USER DATA.
// EXEC ASSEMBLY,SIZE=2000K
PRINT NONE
DFHCNV TYPE=INITIAL
  DFHCNV TYPE=ENTRY,RTYPE=PC,ENAME=DFHWBHH,
    CLINTCP=8859-1,SRVERCP=037,USREXIT=NO
  DFHCNV TYPE=SELECT,OPTION=DEFAULT
  DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=32767,
    LAST=YES
  DFHCNV TYPE=ENTRY,RTYPE=PC,ENAME=DFHWBUD,
    CLINTCP=8859-1,SRVERCP=037,USREXIT=NO
  DFHCNV TYPE=SELECT,OPTION=DEFAULT
  DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=32767,
    LAST=YES
DFHCNV TYPE=FINAL
END
/*
// EXEC LINKEDIT
*/
/*
* $S EJ
***** END OF FILE *****
```

Figure 68. DFHCNV conversion table

A.7 Source listing DFHWBEP sample program

```
DPHEISTG DSCT
DWORD DS D
COPY DFHWBEP
DFHWBEP CSECT
DFHWBEP AMODE 31
DFHWBEP RMODE ANY
*
WBEPR EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
*
EXEC CICS ADDRESS COMMAREA(WBEPR) , ask for COMMAREA address
CLC EIBRESP,DFHRESP(NORMAL) was it good request ?
BNK WBEPR problem, go to common abend
LTR WBEPR,WBEPR do minimal testing on len
BE WBEPR common abend
*
USING DFHWBEP,CWBEPR addressability to COMMAREA
*
* We do not test on the length of the buffer. We know we receive a
* COMMAREA of 32K and are sure that we will not use that much of it.
*
L R5,WBEPR_RESPONSE_PTR pointer to HTTP response
LA R6,0 R6 = 0 to contain datalength
L R7,WBEPR_RESPONSE_LEN R7 = length of HTTP response
LTR R7,R7 Mike Poil
BNP NOBODY Mike Poil
FINDSTRT EQU *
CLC ="</body>' ,0(R5) look for the <body> tag
BN ADDOWN found, we start adding
*
LA R5,1(R5) not found, add 1 to pointer
LA R6,1(R6) and increment length with 1
BCT R7,FINDSTRT decrement resp_len with 1
NOBODY DS 0H Mike Poil
```
L R5,WEBP_RESPONSE_PTR no <body> tag found
L R6,WEBP_RESPONSE_LEN add length to pointer and
AR R5,R6 start adding from there

* We are going to scan through the COMMAREA that we received and
* append all meaningful information that we find in there to
* HTTP response.
*
AZDOWN EQU *
MVC 0(MYHEADL,R5),MYHEAD move our header
LA R5,MYHEADL(R5) adjust pointer
LA R6,MYHEADL(R6) adjust length
CLC WEBP_ERROR_CODE,=H'0' received an error code ?
BE ABENDCOD no, go check abend code

* For the moment there are 43 wbep error codes provided in DFHWBUCD.
* We test on the numeric value of the error code and write the
* corresponding text as given in DFHWBUCD.
*
MVC 0(17,R5),=C'WEBP_ERROR_CODE: ' yes, write field name
LA R5,17(R5) adjust pointer
LA R6,17(R6) adjust length
CLC WEBP_ERROR_CODE,=H'1' was it error code 1 ?
BNE ERRCOD2 no, see if it is 2
MVC 0(L'WBERR1,R5),WBERR1 yes, write text
LA R5,L'WBERR1(R5) adjust pointer
LA R6,L'WBERR1(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD2 EQU *
CLC WEBP_ERROR_CODE,=H'2' was it error code 2 ?
BNE ERRCOD3 no, see if it is 3
MVC 0(L'WBERR2,R5),WBERR2 yes, write text
LA R5,L'WBERR2(R5) adjust pointer
LA R6,L'WBERR2(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD3 EQU *
CLC WEBP_ERROR_CODE,=H'3' was it error code 3 ?
BNE ERRCOD4 no, see if it is 4
MVC 0(L'WBERR3,R5),WBERR3 yes, write text
LA R5,L'WBERR3(R5) adjust pointer
LA R6,L'WBERR3(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD4 EQU *
CLC WEBP_ERROR_CODE,=H'4' was it error code 4 ?
BNE ERRCOD5 no, see if it is 5
MVC 0(L'WBERR4,R5),WBERR4 yes, write text
LA R5,L'WBERR4(R5) adjust pointer
LA R6,L'WBERR4(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD5 EQU *
CLC WEBP_ERROR_CODE,=H'5' was it error code 5 ?
BNE ERRCOD6 no, see if it is 6
MVC 0(L'WBERR5,R5),WBERR5 yes, write text
LA R5,L'WBERR5(R5) adjust pointer
LA R6,L'WBERR5(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD6 EQU *
CLC WEBP_ERROR_CODE,=H'6' was it error code 6 ?
BNE ERRCOD7 no, see if it is 7
MVC 0(L'WBERR6,R5),WBERR6 yes, write text
LA R5,L'WBERR6(R5) adjust pointer
LA R6,L'WBERR6(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD7 EQU *
CLC WEBP_ERROR_CODE,=H'7' was it error code 7 ?
BNE ERRCOD8 no, see if it is 8
MVC 0(L'WBERR7,R5),WBERR7 yes, write text
LA R5,L'WBERR7(R5) adjust pointer
LA R6,L'WBERR7(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD8 EQU *
CLC WEBP_ERROR_CODE,=H'8' was it error code 8 ?
BNE ERRCOD9 no, see if it is 9
MVC 0(L'WBERR8,R5),WBERR8 yes, write text
LA R5,L'WBERR8(R5) adjust pointer
LA R6,L'WBERR8(R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD9 EQU *
CLC  WBEP_ERROR_CODE,=H'9'  was it error code 9 ?
BNK  ERRCOD10  no, see if it is 10
MVC  0(L'WBERR9,R5),WBERR9  yes, write text
LA  R5,L'WBERR9(,R5)  adjust pointer
LA  R6,L'WBERR9(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD10 EQU *

CLC  WBEP_ERROR_CODE,=H'10'  was it error code 10 ?
BNK  ERRCOD11  no, see if it is 11
MVC  0(L'WBERR10,R5),WBERR10  yes, write text
LA  R5,L'WBERR10(,R5)  adjust pointer
LA  R6,L'WBERR10(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD11 EQU *

CLC  WBEP_ERROR_CODE,=H'11'  was it error code 11 ?
BNK  ERRCOD12  no, see if it is 12
MVC  0(L'WBERR11,R5),WBERR11  yes, write text
LA  R5,L'WBERR11(,R5)  adjust pointer
LA  R6,L'WBERR11(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD12 EQU *

CLC  WBEP_ERROR_CODE,=H'12'  was it error code 12 ?
BNK  ERRCOD13  no, see if it is 13
MVC  0(L'WBERR12,R5),WBERR12  yes, write text
LA  R5,L'WBERR12(,R5)  adjust pointer
LA  R6,L'WBERR12(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD13 EQU *

CLC  WBEP_ERROR_CODE,=H'13'  was it error code 13 ?
BNK  ERRCOD14  no, see if it is 14
MVC  0(L'WBERR13,R5),WBERR13  yes, write text
LA  R5,L'WBERR13(,R5)  adjust pointer
LA  R6,L'WBERR13(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD14 EQU *

CLC  WBEP_ERROR_CODE,=H'14'  was it error code 14 ?
BNK  ERRCOD15  no, see if it is 15
MVC  0(L'WBERR14,R5),WBERR14  yes, write text
LA  R5,L'WBERR14(,R5)  adjust pointer
LA  R6,L'WBERR14(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD15 EQU *

CLC  WBEP_ERROR_CODE,=H'15'  was it error code 15 ?
BNK  ERRCOD16  no, see if it is 16
MVC  0(L'WBERR15,R5),WBERR15  yes, write text
LA  R5,L'WBERR15(,R5)  adjust pointer
LA  R6,L'WBERR15(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD16 EQU *

CLC  WBEP_ERROR_CODE,=H'16'  was it error code 16 ?
BNK  ERRCOD17  no, see if it is 17
MVC  0(L'WBERR16,R5),WBERR16  yes, write text
LA  R5,L'WBERR16(,R5)  adjust pointer
LA  R6,L'WBERR16(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD17 EQU *

CLC  WBEP_ERROR_CODE,=H'17'  was it error code 17 ?
BNK  ERRCOD18  no, see if it is 18
MVC  0(L'WBERR17,R5),WBERR17  yes, write text
LA  R5,L'WBERR17(,R5)  adjust pointer
LA  R6,L'WBERR17(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD18 EQU *

CLC  WBEP_ERROR_CODE,=H'18'  was it error code 18 ?
BNK  ERRCOD19  no, see if it is 19
MVC  0(L'WBERR18,R5),WBERR18  yes, write text
LA  R5,L'WBERR18(,R5)  adjust pointer
LA  R6,L'WBERR18(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD19 EQU *

CLC  WBEP_ERROR_CODE,=H'19'  was it error code 19 ?
BNK  ERRCOD20  no, see if it is 20
MVC  0(L'WBERR19,R5),WBERR19  yes, write text
LA  R5,L'WBERR19(,R5)  adjust pointer
LA  R6,L'WBERR19(,R6)  adjust length
B  ABENDCOD  and test for an abend code

ERRCOD20 EQU *

CLC  WBEP_ERROR_CODE,=H'20'  was it error code 20 ?
BNE ERRCOD21

MVC 0(L'WBERR20,R5),WBERR20
LA R5,L'WBERR20(,R5) adjust pointer
LA R6,L'WBERR20(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD21 EQU *

CLC WBEP_ERROR_CODE,=H'21' was it error code 21 ?
BNE ERRCOD22 no, see if it is 22
MVC 0(L'WBERR21,R5),WBERR21
LA R5,L'WBERR21(,R5) adjust pointer
LA R6,L'WBERR21(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD22 EQU *

CLC WBEP_ERROR_CODE,=H'22' was it error code 22 ?
BNE ERRCOD23 no, see if it is 23
MVC 0(L'WBERR22,R5),WBERR22
LA R5,L'WBERR22(,R5) adjust pointer
LA R6,L'WBERR22(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD23 EQU *

CLC WBEP_ERROR_CODE,=H'23' was it error code 23 ?
BNE ERRCOD24 no, see if it is 24
MVC 0(L'WBERR23,R5),WBERR23
LA R5,L'WBERR23(,R5) adjust pointer
LA R6,L'WBERR23(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD24 EQU *

CLC WBEP_ERROR_CODE,=H'24' was it error code 24 ?
BNE ERRCOD25 no, see if it is 25
MVC 0(L'WBERR24,R5),WBERR24
LA R5,L'WBERR24(,R5) adjust pointer
LA R6,L'WBERR24(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD25 EQU *

CLC WBEP_ERROR_CODE,=H'25' was it error code 25 ?
BNE ERRCOD26 no, see if it is 26
MVC 0(L'WBERR25,R5),WBERR25
LA R5,L'WBERR25(,R5) adjust pointer
LA R6,L'WBERR25(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD26 EQU *

CLC WBEP_ERROR_CODE,=H'26' was it error code 26 ?
BNE ERRCOD27 no, see if it is 27
MVC 0(L'WBERR26,R5),WBERR26
LA R5,L'WBERR26(,R5) adjust pointer
LA R6,L'WBERR26(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD27 EQU *

CLC WBEP_ERROR_CODE,=H'27' was it error code 27 ?
BNE ERRCOD28 no, see if it is 28
MVC 0(L'WBERR27,R5),WBERR27
LA R5,L'WBERR27(,R5) adjust pointer
LA R6,L'WBERR27(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD28 EQU *

CLC WBEP_ERROR_CODE,=H'28' was it error code 28 ?
BNE ERRCOD29 no, see if it is 29
MVC 0(L'WBERR28,R5),WBERR28
LA R5,L'WBERR28(,R5) adjust pointer
LA R6,L'WBERR28(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD29 EQU *

CLC WBEP_ERROR_CODE,=H'29' was it error code 29 ?
BNE ERRCOD30 no, see if it is 30
MVC 0(L'WBERR29,R5),WBERR29
LA R5,L'WBERR29(,R5) adjust pointer
LA R6,L'WBERR29(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD30 EQU *

CLC WBEP_ERROR_CODE,=H'30' was it error code 30 ?
BNE ERRCOD31 no, see if it is 31
MVC 0(L'WBERR30,R5),WBERR30
LA R5,L'WBERR30(,R5) adjust pointer
LA R6,L'WBERR30(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD31 EQU *

CLC WBEP_ERROR_CODE,=H'31' was it error code 31 ?
BNE ERRCOD32 no, see if it is 32
Appendix A. Listings

MVC 0(L’WBERR31,R5),WBERR31 yes, write text
LA R5,L’WBERR31(,R5) adjust pointer
LA R6,L’WBERR31(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO32 EQU *
CLC WBEP_ERROR_CODE,=H’32’ was it error code 32 ?
BNK ERRCDO33 no, see if it is 33
MVC 0(L’WBERR32,R5),WBERR32 yes, write text
LA R5,L’WBERR32(,R5) adjust pointer
LA R6,L’WBERR32(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO33 EQU *
CLC WBEP_ERROR_CODE,=H’33’ was it error code 33 ?
BNK ERRCDO34 no, see if it is 34
MVC 0(L’WBERR33,R5),WBERR33 yes, write text
LA R5,L’WBERR33(,R5) adjust pointer
LA R6,L’WBERR33(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO34 EQU *
CLC WBEP_ERROR_CODE,=H’34’ was it error code 34 ?
BNK ERRCDO35 no, see if it is 35
MVC 0(L’WBERR34,R5),WBERR34 yes, write text
LA R5,L’WBERR34(,R5) adjust pointer
LA R6,L’WBERR34(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO35 EQU *
CLC WBEP_ERROR_CODE,=H’35’ was it error code 35 ?
BNK ERRCDO36 no, see if it is 36
MVC 0(L’WBERR35,R5),WBERR35 yes, write text
LA R5,L’WBERR35(,R5) adjust pointer
LA R6,L’WBERR35(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO36 EQU *
CLC WBEP_ERROR_CODE,=H’36’ was it error code 36 ?
BNK ERRCDO37 no, see if it is 37
MVC 0(L’WBERR36,R5),WBERR36 yes, write text
LA R5,L’WBERR36(,R5) adjust pointer
LA R6,L’WBERR36(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO37 EQU *
CLC WBEP_ERROR_CODE,=H’37’ was it error code 37 ?
BNK ERRCDO38 no, see if it is 38
MVC 0(L’WBERR37,R5),WBERR37 yes, write text
LA R5,L’WBERR37(,R5) adjust pointer
LA R6,L’WBERR37(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO38 EQU *
CLC WBEP_ERROR_CODE,=H’38’ was it error code 38 ?
BNK ERRCDO39 no, see if it is 39
MVC 0(L’WBERR38,R5),WBERR38 yes, write text
LA R5,L’WBERR38(,R5) adjust pointer
LA R6,L’WBERR38(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO39 EQU *
CLC WBEP_ERROR_CODE,=H’39’ was it error code 39 ?
BNK ERRCDO40 no, see if it is 40
MVC 0(L’WBERR39,R5),WBERR39 yes, write text
LA R5,L’WBERR39(,R5) adjust pointer
LA R6,L’WBERR39(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO40 EQU *
CLC WBEP_ERROR_CODE,=H’40’ was it error code 40 ?
BNK ERRCDO41 no, see if it is 41
MVC 0(L’WBERR40,R5),WBERR40 yes, write text
LA R5,L’WBERR40(,R5) adjust pointer
LA R6,L’WBERR40(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO41 EQU *
CLC WBEP_ERROR_CODE,=H’41’ was it error code 41 ?
BNK ERRCDO42 no, see if it is 42
MVC 0(L’WBERR41,R5),WBERR41 yes, write text
LA R5,L’WBERR41(,R5) adjust pointer
LA R6,L’WBERR41(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCDO42 EQU *
CLC WBEP_ERROR_CODE,=H’42’ was it error code 42 ?
BNK ERRCDO43 no, see if it is 43
MVC 0(L’WBERR42,R5),WBERR42 yes, write text
LA R5,L'WBERR42(,R5) adjust pointer
LA R6,L'WBERR42(,R6) adjust length
B ABENDCOD and test for an abend code

ERRCOD43 EQU *
CLC WBEP_ERROR_CODE,=H'43' was it error code 43 ?
BNE ERRERR no, then it is unknown
MVC 0(L'WBERR43,R5),WBERR43 yes, write text
LA R5,L'WBERR43(,R5) adjust pointer
LA R6,L'WBERR43(,R6) adjust length
B ABENDCOD and test for an abend code

ERRERR EQU *
MVC 0(L'WBERRERR,R5),WBERRERR write not found in DFHWBUCD
LA R5,L'WBERRERR(,R5) adjust pointer
LA R6,L'WBERRERR(,R6) adjust length
MVC 0(17,R5),=C'WBEP_ABEND_CODE: ' move field name
LA R5,17(,R5) adjust pointer
LA R6,17(,R6) adjust length
MVC 0(4,R5),WBEP_ABEND_CODE move the abend code itself
LA R5,4(,R5) adjust pointer
LA R6,4(,R6) adjust length

MESSAGE EQU *
L R7,WBEP_MESSAGE_LEN load message length
LTR R7,R7 is it zero ?
BZ CLADDR yes, look for server addr
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(14,R5),=C'WBEP_MESSAGE: ' move field name
LA R5,14(,R5) adjust pointer
LA R6,14(,R6) adjust length
L R8,WBEP_MESSAGE_PTR load pointer to message
BCTR R7,0 suppose <256; -1 for mvc
EX R7,MUDEVMSG and move the message text
A R5,WBEP_MESSAGE_LEN adjust pointer
A R6,WBEP_MESSAGE_LEN adjust length
B CLADDR see client address given

MUDEVMSG MVC 0(0,R5),0(R8)

CLADDR EQU *
IC R7,WBEP_CLIENT_ADDRESS_LEN load client addr length
LTR R7,R7 is it zero ?
BZ SADDR yes, look for server addr
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(21,R5),=C'WBEP_CLIENT_ADDRESS: ' move field name
LA R5,21(,R5) adjust pointer
LA R6,21(,R6) adjust length
L R8,WBEP_CLIENT_ADDRESS load pointer to client addr
BCTR R7,0 -1 for mvc
EX R7,MUDEVMSG move client address
LA R7,1(,R7) +1 to have correct length
AR R5,R7 adjust pointer
AR R6,R7 adjust length

SADDR EQU *
IC R7,WBEP_SERVER_ADDRESS_LEN load server addr length
LTR R7,R7 is it zero ?
BZ TCPIPSER yes, look for TCPIPIService
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(21,R5),=C'WBEP_SERVER_ADDRESS: ' move field name
LA R5,21(,R5) adjust pointer
LA R6,21(,R6) adjust length
LA R8,WBEP_SERVER_ADDRESS load pointer to server addr
BCTR R7,0 -1 for mvc
EX R7,MUDEVMSG move server address
Appendix A. Listings

105

LA R7,1(,R7) +1 to have correct length
AR R5,R7 adjust pointer
AR R6,R7 adjust length

TCPIPSER EQU *

CLI WBBEP_TCPIPIService_Name,X'00' test if present on 1st byte
BE CONVERT test converter
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(24,R5),-'WBBEP_TCPIPIService_Name:' move field name
LA R5,24(,R5) adjust pointer
LA R6,24(,R6) adjust length
MVC 0(8,R5),WBBEP_TCPIPIService_Name move 8 bytes name
LA R5,8(,R5) adjust pointer
LA R6,8(,R6) adjust length

CONVERTR EQU *

CLI WBBEP_CONVERTER_PROGRAM,X'00' test if present on 1st byte
BE TARGET not present, test program
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(24,R5),-'WBBEP_CONVERTER_PROGRAM:' move field name
LA R5,24(,R5) adjust pointer
LA R6,24(,R6) adjust length
MVC 0(8,R5),WBBEP_CONVERTER_PROGRAM move 8 bytes converter
LA R5,8(,R5) adjust pointer
LA R6,8(,R6) adjust length

TARGET EQU *

CLI WBBEP_TARGET_PROGRAM,X'00' test if present on 1st byte
BE Failing not present, test failing
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(21,R5),-'WBBEP_TARGET_PROGRAM:' move field name
LA R5,21(,R5) adjust pointer
LA R6,21(,R6) adjust length
MVC 0(8,R5),WBBEP_TARGET_PROGRAM move 8 bytes program name
LA R5,8(,R5) adjust pointer
LA R6,8(,R6) adjust length

FAILING EQU *

CLI WBBEP_FAILING_PROGRAM,X'00' test if present on 1st byte
BE ANALRESP not present, test analyzer
MVC 0(3,R5),NL write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(22,R5),-'WBBEP_FAILING_PROGRAM:' move field name
LA R5,22(,R5) adjust pointer
LA R6,22(,R6) adjust length
MVC 0(8,R5),WBBEP_FAILING_PROGRAM move 8 bytes program name
LA R5,8(,R5) adjust pointer
LA R6,8(,R6) adjust length

* We do not format WBBEP_HTTP_RESPONSE_CODE as it is the same as
* the one we have in the response code text that is passed in the
* COMMAREA. This response code will be on the top of our screen.

ANALRESP EQU *

CLC WBBEP_ANALYZER_RESPONSE,='F'0=' analyzer response code ?
BE CONVERST no, test for converter code
MVC 0(3,R5),NL yes, write on a new line
LA R5,3(,R5) adjust pointer
LA R6,3(,R6) adjust length
MVC 0(24,R5),-'WBBEP_ANALYZER_RESPONSE:' write field name
LA R5,24(,R5) adjust pointer
LA R6,24(,R6) adjust length
CLC WBBEP_ANALYZER_RESPONSE,='F'4=' is it code 4 ?
BNE ANALR8 no, see if it is 8
MVC 0(L'RESPE,R5),RESPE yes, write EXCEPTION
LA R5,L'RESPE(,R5) adjust pointer
LA R6,L'RESPE(,R6) adjust length
B ANALREAS go for the reason code

ANALR8 EQU *

CLC WBBEP_ANALYZER_RESPONSE,='F'8=' is it code 8 ?
BNE ANALR12 no, see if it is 12
MVC 0(L'RESPI,R5),RESPI yes, write INVALID
LA R5,L'RESPI(,R5) adjust pointer
LA R6,L'RESPI(,R6) adjust length
B ANALREAS go for the reason code

Appendix A. Listings 105
ANALR12 EQU *  
CLC WBEP_ANALYZER_RESPONSE, =F'12' is it code 12 ?  
BNE ANALUNKN no, write it is unknown  
MVC 0(L'RESPD,R5),RESPD yes, write DISASTER  
LA R5,L'RESPD(,R5) adjust pointer  
LA R6,L'RESPD(,R6) adjust length  
B ANALREAS go for the reason code  
ANALUNKN EQU *  
MVC 0(L'UNKNOWN,R5),UNKNOWN write anal resp cod unknown  
LA R5,L'UNKNOWN(,R5) adjust pointer  
LA R6,L'UNKNOWN(,R6) adjust length  
L R7,WBEP_ANALYZER_RESPONSE load response value  
CVD R7,DWORD convert to decimal  
OI DWORD+7,X'0F' make it printable  
UNPK 0(4,R5),DWORD+4(4) unpack to fullword  
LA R5,4(R5) adjust pointer  
LA R6,4(R6) adjust length  
ANALREAS EQU *  
MVC 0(3,R5),NL write on a new line  
LA R5,3(R5) adjust pointer  
LA R6,3(R6) adjust length  
MVC 0(22,R5),=C'WBEP_ANALYZER_REASON: ' write field name  
LA R5,22(R5) adjust pointer  
LA R6,22(R6) adjust length  
L R7,WBEP_ANALYZER_REASON load reason value  
CVD R7,DWORD convert to decimal  
LA R5,4(R5) adjust pointer  
LA R6,4(R6) adjust length  
CONVRESP EQU *  
CLC WBEP_CONVERTER_RESPONSE, =F'0' converter response given ?  
BE ENDED no, suppose we're through  
MVC 0(3,R5),NL write on a new line  
LA R5,3(R5) adjust pointer  
LA R6,3(R6) adjust length  
MVC 0(25,R5),=C'WBEP_CONVERTER_REASON: ' move field name  
LA R5,25(R5) adjust pointer  
LA R6,25(R6) adjust length  
CLC WBEP_CONVERTER_RESPONSE, =F'4' test if value is 4  
BE ENDED no, see if it is 8  
MVC 0(L'RESPE,R5),RESPE yes, write EXCEPTION  
LA R5,L'RESPE(,R5) adjust pointer  
LA R6,L'RESPE(,R6) adjust length  
B CONVREAS go for converter reason cod  
CONVR8 EQU *  
CLC WBEP_CONVERTER_RESPONSE, =F'8' test if value is 8  
BNE CONVR12 no, see if it is 12  
MVC 0(L'RESPI,R5),RESPI yes, write INVALID  
LA R5,L'RESPI(,R5) adjust pointer  
LA R6,L'RESPI(,R6) adjust length  
B CONVREAS go for converter reason cod  
CONVR12 EQU *  
CLC WBEP_CONVERTER_RESPONSE, =F'12' test if value is 12  
BNE CONVUNKN no, go to write unknown  
MVC 0(L'RESPD,R5),RESPD yes, write DISASTER  
LA R5,L'RESPD(,R5) adjust pointer  
LA R6,L'RESPD(,R6) adjust length  
B CONVREAS go for converter reason cod  
CONVUNKN EQU *  
MVC 0(L'UNKNOWN,R5),UNKNOWN write that it is unknown  
LA R5,L'UNKNOWN(,R5) adjust pointer  
LA R6,L'UNKNOWN(,R6) adjust length  
L R7,WBEP_CONVERTER_RESPONSE load converter response value  
CVD R7,DWORD convert to decimal  
OI DWORD+7,X'0F' make it printable  
UNPK 0(4,R5),DWORD+4(4) unpack to fullword  
LA R5,4(R5) adjust pointer  
LA R6,4(R6) adjust length  
CONVREAS EQU *  
MVC 0(3,R5),NL write on a new line  
LA R5,3(R5) adjust pointer  
LA R6,3(R6) adjust length  
MVC 0(23,R5),=C'WBEP_CONVERTER_REASON: ' move the field name  
LA R5,23(R5) adjust pointer  
LA R6,23(R6) adjust length  
L R7,WBEP_CONVERTER_REASON load converter reason value  
CVD R7,DWORD convert to decimal
OI DWORD+7,X'0F'
make it printable
UNPK 0(4,R5),DWORD+4(4)
unpack to fullword
LA R5,4(R5)
adjust pointer
LA R6,4(R6)
adjust length
ENDED EQU *
MVC 0(7,R5),-C'</BODY>'
move the ending body tag
LA R6,7(R6)
adjust length
ST R6,WEBEP_RESPONSE_LEN
and store it for CICS
*
EXEC CICS RETURN ,
return to DFHWBBLI
*
WEBEP EQU *
EXEC CICS DUMP TRANSACTION DUMP CODE('WEBEP') NOW Handle
*
MYHEAD EQU *
MYHEADL EQU *-MYHEAD
WBERR1 DC C'bloio_greater_than_32k_response' equ 1
WBERR2 DC C'commarea_no_content' equ 2
WBERR3 DC C'dfhwbbli_document_not_found' equ 3
WBERR4 DC C'dfhwbbli_codepage_not_found' equ 4
WBERR5 DC C'dfhwbbli_api_error' equ 5
WBERR6 DC C'dfhwbbli_link_failed_termerr' equ 6
WBERR7 DC C'dfhwbbli_link_failed_invreq' equ 7
WBERR8 DC C'dfhwbbli_link_failed_lengerr' equ 8
WBERR9 DC C'dfhwbbli_link_failed_pgmiderr' equ 9
WBERR10 DC C'dfhwbbli_link_failed_sysiderr' equ 10
WBERR11 DC C'dfhwbbli_link_failed_rolledback' equ 11
WBERR12 DC C'dfhwbbli_link_failed_notauth' equ 12
WBERR13 DC C'dfhwbbli_link_failed' equ 13
WBERR14 DC C'invalid_decode_parameter_list' equ 14
WBERR15 DC C'decode_error' equ 15
WBERR16 DC C'invalid_encode_parameter_list' equ 16
WBERR17 DC C'encode_error' equ 17
WBERR18 DC C'save_certificate_failed' equ 18
WBERR19 DC C'dfhwbbli_abend_handler_invoked' equ 19
WBERR20 DC C'invalid_attach' equ 20
WBERR21 DC C'receive_error' equ 21
WBERR22 DC C'analyze_link_error' equ 22
WBERR23 DC C'dfhwbxnx_codepage_error' equ 23
WBERR24 DC C'no_analyzer_specified' equ 24
WBERR25 DC C'receive_storage_error' equ 25
WBERR26 DC C'header_length_error' equ 26
WBERR27 DC C'dfhwbxnx_logic_error' equ 27
WBERR28 DC C'link_dfhwbbli_failed' equ 28
WBERR29 DC C'analyze_error' equ 29
WBERR30 DC C'analyze_datalength_error' equ 30
WBERR31 DC C'not_authorized_to_start_alias' equ 31
WBERR32 DC C'dfhwbbli_bad_previous_web_send' equ 32
WBERR33 DC C'bad_commarea_response' equ 33
WBERR34 DC C'alias_task_purged' equ 34
WBERR35 DC C'security_unknown_response' equ 35
WBERR36 DC C'security_esm_not_responding' equ 36
WBERR37 DC C'security_application_notauth' equ 37
WBERR38 DC C'security_userid_revoked' equ 38
WBERR39 DC C'security_esm_not_responding' equ 39
WBERR41 DC C'security_invalid_userid' equ 41
WBERR42 DC C'attach_logic_error' equ 42
WBERR43 DC C'user_not_authorised' equ 43
WBERRERR DC C'WEBED_ERROR_CODE not found in DFHWBUCD list. Value is '
*
UNKNOWN DC C'unknown. Value is '
RESPE DC C'EXCEPTION'
RESPI DC C'INVALID'
RESPD DC C'DISASTER'
*
Appendix B. Special notices

This publication is intended to help customers who are implementing the CICS WebSupport and the 3270 bridge in an environment of CICS Transaction Server for VSE/ESA Version 1 Release 1.1. The information in this publication is not intended as the specification of any programming interfaces that are provided by CICS Transaction Server for VSE/ESA Version 1 Release 1.1. See the PUBLICATIONS section of the IBM Programming Announcement for CICS Transaction Server for VSE/ESA Version 1 Release 1.1 for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. 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 customer's 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 environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

IBM ©
Redbooks
Redbooks Logo
AS/400
AT

© Copyright IBM Corp. 2000
The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjøbenhavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

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

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

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.
Appendix C. Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

C.1 IBM Redbooks

For information on ordering these publications see “How to get IBM Redbooks” on page 113.

- VSE/ESA as a Web Server, SG24-2040
- CICS Transaction Server for OS/390 Version 1 Release 3: Web Support and 3270 Bridge, SG24-5480
- Migration to VSE/ESA 2.4 and CICS Transaction Server for VSE/ESA 1.1, SG24-5595
- Implementation of VSE/ESA 2.4 and CICS Transaction Server for VSE/ESA 1.1, SG24-5624
- e-business Solutions for VSE/ESA, SG24-5662
- e-business Connectivity for VSE/ESA, SG24-5950

C.2 IBM Redbooks collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at ibm.com/redbooks for information about all the CD-ROMs offered, updates and formats.

<table>
<thead>
<tr>
<th>CD-ROM Title</th>
<th>Collection Kit Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM System/390 Redbooks Collection</td>
<td>SK2T-2177</td>
</tr>
<tr>
<td>IBM Networking Redbooks Collection</td>
<td>SK2T-6022</td>
</tr>
<tr>
<td>IBM Transaction Processing and Data Management Redbooks Collection</td>
<td>SK2T-8038</td>
</tr>
<tr>
<td>IBM Lotus Redbooks Collection</td>
<td>SK2T-8039</td>
</tr>
<tr>
<td>Tivoli Redbooks Collection</td>
<td>SK2T-8044</td>
</tr>
<tr>
<td>IBM AS/400 Redbooks Collection</td>
<td>SK2T-8049</td>
</tr>
<tr>
<td>IBM Netfinity Hardware and Software Redbooks Collection</td>
<td>SK2T-8046</td>
</tr>
<tr>
<td>IBM RS/6000 Redbooks Collection</td>
<td>SK2T-8043</td>
</tr>
<tr>
<td>IBM Application Development Redbooks Collection</td>
<td>SK2T-8037</td>
</tr>
<tr>
<td>IBM Enterprise Storage and Systems Management Solutions</td>
<td>SK3T-3694</td>
</tr>
<tr>
<td>TCP/IP for VSE/ESA</td>
<td>SK2T-1336</td>
</tr>
</tbody>
</table>

C.3 Other resources

These publications are also relevant as further information sources:

- CICS Transaction Server for VSE/ESA Migration Guide Release 1, GC33-1646
- CICS Problem Determination Guide, GC33-1663
- CICS Transaction Server for VSE/ESA Enhancement Guide Release 1, GC34-5763

© Copyright IBM Corp. 2000
C.4 Referenced Web sites

These Web sites are also relevant as further information sources:

- Visit the VSE/ESA home page at: http://www.ibm.com/vse
- Visit the CICS home page at: http://www.software.ibm.com/ts/cics
- For an overview of the CICS transaction gateway, visit the Web site at: http://www.ibm.com/software/ts/cics/platforms/internet/tgw30
- For information on the MQSeries family, log on to: http://www.ibm.com/software/ts/mqseries
- For information on DB2 Connect, log on to: http://www.ibm.com/software/data/db2/db2connect
- For information on WebSphere Host On-Demand, visit the Web site at: http://www.ibm.com/software/network/hostondemand
- For information on CICS, log on to: http://www-4.ibm.com/software/ts/cics
- VSE/ESA performance information and results are available at the VSE/ESA Internet home page at: http://www.s390.ibm.com/vse
How to get IBM Redbooks

This section explains how both customers and IBM employees can find out about IBM Redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** [ibm.com/redbooks](http://ibm.com/redbooks)
  
  Search for, view, download, or order hardcopy/CD-ROM Redbooks from the Redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

  Redpieces are Redbooks in progress; not all Redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**
  
  Send orders by e-mail including information from the IBM Redbooks fax order form to:

  - In United States or Canada: [pubscan@us.ibm.com](mailto:pubscan@us.ibm.com)
  - Outside North America: 

    Contact information is in the “How to Order” section at this site: [http://www.elink.ibmlink.ibm.com/pbl/pbl](http://www.elink.ibmlink.ibm.com/pbl/pbl)

- **Telephone Orders**

  - United States (toll free): 1-800-879-2755
  - Canada (toll free): 1-800-IBM-4YOU

- **Fax Orders**

  - United States (toll free): 1-800-445-9269
  - Canada: 1-403-267-4455

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the Redbooks Web site.

---

**IBM Intranet for Employees**

IBM Redbooks fax order form

Please send me the following:

<table>
<thead>
<tr>
<th>Title</th>
<th>Order Number</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

First name
Last name

Company

Address

City
Postal code
Country

Telephone number
Telefax number
VAT number

☐ Invoice to customer number

☐ Credit card number

Credit card expiration date
Card issued to
Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.
### Index

#### Numerics
- 2 tier solution 3
- 3 tier solution 3
- 3270 bridge 9, 15, 43

#### A
- AMODE 28
- Analyzer program 15
- asynchronous access 4

#### B
- background 37
- BMS support 45
- body completion 37
- Bridge Exit program 9
- business logic 23

#### C
- cataloging graphics 40
- CEDX 68
- CETR Component Trace Options 67
- CICS 4
- CICS Transaction Gateway (CTG) 1
- CICS Universal Client 1
- CICS Web Support
  - advantages 4
  - characteristics 4
  - customizing 15
  - implementing 11
  - installation verification 14
  - limitations 5
- client code page 25
- COMMAREA 1, 23, 24, 27, 43
- connectors
  - DB2 2
  - MQSeries 2
  - VSE/ESA native 2
- Converter Program 16, 24
  - decode 16
  - encode 17
- CSOL 8, 9, 24
- CWBA 8, 28, 32
- CWBG 9
- CWBM 9
- CWS 23
  - data flow 23
- CWS on VSE/ESA 28
- CWXN 8, 24

#### D
- DB2 2
- debug problems 67
- debugging with CEDX 68
- Decode function 8, 24, 43
- DEMO 34
- DFHSSOT 12
- DFHSWB1A 15, 34
- DFHSWB1A 11
- DFHSWBSC 11
- DFHSWBSN 11
- DFHCNV 8, 24
- DFHCNV 13
- DFHSIT 12
- DFHWBADX 16
- DFHWBEP 17
- DFHWBLT 43
- DFHWBTTA 9, 44
- Distributed Program Link 1
- DOCTEMPLATE definition 13, 45
- DOCTEMPLATE 13
- dumps 67

#### E
- EIBCALEN 28
- Encode function 8, 24
- EXEC CICS
  - DELETEQ TS 25
  - DELETO 27
  - DOCUMENT CREATE 26
  - DOCUMENT INSERT 26
  - DOCUMENT RETRIEVE 27
  - DOCUMENT SET 27
  - EXTRACT TCP/IP 26
  - READQ 27
  - WEB ENDBROWSE FORMFIELD 26
  - WEB ENDBROWSE HTTPHEADER 26
  - WEB EXTRACT 26
  - WEB READ FORMFIELD 26
  - WEB READ HTTPHEADER 26
  - WEB READNEXT FORMFIELD 26
  - WEB READNEXT HTTPHEADER 26
  - WEB RECEIVE 25
  - WEB RETRIEVE 26
  - WEB SEND 25
  - WEB STARTBROWSE FORMFIELD 26
  - WEB STARTBROWSE HTTPHEADER 26
  - WEB WRITE 26
  - WRITEQ 27
- EXTRCT1 28
  - Catalog documents 28
  - Compile 28
  - DOCTEMPLATE definitions 28
- EXTRCTF1 28
- EXTRCTH1 28

#### F
- FORMFIELD 37
- FTP GIF or JPEG 41
IBM Redbooks review

Your feedback is valued by the Redbook authors. In particular we are interested in situations where a Redbook "made the difference" in a task or problem you encountered. Using one of the following methods, please review the Redbook, addressing value, subject matter, structure, depth and quality as appropriate.

- Use the online Contact us review redbook form found at [ibm.com/redbooks](http://ibm.com/redbooks)
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

<table>
<thead>
<tr>
<th>Document Number</th>
<th>Redbook Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG24-5997-00</td>
<td>CICS Transaction Server for VSE/ESA: CICS Web Support</td>
</tr>
</tbody>
</table>

What other subjects would you like to see IBM Redbooks address?

Please rate your overall satisfaction:

- O Very Good
- O Good
- O Average
- O Poor

Please identify yourself as belonging to one of the following groups:

- O Customer
- O Business Partner
- O Solution Developer
- O IBM, Lotus or Tivoli Employee
- O None of the above

Your email address: The data you provide here may be used to provide you with information from IBM or our business partners about our products, services or activities.

- O Please do not use the information collected here for future marketing or promotional contacts or other communications beyond the scope of this transaction.

Questions about IBM's privacy policy?: The following link explains how we protect your personal information.

[ibm.com/privacy/yourprivacy/](http://ibm.com/privacy/yourprivacy/)
CICS Transaction Server for VSE/ESA: CICS Web Support

CICS Web Support (CWS) is a new function in CICS Transaction Server for VSE/ESA 1.1.1. CWS first became available in OS/390 with CICS TS 1.2, and was enhanced in Release 1.3. This 1.3 functionality was ported to CICS TS for VSE/ESA 1.1.1.

CICS Web Support is an effective solution for the VSE/ESA user community. It is a powerful 2-tier Web enablement solution that is easy to plan for and simple to implement. Our goal for this IBM Redbook is to provide you with the information to use this great e-business connector. CWS unites browser technology with S/390. This provides tremendous flexibility to the end-user community while capitalizing on S/390 performance, reliability, scalability, availability, and data integrity.

This redbook discusses and positions the new CICS TS for VSE/ESA 1.1.1 CICS Web Support (CWS) and 3270 bridge. It provides a broad understanding of the new architecture, together with examples and samples to help customers in their planning and implementation of CWS. CWS employs a unique approach for using an e-business connector in the VSE/ESA environment.

This redbook also discusses planning for CWS, installation, and customization. It provides the guidance you need to design new solutions and upgrade existing solutions.