A method of providing Web access to data using dynamic generation of Web pages by a mainframe computer (16, 18) connected to a Web server (12, 14). The mainframe computer (16, 18) has legacy programs, legacy data, and legacy subroutines on it (legacy means existing information systems which may not be specifically adapted to Web access). A Web control program on the mainframe generates Web pages for the user. A state key is generated by the mainframe upon a user inputting identification data and is used to determine access allowed without the user reentering identification data.
<table>
<thead>
<tr>
<th>Code</th>
<th>Country</th>
<th>Code</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>Albania</td>
<td>ES</td>
<td>Spain</td>
</tr>
<tr>
<td>AM</td>
<td>Armenia</td>
<td>FI</td>
<td>Finland</td>
</tr>
<tr>
<td>AT</td>
<td>Austria</td>
<td>FR</td>
<td>France</td>
</tr>
<tr>
<td>AU</td>
<td>Australia</td>
<td>GA</td>
<td>Gabon</td>
</tr>
<tr>
<td>AZ</td>
<td>Azerbaijan</td>
<td>GB</td>
<td>United Kingdom</td>
</tr>
<tr>
<td>BA</td>
<td>Bosnia and Herzegovina</td>
<td>GE</td>
<td>Georgia</td>
</tr>
<tr>
<td>BB</td>
<td>Barbados</td>
<td>GH</td>
<td>Ghana</td>
</tr>
<tr>
<td>BE</td>
<td>Belgium</td>
<td>GN</td>
<td>Guinea</td>
</tr>
<tr>
<td>BF</td>
<td>Burkina Faso</td>
<td>GR</td>
<td>Greece</td>
</tr>
<tr>
<td>BG</td>
<td>Bulgaria</td>
<td>HU</td>
<td>Hungary</td>
</tr>
<tr>
<td>BJ</td>
<td>Benin</td>
<td>IE</td>
<td>Ireland</td>
</tr>
<tr>
<td>BR</td>
<td>Brazil</td>
<td>IL</td>
<td>Israel</td>
</tr>
<tr>
<td>BY</td>
<td>Belarus</td>
<td>IS</td>
<td>Iceland</td>
</tr>
<tr>
<td>CA</td>
<td>Canada</td>
<td>IT</td>
<td>Italy</td>
</tr>
<tr>
<td>CF</td>
<td>Central African Republic</td>
<td>JP</td>
<td>Japan</td>
</tr>
<tr>
<td>CG</td>
<td>Congo</td>
<td>KE</td>
<td>Kenya</td>
</tr>
<tr>
<td>CH</td>
<td>Switzerland</td>
<td>KG</td>
<td>Kyrgyzstan</td>
</tr>
<tr>
<td>CI</td>
<td>Côte d'Ivoire</td>
<td>KP</td>
<td>Democratic People's Republic of Korea</td>
</tr>
<tr>
<td>CM</td>
<td>Cameroon</td>
<td>KR</td>
<td>Republic of Korea</td>
</tr>
<tr>
<td>CN</td>
<td>China</td>
<td>KZ</td>
<td>Kazakhstan</td>
</tr>
<tr>
<td>CZ</td>
<td>Czech Republic</td>
<td>LC</td>
<td>Saint Lucia</td>
</tr>
<tr>
<td>DE</td>
<td>Germany</td>
<td>LI</td>
<td>Liechtenstein</td>
</tr>
<tr>
<td>DK</td>
<td>Denmark</td>
<td>LJ</td>
<td>Sri Lanka</td>
</tr>
<tr>
<td>EE</td>
<td>Estonia</td>
<td>LS</td>
<td>Lesotho</td>
</tr>
<tr>
<td>EU</td>
<td>European Union</td>
<td>LT</td>
<td>Lithuania</td>
</tr>
<tr>
<td>LU</td>
<td>Luxembourg</td>
<td>LV</td>
<td>Latvia</td>
</tr>
<tr>
<td>MC</td>
<td>Monaco</td>
<td>MD</td>
<td>Republic of Moldova</td>
</tr>
<tr>
<td>MG</td>
<td>Madagascar</td>
<td>MK</td>
<td>Former Yugoslav Republic of Macedonia</td>
</tr>
<tr>
<td>ML</td>
<td>Mali</td>
<td>MN</td>
<td>Mongolia</td>
</tr>
<tr>
<td>MR</td>
<td>Mauritania</td>
<td>MW</td>
<td>Malawi</td>
</tr>
<tr>
<td>MX</td>
<td>Mexico</td>
<td>NE</td>
<td>Niger</td>
</tr>
<tr>
<td>NL</td>
<td>Netherlands</td>
<td>NO</td>
<td>Norway</td>
</tr>
<tr>
<td>NZ</td>
<td>New Zealand</td>
<td>PL</td>
<td>Poland</td>
</tr>
<tr>
<td>PT</td>
<td>Portugal</td>
<td>RO</td>
<td>Romania</td>
</tr>
<tr>
<td>RU</td>
<td>Russian Federation</td>
<td>SD</td>
<td>Sudan</td>
</tr>
<tr>
<td>SE</td>
<td>Sweden</td>
<td>SG</td>
<td>Singapore</td>
</tr>
<tr>
<td>SI</td>
<td>Slovenia</td>
<td>SK</td>
<td>Slovakia</td>
</tr>
<tr>
<td>SN</td>
<td>Senegal</td>
<td>SZ</td>
<td>Swaziland</td>
</tr>
<tr>
<td>TD</td>
<td>Chad</td>
<td>TG</td>
<td>Togo</td>
</tr>
<tr>
<td>TJ</td>
<td>Tajikistan</td>
<td>TM</td>
<td>Turkmenistan</td>
</tr>
<tr>
<td>TR</td>
<td>Turkey</td>
<td>TT</td>
<td>Trinidad and Tobago</td>
</tr>
<tr>
<td>UA</td>
<td>Ukraine</td>
<td>US</td>
<td>United States of America</td>
</tr>
<tr>
<td>VG</td>
<td>United States of America</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UZ</td>
<td>Uzbekistan</td>
<td>VN</td>
<td>Vietnam</td>
</tr>
<tr>
<td>YU</td>
<td>Yugoslavia</td>
<td>ZW</td>
<td>Zimbabwe</td>
</tr>
</tbody>
</table>

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.
WEB PAGE ACCESSING OF DATA BASES AND MAINFRAMES

Copyright

A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by any one of the patent disclosure, as it appears in the Patent and Trademark office patent files or records, but otherwise reserves all copyright rights whatsoever.

Background of the Invention

The present invention relates to distribution of information over a computer network using hypertext links and Web pages. More specifically, the present invention relates to information distribution by interaction of a Web page on a client, a Web server connected to the client via the Internet, and programs, such as a database program, on a mainframe computer.

The explosive growth of the Internet over the last several years has dramatically improved the ability to quickly distribute vast amounts of data to any interested person. Additionally, the use of cookies, Secure Sockets Layers (SSL), and numerous other techniques have allowed the Internet to provide distribution of information restricted to an authorized person only.

An example may best illustrate some considerations in the situation. A book store may have a Web server that allows customers and potential customers access to information. The information might be "unrestricted
access" information such as a list of books that the book store is willing to sell. The information might be "customer specific" information, such as the list of books currently under order by a customer, the customer's account balance, or other such information that should be accessible only to the given customer (apart from appropriate employees of the book store who may need access to such restricted information). Moreover, there might be information available to a class of customers, but which should not be accessible by all persons. Thus, a list of books at special discounted prices could be available to any member of a preferred buyer group, but not to others.

Regardless of whether information is unrestricted, restricted to a single user, or has some intermediate level of restriction, distribution of the information via the Web requires that a Web server handle the information. Several methods exist to enable the Web server to handle the information.

The most direct technique is for the book store's programmers or other employees to put the information on the Web server. Thus, a book store setting up a Web site can simply take its list of books from whatever computer it is currently stored on and reformat the list as needed. However, when the list is updated at its current storage location (such as a mainframe computer), the book store's employees would also have to update the list on the Web server. The disadvantages of requiring the same information to be updated at multiple storage locations
(such as in the mainframe computer and in the Web server), the possibility of data updates being made inconsistently, and the increase in storage demands required when the data must be stored in one location (such as a mainframe) for internal company use and in another location (such as a Web server) for access via the internet or other outside access.

Another technique for providing Web access to data is for the book store’s programmers to write specific interface programs to translate data stored in a database on the book store’s mainframe computer. For example, an interface program can convert database data into hypertext markup language (HTML) such that a Web browser may display the data as a Web page. However, generating such interface programs requires skills that may be outside the experience of the employees who manage the mainframe and its databases. Further, such interface programs may be required for different types of data access. If a list of available books is stored in one database with a given format, and a list of customers is stored in a different format in a different database, a different interface program may be required for each. Considering also that a list of book publishers may be stored in yet another format and that the book store might allow each book publisher to view its own account information, yet another interface program would have to be developed. Developing interface programs for each database or each data format is expensive and time-consuming.
In addition to problems associated with allowing Web access to data on mainframe databases, security considerations often complicate access to information. For example, if the book store wanted to allow customers access to details about the status of their book orders or their account, the Web server often places a so-called cookie on the customer’s computer. The cookie is a small file of data or code identifying the customer’s computer to the Web server. However, many computer users do not want their computers accepting cookies from Web servers and they therefore limit the use of this. A further problem with some Web access security arrangements is that some are too easily overcome. For example, a person may "spoof" another’s identity (impersonate another) in order to obtain restricted data.

Yet another consideration in distributing large amounts of data via the Internet is the need to be able to manage and administer a Web site. Often Web site management requires complex knowledge and considerable effort. Changing the appearance of a Web page (such as changing the presentation of data on a Web page) can be a time-consuming process which requires highly specialized expertise. (As used herein, changing the appearance means causing the Web page to appear differently to all who access it by causing it to display data from the computer storing the Web page, and shall not include simply appending user comments or designs to a Web page as is
Objects and Summary of the Invention

Accordingly, it is a primary object of the present invention to provide a new and improved technique of distributing data via the Internet or other computer network, bulletin board, etc.

A more specific object of the present invention is to provide for Internet distribution of data stored in a database without the need for storing the data on a Web server.

A further object of the present invention is to provide ready Web access to data stored in various formats without the need to develop different programs.

Yet another object of the present invention is to provide a new and improved security arrangement for limiting access to Web data to authorized users.

A further object of the present invention is to provide convenient management of a Web site without requiring advanced programming skills or expertise in HTML.

The above and other features of the present invention which will be more readily understood when the following detailed description is considered in conjunction with the accompanying drawings are realized by a method of providing Web access to data. A Web server for distribution of data to users is provided. A database is
provided on a database computer operably connected to the Web server, the database having a database program that provides data in a given format. Upon a user requesting data in the database, the Web server forwards the request to the database, the database program accesses the data in the given format. The database computer runs a Web control program that generates a Web page with the requested data and supplies the generated Web page to the user. The Web control program marries file definition objects and page definition objects to generate Web pages. The Web server is on a Web server computer different from the database computer. The database computer is a mainframe computer. The mainframe computer supplies the generated Web page to the user via the Web server computer. Upon a user requesting data not in the mainframe computer, the Web control program causes the mainframe computer to access data on a remote computer and the Web control program then generates a Web page with the requested data and supplies the generated Web page to the user. Responsive to a user, the Web control program calls a subroutine on the mainframe computer, which subroutine is independent of the Web control program and is a legacy subroutine. The method further includes the steps of: having a user supply an identification code, forwarding the identification code from the Web server computer to the mainframe computer, the Web control program then generating a state key, incorporating the state key into a Web page supplied to that user, and wherein the mainframe computer detects the
state key from the Web page upon the user requesting data and decides whether that user can access the requested data based on the state key. The state key is randomly generated for a given access session and the state key loses its ability to authorize data access if it is not sent by the user to the mainframe computer at least once during a time-out interval. The Web control program receives a page change command from a user using a Web browser and, responsive to the page change command, the Web control program changes a given Web page to a changed Web page such that any users accessing the given Web page now receive the changed Web page, the page change command changing the appearance of a given Web page.

The present invention may alternately be described as a method of providing Web access to data, the steps including: providing a Web server for distribution of data to users; providing a database on a database computer operably connected to the Web server, the database having a database program that provides data in a given format; providing a Web control program on the database computer; having a user supply an identification code to the Web server, the Web control program then generating a state key and incorporating the state key into a Web page supplied to that user, and wherein the database computer detects the state key from the Web page upon the user requesting data and decides whether that user can access the requested data based on the state key. The state key loses its ability to authorize data access if it is not
sent by the user to the database computer at least once during a time-out interval. The Web server is on a Web server computer different from the database computer. The method further includes the step of forwarding the identification code from the Web server to the database computer. The database computer is a mainframe computer. The state key is randomly generated for a given access session.

The present invention may alternately be described as a method of managing a Web site, the steps including: providing a Web control program on a computer; having the Web control program receive a page change command from a user using a Web browser to access the Web site; and, responsive to the page change command, the Web control program changing a given Web page to a changed Web page such that any users accessing the given Web page now receive the changed Web page, the page change command changing the appearance of a given Web page. The Web control program marries file definition objects and page definition objects to generate Web pages.

The computer having the Web control program is a mainframe computer and users request data on the mainframe computer via a Web server on a Web server computer. The Web control program generates HTML through a completely table driven process, independent of file definitions and page definitions. The method further includes the steps of: having a user supply an identification code, forwarding the identification code from the Web server computer to the
mainframe computer, the Web control program then generating a state key, incorporating the state key into a Web page supplied to that user, and wherein the mainframe computer detects the state key from the Web page upon the user requesting data and decides whether that user can access the requested data based on the state key.

The method further includes the steps of: providing a database on the mainframe computer, the database having a database program that provides data in a given format; upon a user requesting data in the database, the Web server forwarding the request to the database, the database program accessing the data in the given format, and, by operation of the Web control program, generating a Web page with the requested data and supplying the generated Web page to the user.

Brief Description of the Drawings

The above and other features of the present invention will be more readily understood when the following detailed description is considered in conjunction with the accompanying drawings wherein like characters represent like parts throughout the several views and in which:

FIG. 1 is a simplified diagram of the arrangement used for the present invention;

FIG. 2 is a simplified view of the appearance of a Web page according to the present invention;
FIG. 3 is a simplified view of the appearance of an identification Web page according to the present invention;

FIG. 4 is a simplified view of the appearance of a restricted Web page according to the present invention;

FIGS. 5A and 5B are parts of a simplified overall flowchart of the handling of a Web request for data;

FIG. 6 is a flowchart of function determination, showing more details of parts of the flowchart of FIGS. 5A and 5B;

FIG. 7 is a flowchart of driven Web page data access; and

FIG. 8 is a simplified diagram illustrating a recursive Web page change feature of the present invention.

Detailed Description

Turning now to FIG. 1, some very basic concepts of the present invention will be discussed. Users accessing the World Wide Web of the Internet use their computers 10 running a Web browser program to access a Web server 12. (For ease of illustration, connections intermediate to the computers 10 and Web server 12, such as computers of an Internet service provider, are not shown.) The Web server 12 may be connected to receive information from another Web server such as 14. Additionally, the Web server 12 is connected to a mainframe computer 16 that
has legacy programs, legacy data, and legacy subroutines on it. (Legacy means existing information systems which may not be specifically adapted to Web access).

Previously, and as discussed in the background above, providing access to data on the mainframe via the Web required one to select from a number of unattractive options. The data could be duplicated on the Web server with the associated expansion in storage demands and increase in complexity in updating data (the data would have to be changed at each location where it is stored). Alternately, specific programs could be developed to allow Web access to particular data in the mainframe.

Advantageously, the present invention uses a Web control program that can generate Web pages in response to requests by users at computers 10. The Web control program runs on the mainframe 16 and generates the HTML needed to display particular data as part of a Web page. The Web access program on computer 16 additionally may, in response to a request for data that is not stored on mainframe computer 16, send a request for the data to another mainframe 18, receive the requested data back, put it into Web format (i.e., mark it up into HTML), and supply the Web page to the user. The mainframe computer 16 can access data at other computers by EDI (electronic data interchange), over TCP/IP (transport control protocol/internet protocol) or other protocols.

The Web control program is part of a Web management system also including a program that runs on the Web
server 12, which Web server is preferably running on a separate computer from the database-holding mainframe computer 16. In the discussion that follows, the Web management system will be referred to by the name EAGLE (Enhanced Application Generation Language for the Enterprise). EAGLE consists of several different and distinct parts including:

- There is a C program that runs as a CGI (common gateway interface) program on a Web server to extract and format data from forms that are submitted from a Web browser.

- There is a routing mechanism and a management structure for invoking CICS programs that can (Note: CICS is an IBM trademark.) produce Web pages. (As used herein, CICS stands for customer information control system running on the mainframe.) These pages can be created in any one of the three methods described below. Pages that are generated may then be interactively enhanced with styles, buttons, files to be read and/or updated, subroutines to be called, and many other variables using CICS-created objects.

- There is a state engine that keeps track of Web users and is able to maintain a session for Web users as well as do timeouts and re-establish
sessions using CICS programs.

- There is a CICS-based HTML generator that allows interactive creation of Web pages that can manipulate mainframe databases by reading, browsing, and updating VSAM and DB2. This one module can manage any number of different files, including fixed-length and segmented VSAM files, as well as DB2 databases. This program uses CICS-created objects for HTML generation including (but not limited to) file definitions, page definitions, drop-down lists, and Web page text.

- There is a set of development tools that facilitate the writing of EAGLE subroutines by generating object modules from simple templates that at the same time allow very flexible customizing of the modules. These object modules have the ability to read, browse, and update mainframe databases including VSAM (a specific brand name of indexed file structure) and DB2. (a specific database).

- Programs can be written and executed in any CICS-supported language without using any of the EAGLE development tools.

- This product can send and receive TCP/IP requests from any TCP/IP enabled source including Internet e-mail.
• Except for the C program that resides on the Web server, all of the programs are written in CICS Command Level Assembler language and run on IBM mainframes. This allows a level of security and scalability not found in other multi-tiered products. This would be especially useful in providing legacy mainframe systems and databases with Web access.

• All of the CICS objects necessary for operating EAGLE are created and maintained via the Web using EAGLE itself or by using mainframe terminal sessions (i.e., IBM3270).

Before supplying some specific examples of the Web management system in operation, some highly advantageous aspects of EAGLE will be discussed.

The combination of a CICS-based state engine and a program management structure that allows dynamic invocation of CICS programs means that the Web pages managed by EAGLE are associated with CICS-based objects that can be created and maintained interactively via the Web. Any Web page which is a part of EAGLE may call any subroutine, specify any files, pass variable data to the called subroutine, invoke additional subroutines or pages by way of return codes to the routing mechanism (error processing, rerouting of a request, pages by way of
return codes to the routing mechanism (error processing, rerouting of a request, etc.) These pages may require a PIN or other user authentication, invoke a process external to CICS, use a variable style-sheet, and include button objects. The flexibility of this system provides other features too numerous to mention here.

The ability to generate subroutines from CICS-based page templates that can be attached to EAGLE easily and seamlessly is another advantageous feature of EAGLE.

The existence of one single CICS program that can marry file definition objects and page definition objects resulting in the creation of a Web page that allows reading, updating, adding, deleting and browsing of VSAM files is an extraordinarily advantageous feature of the present invention. The EAGLE interactive HTML generator is completely table-driven and is independent of the actual file and page definitions.

The EAGLE subroutines may be written in any CICS supported language.

Presently, and in contest to the present invention, the accessing of data on a mainframe from the Web is done using multi-tiered systems, most often, "screen-scraping". These systems routinely contain many different programming languages and require many different skill sets. These skill sets are often very different from the skill sets of the mainframe staff and require intense training and frequent use. Also, the multi-tiered systems require other computers to act as
proxies for the mainframe, which adds a significant overhead cost. EAGLE cuts through these issues by operating on the mainframe using the language skills already possessed by the mainframe staff. EAGLE runs on the mainframe so that no extra machines are required for accessing the mainframe from the Web. This reduces the maintenance costs dramatically and reduces the number of points of failure as well. This methodology also cuts out the use of extra software products that not only add overhead to any transaction but also increase the complexity of the application and increase the difficulty of debugging errors.

Turning now to FIG. 2, a specific example of a Web page that may be generated by the present invention is shown. In this and the following the Web pages will be discussed in the context of a university, but it will be readily understood that the invention has applicability to other contexts. FIG. 2 is a menu page where various menu items are hypertext, meaning that a computer user at a computer 10 (FIG. 1) can click on (put mouse cursor at and press enter or press left mouse button) and thereby select one of the menu items. For example, clicking on GRADES, will allow a university student to access his or her grades.

Before allowing access to the grades, the identification Web page of FIG. 3 requires the student to enter a student identification number and personal identification number. Upon the student supplying this
information, the Web server 12 (FIG. 1) passes it to mainframe 16. The mainframe 16 generates a state key and incorporates the state key into the next Web page (FIG. 4) displayed for the student. Specifically, the state key is randomly generated as a session key for this particular student. The state key is hidden on the Web page, meaning that a browser will not display it. However, when the student goes to another Web page restricted to that student (for example going from grades to transcripts), the manner of insertion of the state key on the Web page ensures that the browser program running on the computer 10 transmits the state key to the Web server 12. The Web server 12 in turn supplies the state key to the mainframe 16. Upon mainframe 16 confirming that the state key corresponds to the student in question, the transcript of the student can be accessed. Importantly, a part of EAGLE on the mainframe 16 disables a given state key if the state key is not received for a given time interval (timeout period). This reduces the chances that someone can readily access the private records of a student who steps away from his or her computer without remembering to close the program.

Other menu items of FIG. 2 may allow access to unrestricted information such as the course schedule for a coming semester. Such unrestricted information may be available without going through identification procedures or may require one to log in with a guest identification.
Turning now to the flowchart of FIGS. 5A and 5B (the latter being a continuation of FIG. 5A), the processing of a Web request for data will be discussed in detail. At block 20, a user at a computer 10 (refer momentarily back to FIG. 1) requests data from mainframe 16. The user's Web browser sends the request. At block 22, the Web server 12 formats the request such that mainframe 16 can respond. The mainframe 16 is running a CICS listener program such as supplied by IBM for use on its mainframes. At block 24, this passes control to CICS on the mainframe 16. Next, block 26 determines if the state key has been established and verified. If the particular data request requires a security, a screen such as FIG. 3 will prompt the user to enter information causing access authentication, and the association of access authentication to a state key. Next block 28 reads a primary function table to determine the functions required to supply the requested data.

The primary function table supplies the following:

- Program to execute, if applicable,
- Whether to LINK, XCTL, or START appropriate program,
- Files/tables called programs to be used,
- Page definition table to be used,
- Additional subroutines to call,
- Route code definitions,
- Additional data to be used with program,
- Set update flag (Y/N, meaning yes or no),
PIN/Password protected page (Y/N) (some actions such as dropping a course may require reentry of PIN to increase protection against someone taking over when an authorized user has left without closing the Web browser program), and call an external data source, if needed, supplies record key to table with all necessary information.

Block 30 then runs any program needed to respond to the request. If no program is called, block 32 simply formats a Web page and sends it back to the user via the Web server 12, which Web server may simply relay, without change, the Web page or may make minor insertions before supplying the Web page to the user. Block 32 puts together all the parts of the outgoing Web page including the page style, buttons, or other objects; the generated HTML, state key, and cache information. If XCTL or START is required, control goes to block 34 which then closes the connection.

If a LINK is made, a program on the mainframe is run Block 36 reads incoming data using page definition table. Further block 38 allows subroutines to determine whether to continue with this program at block 42 or at block 40 set a route code and return to the main routing routine. Data access uses one or more of the following methods: dynamic page definition and file definition tables, and fixed object module(s).
Block 44 updates files if appropriate and if data passes all edits. Block 46 formats data for display using one or a combination of the following methods: dynamic page definition and file definition tables, and fixed object module(s).

Block 48 reads route code. If none is set, control goes to block 50 such that HTML is sent to the Web server, which in turn sends a response to the Web browser on computer 10.

If the route code is set, block 52 interprets the route code, and via block 54, returns to main routing section at block 28.

With reference now to FIG. 6, more details of block 28 of FIG. 5A will be given. At block 56, a primary function table 58 is read.

(A primary function table is given as Appendix 1 attached; a primary Web page format is attached as Appendix 2; a Web page forms and styles is attached as Appendix 3; a data access, edits, and layout definition is attached as Appendix 4; and a Web page object is attached as Appendix 5. These give specific examples of particular features used in the preferred embodiment.)

Block 56 leads to block 60 which tests if the page being requested is a protected page (i.e., restricted access). If so, block 62 returns a function to block 56 to test if the user is authorized. If block 60 determines that the page is not protected, control goes to block 64 which determines if pre-function checks are
needed. If yes, block 64 leads to blocks 66 and 68 which perform the checks and determine if the checks were passed. If so, the data appears proper and control goes to the continue at 70. If not, an error is noted at block 72.

Turning now to FIG. 7, a flowchart for driven Web page data access starts with block 74 which determines whether the page will use an object module. If yes, control goes to block 76 that provides Web page data access defined by the object module and leads to continue block 78. If no, a table driven process of data access at block 80 uses a primary Web page format 82, Web page objects 84, Web page forms and styles 86 and data access, edits, and layout definitions 88. Components 82 to 84 may be as shown on appendices 2 to 5.

FIG. 8 is a simplified example of how the present invention allows one, with proper authorization, to change a Web page in a recursive manner. The Web page 92 is displayed at a user's computer connected to Web server 12 and mainframe computer 16. The user uses the web browser program on his or her computer to send a page change command to server 12 and mainframe 16. This page change command, causes the generation of a revised version of the original Web page, this being changed Web page 94. The revision may include different formatting, style, data, etc., but will replace the old Web page 92. The page change command, as used herein, changes the appearance of the original Web page, meaning that it
causes the Web page to appear differently to all who access it by causing it to display data from the computer storing the Web page. As used herein, "changes the appearance" or "changing the appearance" shall not include simply appending user comments or designs to a Web page as is common for Web-based bulletin board or public comment boards. By allowing recursive control (controlling a given Web page from that Web page being on a person's browser, not on the person's Web page editor program) of Web pages, the management of a large Web site can be quite efficiently performed.

Attached as appendices A, B, C, D, E, F, G, and H are copyrighted programs which run on a mainframe computer in a specific implementation of the invention. Attached as appendix J is a copyrighted program in C called Nirvana and which runs on the web server computer in a specific implementation of the invention. Certain non-critical information such as ports that are used on the actual implementation have been changed or left out to reduce exposure to unauthorized persons overcoming security measures in a specific implementation of the technique.

Although specific constructions have been presented herein, it is to be understood that these are for illustrative purposes only. Various modifications and adaptations will be apparent to those of skill in the art. In view of possible modifications, it will be
appreciated that the scope of the present invention should be determined by reference to the claims appended hereto.
## Appendix I -- PRIMARY FUNCTION TABLE

<table>
<thead>
<tr>
<th>Topic</th>
<th>Description</th>
<th>MIDAS MENU SYSTEM</th>
<th>MMS 10/19/98 14:01:16 $BO3</th>
<th>DI6V</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>MI-XSCRIPT</td>
<td>Topic</td>
<td>COMADATA:</td>
<td>Auto?:</td>
<td>XCTL</td>
<td>To RAMIOROOG</td>
</tr>
<tr>
<td>1 13</td>
<td>14</td>
<td>COMAMORE:</td>
<td>Auto?:</td>
<td>Link</td>
<td>To</td>
</tr>
<tr>
<td>3 15</td>
<td>16</td>
<td>COMAMOR2:</td>
<td>Start:</td>
<td>GATA</td>
<td>MSYS</td>
</tr>
<tr>
<td>5 17</td>
<td>18</td>
<td>Not Auth:</td>
<td>Update?:</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>6 19</td>
<td>20</td>
<td>Not Open:</td>
<td>Student:</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>7 21</td>
<td>22</td>
<td>Invalid</td>
<td>Targ/Hdme:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 23</td>
<td>24</td>
<td>Return</td>
<td>Not Available</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 25</td>
<td>26</td>
<td>Process</td>
<td>Phase(s) T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 27</td>
<td>28</td>
<td>Choice</td>
<td>Check</td>
<td>Holds Y Owner RA</td>
<td></td>
</tr>
<tr>
<td>11 29</td>
<td>30</td>
<td>Offuse</td>
<td>Check</td>
<td>PIN Y MI-PIN</td>
<td></td>
</tr>
<tr>
<td>12 31</td>
<td>32</td>
<td>Xct to</td>
<td>Control</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Subroutines

<table>
<thead>
<tr>
<th>File</th>
<th>Chain</th>
<th>Subroutines</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>CRDBI2</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>File 2</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>File 3</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>File 4</td>
</tr>
<tr>
<td>4-0</td>
<td>1 Sess-2</td>
<td>128.227.75.2</td>
</tr>
</tbody>
</table>

## Appendix II -- PRIMARY WEB PAGE FORMAT

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Primary Web Page</title>
  </head>
  <body>
    <!-- HTML code for primary web page -->
  </body>
</html>
```

## Appendix III -- WEB PAGE FORMS & STYLES

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Web Page Forms & Styles</title>
  </head>
  <body>
    <!-- HTML code for web page forms & styles -->
  </body>
</html>
```
**Appendix IV -- DATA ACCESS, EDITS AND LAYOUT DEFINITIONS**

```
| NERCTST5 | MIDAS MODULE GENERATOR | MID | 10/19/98 14:32:21 | SBO3
| SELECT   |                       |     |                 | 008 001 17F
| Program  | Description | APPID | DelFlId | Lns: 10 | Max Segs: 146 |
| RAM10V00 | MIDAS TEXT   | MTEX | LINE     | Qlins: 200 | HedLen: 0449 |
| Upd/Lst: U/C/M (CMB) | Fixed? | Log?: | LinLen: 100 | SC-Offset: 0447 |
| Typ (BSF): SC Typ | Used? | Inseg?: | Limit: 100 | SegLen: 0079 |
| Blank Segs? | Align: | Ed | Key H | o Base | Edits y 0 | B D T Flad |
| Seq FLD | U M Out Ln Off | S H S I | Key NR### | p Ln GR L | P B Off | Header |
| 005 INPT | Y Y | 10 000 H 1 | 10 | 00 0 | S 0001 | MIDAS Text |
| 010 DESC | Y Y | 40 H | 40 | 00 0 | D 0011 | Description |
| 015 CTRL | Y Y | 10 H | 10 | 00 0 | D 0051 | Control |
| 020 ---- | Y | 01 H | 01 | 00 0 | E |
| 025 TITL | Y Y | 30 H | 30 | 00 0 | S 0247 | Title |
| 030 ---- | Y | 01 H | 01 | 00 0 | E |
| 035 LINE | Y Y | 79 S | 79 | 00 0 | S 0000 = |
| 040 ---- | Y | 01 S | 01 | 00 0 | E |
```

**Appendix V -- WEB PAGE OBJECTS**

```
| NERCTST5 | GATA TEXT | TXT | 10/19/98 15:24:37 | SBO3 |
| SELECT   | GATA TEXT KEY |     |                 | KN |
| GEN-ADD-7 |                        |     |                 |    |

TEXT

7. SECTION ### is a DUPLICATE of SECTION @@@@@
You must drop @@@@@ before adding another section of this same course.
CALL SUBROUTINE SCHED
```

SUBSTITUTE SHEET (RULE 26)
Appendix A

TSO FOREGROUND HARD
DSNAME RA-PATENT (RAMIOOOO)

* RAMIOOOO TITLE 'READ FROM CICS TCPIP AND WRITE RECORDS TO A TD QUEUE'
* ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT
* MECHANISM AND PROTOCOLS, ARE COPYRIGHT 1998 BY THE
* UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED.

SUBROUTINE RAMIOOOO

------------

FUNCTION:
READ A TCPIP SOCKET STREAM ORIGINATED FROM A CLIENT
DETERMINE PROPER FUNCTION TO PERFORM
LINK OR XCTL TO APPROPRIATE PROGRAM TO PERFORM THAT FUNCTION
RESPOND TO CLIENT WITH THE RESULTS

COPY RARAM1S
COPY RAMICOMM
COPY RARAMDAS
COPY RARAMMEN
USING RARAMMEN,R2

COPY RARATCPB
USING RARATCPB,R12

RAQUELEN EQU 32000
DSECT
RAQUEUE DS CL(RAQUELEN)
COPY REGISTER DEFINITION
COPY RARAREGS

DYNAMIC STORAGE
DFHEISTG
COPY RARASPAD

MIDASTE DS CL8
PEEKFLAG DS F
TEMPMVAR DS F
TEMPMVLN DS F
OPTVAL DS OD
OPTVONOF DS F
OPTVLEN DS F
OPTLEN DS F
OPTNAME DS F
TIMEOUT DS OD
TIMEOUTS DS F
TIMEOUTM DS F
*SNDMASK DS D
*SNDMASK DS D
*SNDMASK DS D
*RETMASK DS D
*RETMASK DS D
*RETMASK DS D
RSNDMASK DS F
WSNDMASK DS F
ESNDMASK DS F
RRETMASK DS F
WRETMASK DS F
ERETMASK DS F
MAXSOC DS F

SUBSTITUTE SHEET (RULE 26)
<table>
<thead>
<tr>
<th>Variable</th>
<th>DS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLAGS</td>
<td>F</td>
<td>FLAGS FOR SOCKET CALLS</td>
</tr>
<tr>
<td>NAME</td>
<td>CL16</td>
<td>SOCKET ADDRESS</td>
</tr>
<tr>
<td>NFAMILY</td>
<td>H</td>
<td>TCP/IP ADDRESSING FAMILY</td>
</tr>
<tr>
<td>NPORT</td>
<td>H</td>
<td>PORT DESCRIPTION</td>
</tr>
<tr>
<td>NADDR</td>
<td>F</td>
<td>IP ADDRESS</td>
</tr>
<tr>
<td>NDZERO</td>
<td>XL8</td>
<td>RESERVED (MUST BE ZEROS)</td>
</tr>
<tr>
<td>CLNTHNOL</td>
<td>H</td>
<td>SOCKET DESCRIPTOR OF CLIENT MACHINE</td>
</tr>
<tr>
<td>BUF</td>
<td>CLIZO</td>
<td>BUFFER FOR SOCKET READ/WRITE</td>
</tr>
<tr>
<td>CL4TLBUF</td>
<td>CL120</td>
<td>CONTROL BUFFER FOR SOCKET READ</td>
</tr>
<tr>
<td>PHASE</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>PASSXCTL</td>
<td>CL8</td>
<td></td>
</tr>
<tr>
<td>SAVETOPC</td>
<td>CL10</td>
<td></td>
</tr>
<tr>
<td>DONETOPC</td>
<td>CL10</td>
<td></td>
</tr>
<tr>
<td>IDENT</td>
<td>OCL16</td>
<td></td>
</tr>
<tr>
<td>INITMVSA</td>
<td>CL8</td>
<td></td>
</tr>
<tr>
<td>INITADSN</td>
<td>CL8</td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td>CL10</td>
<td></td>
</tr>
<tr>
<td>SAVEFICE</td>
<td>CL6</td>
<td></td>
</tr>
<tr>
<td>TCPV4</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>TOTLOLEN</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>GOTONE</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>FACTSRQ</td>
<td>C</td>
<td>SAVE SIZE OF BUFFER FOR SOCKET READ</td>
</tr>
<tr>
<td>FACTWAIT</td>
<td>C</td>
<td>SAVE REG 14 FOR BAL</td>
</tr>
<tr>
<td>SVNBYTES</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>SAVER14R</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>MAXITEMS</td>
<td>EQU 90</td>
<td></td>
</tr>
<tr>
<td>ITEMLLEN</td>
<td>EQU 6</td>
<td></td>
</tr>
<tr>
<td>QUEADDRS</td>
<td>CL</td>
<td>((MAXITEMS) * (ITEMLLEN))</td>
</tr>
</tbody>
</table>

**EXEC CICS HANDLE CONDITION**

```
EXEC CICS HANDLE CONDITION DSIDERR(ERROR) x
ERROR(ERROR) TERMIDERR(ERROR) SYSIDERR(ERROR) x
ISCNVREQ(ERROR) INVRIQ(ERROR) IOERR(ERROR) x
DISABLED(ERROR) ILLOGIC(ERROR) LENGERR(ERROR) x
NOTAUTH(ERROR) NOSTG(ERROR) x
```

**EXEC CICS ASSIGN STARTCODE(CODE)**

```
EXEC CICS ASSIGN STARTCODE(CODE)

CLC CODE = 'C'SD'
BNE RETURN

MVC CONSTANT, CL8 'DFHEIST' STORAGE LABEL
MVC PGRMNAM., = CL8 'RAM10000' PROGRAM NAME
```

**RECEIVE INPUT FROM CICS TCP/IP LISTENER**

```
A000 EQU *
MVC TCPIPLEN = AL2(TCPINS)
EXEC CICS RETRIEVE INTO(TCPINPUT) LENGTH(TCPINPUT) LENGTH(TCPINLEN) x
NONHANDLE
```

*SUBSTITUTE SHEET (RULE 26)*
CLC  TCPINLEN.=AL2(TCPIN3)  COMPARE LENGTH
BNE  PROGERR      BAD COMMAREA

MVC  SAVENOPE,BLANKS

DOSTEST MVC  DATAAREA, DATAAREA

GET MAIN STORAGE FOR COMMAREA AND RAMDAS FILE

SR  R14,R14  CLEAR REG
LA  R14,COMAHDL  GET CO'S AREA LENGTH
AH  R14,'H500' CHANGE THIS ONE ONLY
STH R14,COMLTH
SR  R14-R:L4  CLEAR REG
LA  R14-MDASL  GET LENGTH FOR TOP OF MDAS
AH  R14,COMLTH  ADD TO GET TOTAL NEEDED
XC  RETCODE,RETCODE  CLEAR RETURN CODE
XC  BUF,BUF  CLEAR AREA
LA  R6,BUF  ADDRESS OF AREA
BAL R14,GETRESPA  GET MESSAGE

L  R14,RETCODE
BCTR R14,0
LTR R14,R14
BM  NODATA
EX  R14-BUFBLANK  COMPARE TO BLANKS
BH  ERROR

NODATA EQU  *
BAL  R14,SENDSEND
BAL  R14,READSKT

CLI  FACTSREQ,'C\Y'
BE  TESTAREA
CLC  DATAAREA(4),'C\TEST'
BNE  NOTTEST

TESTAREA EQU  *
MVC  DATAAREA,DATAAREA

CHECK TO SEE IF THIS IS A CALL FOR INFORMATION ONLY

NOTTEST EQU  *
STATEMG EQU  *
EXEC CICS LINK PROGRAM 'RAM 0300') COMMAREA(RARAMDAS)  x
LENGTH(CMLENGTH)
BE  CHKWAIT  YES, CHECK TO SEE CLIENT IS WAITING.
CLI  COMAMPROF,'C\P'  IS THIS A RESPONSE TO A REQUEST?
BE  FACTCLOS  YES, RESPOND & CLOSE

CLC  COMAMESS(6),'C\VERIFY'  BLANK
BNE  READ2ND

CLC  COMATOPL,COMATOPC
BE  CHECKEM

NORESOUR MVC  COMATOPC(10),'C\MI-NORESRC'
B  READ2ND

CLC  BUF(9),'C\CLOSE'  CLOSE?  COMMENTED OUT
* BNE SOCKERR NO, ERROR ERROR, OK

* MVC SOCFUNC,CL16CLOSE SOCKET FUNCTION = CLOSE
CALL EZASOKET,(SOCFUNC,CL16HNDL, x
ERRNO,RETCODE), x
VL.MF(E,PARM,LST)

* CLI FACTSRQ,CY' IS THIS AN INITIAL OUTSIDE REQUEST?
* BE READ2ND YES, PROCESS REQUEST

* MVC TEMPVMLEN,COMAMVLN SAVE WRITE LENGTH

PQUELOOP
* MVC COMAQADD,COMAMVAR SET WRITE FROM ADDRESS
* EQU *
* MVC COMAQLEN,TEMPVMLEN+2 SET WRITE LENGTH
* B PQUELOOP GO WRITE ANOTHER QUEUE

ENDQLOOP EQU *
EXEC CICS DEQ RESOURCE(COMAPQNM) LENGTH(8)

* BE MOVETARG YES, MOVE TARGET
* MVC SAVEFICE,COMAFICE SAVE FICE IN COMMAREA
* CLC COMAFC1,BLANKS IS THERE A FICE CODE?
NOQUE
* MVC COMAPQNM,COMAMQNM
* MVI COMAPQNM,C=''
* EXEC CICS LINK PROGRAM(COMAPROG) COMMAREA(COMMAREA) x
* LENGTH(COMACOML)
* B PROGLINK

READMEN EQU
MVC DONETOPC,COTIATOPC
EXEC CICS READ DATASET('RAMMEN') SET(RZ) LENGTH(HALFWORD) X
RDFLD(COMATOPC)
MVC COMAOFL1,MMEOFL1
MVC COMAMLDS,MMEMNAME
MVC COMAS003,MMEN003
MVC COMAS004,MMEN004
MVC COMAS005,MMEN005
MVC COMAS006,MMEN006
MVC COMAS007,MMEN007
MVC COMAS008,MMEN008
BR R4

VERIFYEQU *
CLI MMENPASS,CY'
BNE CKHOLDS
MVC COMAVERI,MMENPKX
B READ2ND

MVC COMAIBFN,BLANKS
MVC COMACOLR,BLANKS
EXEC CICS LINK PROGRAM(COMAPROG,) COMMAREA(COMMAREA) X
LENGTH(COMACOML)

SUBSTITUTE SHEET (RULE 26)
IF THIS IS PROCESSING A WAITING REQUEST, FACTWAIT=Y, FACTSREQ=Y
USE2 PACK DOUBLE, COMAIBFN(2)
DECR SP DOUBLE,*P'1"
CVB R1.DOUBLE
MH R1,'H','10'
AR R14,R1
CLC 0(10,R14), BLANKS

SEND SEND EQU
ST R14,SPADSV14
A010 EQU *
L R14,SPADSV14
BR R14
READSKT EQU
ST R14,SPADSV14

READ RECORD FROM TCP/IP SOCKET
CLC RETCDS,=F'0' ANYMORE BYTES
BE SOCKERR TCP/IP ERROR NO, GO CLOSE SOCKET
CHECK DIFFERENCE BETWEEN WHAT WAS EXPECTED AND WHAT WAS RECEIVED
L R14,NBYTES LOAD EXPECTED LENGTH
S R14,RETCODE GET DIFFERENCE
LTR R14,R14 TEST REGISTER

MOVENUM MVZ NUMCHECK(\-),WKLEN CHECK TO SEE IF NUMERIC
CHKNUM CLC NUMCHECK(\-),NUMZONE CHECK TO SEE IF NUMERIC
A21 EQU *
MVC NBYTES,=A(L,WKLEN) LENGTH OF AREA
L R14,NBYTES
BCTR R14,0
LTR R14,R14

PUTRQST MVI FACTSREQ,C'Y' MARK AS REQUEST AND
MVC SOCKFUNC,=CL16'RECV' SOCKET FUNCTION = READ
MVC PEEKFLAG,=F'0' SET TO PEAK AT DATA
LH R14,='H'10'
A025 EQU *
STH R14,RECCOL.. SAVE RECV COUNTER
CALL EZASOKET,(SOCCFLN,CLTHNDL PEEKFLAG,
X BNP SOCKERR IF NOT POSITIVE, ERROR
ST R1, NBYTES NUMBER OF BYTES NEEDED
B A025

WAIT FOR CLIENT TO RESPOND

DORECVL EQU *
XC RSNDDMASK,RSNDDMASK INIT FIELDS
XC WSNDDMASK,WSNDDMASK"
XR RRETMASK, RRETMASK, RRETMASK,
X ERNNPRETCODE),
X
L VL, MF=(E,PARMLIST)
C RL, =F'0' POSITIVE RETURN CODE
BL SOCKERR NO MUST BE AN ERROR
LH R14,RECCOUNT RESTORE LOOP COUNTER
BCT R14,A025 GO ASK FOR MORE
B SOCKERR LOOPED MAX TIMES, ERROR

SUBSTITUTE SHEET (RULE 26)
B020  EQU  *  
B    READTEXT

NOTOPEN  EQU  *  
CLC  SAVENOPE,BLANKS  
BH  PUTNOPE  
MVC  COMATOPC(10),=C'NOTOPEN'  
B  READTEXT

PUTNOPE  MVC  COMATOPC,SAVENOPE  
B  READTEXT

SOCKERR  EQU  *  
MVC  COMATOPC(10),=C'SOCKERR'  
B  READTEXT

NOTFNDT  EQU  *  
B  ERROR

NOTFND  EQU  *  
CLC  COMAREGN,BLANKS  
BH  PUTNOTF  
MVC  COMATOPC(10),=C'NOTFOUND'  
B  READTEXT

PUTNOTF  MVC  COMATOPC,COMAREGN  
B  READTEXT

PROGERR  EQU  *  
MVC  COMATOPC(10),=C'PROGERR'  
BE  ERROR2  NO, GO TO ERROR

READQ  EQU  *  
CLI  COMAPROF,C'S FACTS RESPONSE  
BNE  NOPROC

EXEC  CICS  LINK  PROGRAM('RAMIIS00')  COMMAREA(COMMAREA)  
LENGTH(COMACOML)
LA  R0,*  MOVE  NULLS  TO  AREA  
LA  R14,QUEADDRS  
SR  R1,R1  
L  15,=A(L'QUEADDRS)  
MVCL  R14,R0

XC  TOTLQLEN,TOTLQLEN
READQAGN  EQU  *  
EXEC  CICS  LINK  PROGRAM('RARASOKY')  COMMAREA(COMMAREA)  
X
LENGTH(COMACOML)
LH  R14,COMAQML  LOAD  CURRENT  ITEM  NUMBER
MVC  SOCFUNC,=C 'SELECT'  SOCKET  FUNCTION  =SELECT]
CALL  EZASOKET,(SOCFUNC,MAXsoc,TIMEOUT,
BE  GETBCK  NO, GO BACK
CALL  EZACICOS,(BUF,NBYTES),VL,MF=(E,PARMLIST)  TRAN 2 EBCDIC
GETBCK  EQU  RETURN  EQU  
EXEC  CICS  RETURN  EJECT  
TITLE  'ABENDS'  
COPY  RAMIGETP  
QUEUEERR  EQU  
MVC  ABCODE=2,=C'QU'  INITIAL  TSQQUE  READ  ERROR

SUBSTITUTE SHEET (RULE 26)
MVC BUF,MSGZO0B ERROR MESSAGE FOR CLIENT
MVC NBYTES,=A(LMSGZO0B) LENGTH OF ERROR MESSAGE
B ZOO ABEND
PROGERR EQU
mvc ABCODE+2,=CP'R' PROGRAM LOGIC ERROR
MVC BUF,MSGZO0L ERROR MESSAGE FOR CLIENT
MVC NBYTES,=A(LMSGZO0L) LENGTH OF ERROR MESSAGE
B ZOO ABEND
SOCKET RB EQU
MVC ABCODE+2,=CS0 TCP/IP SOCKET ERROR
MVC BUF MSGZO0S ERROR MESSAGE FOR CLIENT
MVC NBYTES,=A(LMSGZO0S) LENGTH OF ERROR MESSAGE
B ZOO ABEND
NOTFNDT EQU
MVC ABCODE+2,=CTX' TCP/IP SOCKET ERROR
MVC BUF MSGZO0T ERROR MESSAGE FOR CLIENT
MVC NBYTES,=A(LMSGZO0T) LENGTH OF ERROR MESSAGE
B ZOOO ABEND
NOTFND EQU
MVC ABCODE+2,=CNF' RECORD NOT FOUND
MVC BUF MSGZO0N ERROR MESSAGE FOR CLIENT
MVC NBYTES,=A(LMSGZO0N) LENGTH OF ERROR MESSAGE
B ZOOO ABEND
ZOOO EQU
B Z010
ST R14,ABR14 SAVE LAST RETURN ADDRESS
USING DFHEIBLK,R14 SET EIB DSECT BASE
L R14,DFHEIBP LOAD EIB ADDRESS
MVC ABRSP,EBRESP SAVE LAST RESPONSE CODE
DROP R14 DROP EIB DSECT BASE
MVC ABCODE,=A(PGRMNAM+4) SET UP ABEND CODE
EXEC CICS DUMP
DUMPCODE(ABC0DE)
PROGRAM STORAGE TASK TERMINAL
SPACE 2 SEND ERROR MESSAGE TO THE CLIENT MACHINE
Z010 EQU
MVC SOCFUNC,=CL16'SEND' SOCKET FUNCTION=SEND
MVC FLAGS,=F'O' CLEAR FLAGS VARIABLE
CALL EZACIC04,(BUF,NBYTES),VL,MF=(E,PARMLIST) TRANS. TO ASCII
SPACE
CALL EZASOKET,(SOCFUNC,CLNTHNDL,FLAGS,
NBYTES,BUF,ERRNO,RETCODE),
VL,MF=(E,PARMLIST)
SPACE 2 B EZACLOSE GO CLOSE TCP/IP SOCKET AND EXIT
SPACE 2 TITLE 'CONSTANTS AND LITERALS'
DS OF
PACKONE DC PL4'1' PACKED DECIMAL ONE
BLANK DC X'40'
BLANKS DC CL120'
ZEROS DC CL30'0'
CKTR TR COMATOPC(‘-‘),UPCASE TRANSLATE TO UPPER CASE
CASE UPANCE DC X'000102030405060708090A0B0C0D0E0F'
DC X'101112131415161718191A1B1C1D1E1F'
DC X'202122232425262728292A2B2C2D2E2F'
DC X'303132333435363738393A3B3C3D3E3F'
DC X'404142434445464748494A4B4C4D4E4F'

SUBSTITUTE SHEET (RULE 26)
CLUFX N DC CL2'rLD'TD QUEUE PREFIX
MAX_SECONDS DC F'00000000'
FIONBIO DC X'L4'8004A77E'
SO_REUSEADDR DC F'00000004'
SO_KEEPAHEAD DC F'00000008'
SO_BROADCAST DC F'00000032'
SO_LINGER DC F'00000128'
SO_SNDBUF DC F'000004097'
SO_ERROR DC F'000004103'
SO_TYPE DC F'000004104'
NUMZONE DC CL28'0000000000000000000000000000000000'
MAXWRK DC F'32787' MAXIMUM RECORD LENGTH
MAXLEN DC H'32717' MAXIMUM RECORD LENGTH
MAXTIMES DC H'10' MAXIMUM SENDS
FILEMSG DC C'OKTOSEND' MESSAGE TO START SENDING DATA
OKMSG DC C'GOTFILEOK' SUCCESSFUL MESSAGE
RECKOMSG DC CL9'GOTDATAOK' SUCCESSFUL RECORD MESSAGE
WAITMSG DC C'WAITING FOR RESOURCES ON MAINFRAME...езульт "WAIT MESSAGE
MSGZ00A DC C'ERROR: COMMAREA LENGTH ERROR'
MSGZ20DB DC C'ERROR: TSQUEUE READ ERROR'
MSGZ20DC DC C'ERROR: CFTR TSQUEUE READ ERROR'
MSGZ20DD DC C'ERROR: LDL115P (EBCDIC->ASCII) ERROR'
MSGZ20DE DC C'ERROR: CICS ERROR'
MSGZ2OF DC C'ERROR: ONLOG VSAM WRITE ERROR'
MSGZ20GH DC C'ERROR: TSQUEUE WRITE ERROR'
MSGZ20H0 DC C'ERROR: TRANSIENT DATA QUE ENQ PROBLEM'
MSGZ20LN DC C'ERROR: PROGRAM LOGI ERROR'
MSGZ20TN DC C'ERROR: MMEN ENTRY NOT FOUND
MSGZ20TO DC C'ERROR: YOUR SESSION HAS TIMED OUT'
MSGZ200 DC C'ERROR: MMEN FILE NOT OPEN'
MSGZ20P DC C'ERROR: BAD RECORD LENGTH'
MSGZ20S DC C'ERROR: TCP/IP SOCKET ERROR'
MSGEND EQU LTORQ
END RAMI0000

CLC RETCODE,F'0' \ ANYMORE BYTES
BE SOCKERR TCP/IP ERROR NO, GO CLOSE SOCKET

CHECK DIFFERENCE BETWEEN WHAT WAS EXPECTED AND WHAT WAS RECEIVED

L R14, NBYTES \ LOAD EXPECTED LENGTH
S R14, RETCODE \ GET DIFFERENCE
LTR R14, R14 \ TEST REGISTER
MOV NUMVEN MVZ NUMCHECK('=', WKLLEN CHECK TO SEE IF NUMERIC
CHKNUM CLC NUMCHECK('=', NUMZONE CHECK TO SEE IF NUMERIC

A021 EQU '*
MVC NBYTES = A(L'WKLEN) LENGTH OF AREA
L R14, NBYTES
BCXR R14, 0
LXR R14, R14

PUTQST MVI FACTSREQ, 'Y' MARK AS REQUEST AND

MVC SOCFUNC = CL16 'RECV' SOCKET FUNCTION = READ
MVC PEEKFLAG = 'F' '0' SET TO PEEK AT DATA
APPENDIX B

TSO FOREGROUND HARDCOPY
DSNAME=RA.PATENT (RAM10100)
DATA SET RAM10100
ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT MECHANISM AND
PROTOCOLS, ARE COPYRIGHT 1998 BY THE UNIVERSITY OF FLORIDA, ALL RIGHTS
RESERVED
SUBROUTIN RAM10100
FUNCTION:
ROAD, COMPRESS AN INCOMMING DATABLOCK.
ADD ADDITIONAL FORMATTING AS DEFINED BY MTEX RECORD
WRITE NECESSARY TS QUEUES
1. RAM10100 FILE RECORD SECT
    COPY RAMCOMM
    USING COMMAREA,RS
    COPY RARAMMEN
    USING RARAMMEN,R9
    COPY RARAMSTY
    USING RARAMTEX,R7
    USING MTEXS,R2
MAXAREA EQU 32767
DATAAREA DSECT
2. WORKING STORAGE
    COPY DFHAID
    SPACE 3270 AID CHARACTERS
    COPY RARAREGS
    EJECT
    DFHEIST DSECT
    SCRATCH PAD AREA
    DOUBLE DS D
    FWORD DS F
    SAVEQADD DS F
    SAVER12 DS F
    DATAQADD DS F
    DATAQLEN DS H
    KEEPQADD DS F
    KEEPQLEN DS H
    SAVEQLEN DS H
    HALFWORD DS H
    SAVEMAX DS H
    QREVRT DS C
    READTEXT DS C
    NEEDPRE DS C
    GOTNIRV DS C
    BEENHERE DS C
    REGID DS CL4
    SPADSKY DS CL30
    STYLTYP DS C
    BALWRITE DS C
    AFTHEAD DS C
    EJECT
PROGRAM RAM10100 STARTS HERE
RAM10100 DFHEIENT CODEREG=(3,8), DATAREG=(13)
RAM10100 AMODE 31
RAM10100 RMODE ANY
SPACE
CLC EIBCALEN,H’0’

SUBSTITUTE SHEET (RULE 26)
BE    RETURN
L    R5,DFHEICAP    GET COMMAREA
SPACE
EXEC CICS HANDLE CONDITION DSIDERR(ERROR)
   ERROR  (ERROR)  TERMIDERR(ERROR)  SYSIDERR(ERROR)
   ISCNVREQ(ERROR)  INVREQERROR  IOERR(ERROR)
   DISABLED(ERROR)  ILOGIC(ERROR)  LENGERR(ERROR)
   NOTAUTH(ERROR)
EXEC CICS HANDLE CONDITION NOTFND(NOTFND) NOTOPEN(NOTOPEN)
DEBUG DEBUG DEBUG DEBUG
CLC COMATOPC(10),=C'XXXXXXXXX'
BNE AFTDBG1
DEBUG1 B AFTDBG1
AFTDBG1 EQU
DEBUG DEBUG DEBUG DEBUG DEBUG
FIRST EQU
CLI COMAPROF,C'S'
BE AFTFIRST
MVC COMAQITE,=H'1'
MVC COMAQ2DO,=C'DELETE'   SET TO DELETE QUEUE
MVC COMAQKEY,COMAMQNM     SET TS QUEUE NAME
EXEC CICS LINK PROGRAM('RARASOKY') COMMAREA(COMMAREA)
   LENGTH(COMACOML)
AFTFIRST EQU
USING MTEX,R2
CLC MTEXLINE(5),=C'STYLE'
BE MTEXLOOP
CLC MTEXLINE(5),=C'STYLE'
BE MTEXLOOP
CLC MTEXLINE(4),=C'FORM'
BE MTEXLOOP
CLC MTEXLINE(5),=C'<BODY'
   /www.isis.ufl.edu/wfimages/logo2.gif"/&lt;/td&gt;&lt;/h3&gt;'  
MVC 95(50,R12),MTEXNAME
MVC 145(27,R12),=C'&lt;/h3&gt;&lt;/tr&gt;&lt;/table&gt;&lt;/hr&gt;'  
MVI 172(R12),X'25'
MVC 173(05,R12),=C'&lt;PRE&gt;
MVI 178(R12),X'25'
LA R12,179(R12)
MVI NEEDPRE,C'Y'
CLC MTEXLINE(4),=C'FORM'
BE MTEXFORM
CLC MTEXLINE(10),=C'############'
BNE NOTDATA
CLI COMAM01,C'Y'
BE AFTDATA
SR R15,R15
LH R1,DATAQLEN
LR R14,R12
LR R15,R1
MVCL R14,R0
AH R12,DATAQLEN
ST R12,SAYER12
MVC COMAQADD,r QADD
AFTDATA LA R2,MTEXLEN(R2)
BCT R6,MTEXLOOP
B AFTERTEX
NOTDATA EQU
CLC MTEXLINE(08),=C'MDASTRAN'
BNE NOTTRAN
CLC COMATOPC.BLANKS
BNH  BCTMTEX
MVC  0(79,R12),MDASTRAN
MVC  0(79,R12),CACHLINE
MVC  44(R12),COMMAQNM
MVI  79(R12),X'25'
B   BCTMTEX

NOTNIR  EQU
MVC  0(79,R12),MTEXTLINE
MVI  79(R12),X'25'

BCTMTEX  EQU
LA  R2,MTEXTSLN(R2)

STYLE  EQU
EXEC  CICS HANDLE CONDITION NOTFND(NOTSY)NOTOPEN(NOSTY)
DISABLE(NOSTY)
LA  R14.9
EX  R14.CKTR
B   TITOVER

TITBCTLA  R15.1(R15)
BCT  R1,TITLOOP

TITOVER  EQU
NOSUBS EQU
CLC  MSTYLINE(8),=C'MDASTRAN'
BNE  CKNIRV
CLC  COMATOPC,BLANKS
BNH  BCTMSTY
MVC  0(79,R12),MDASTRAN
MVC  0(79,R12),CACHLINE
MVC  44(R12),COMMAQNM
MVI  79(R12),X'25'
B   BCTMSTY

NOTNIRV EQU
CKBUTT EQU
CLC  MSTYLINE(7),=C'MBUTTON'
BNE  NOTMBUTT
MVC  0(41,R12),-C'gif'><input type="hidden" names="MDASTRAN"
MVC  41(R12),C value="XX--"/></form>'
MVC  49(10,R12),SPADSKEY
MVI  68(R12),X'15'
LA  R12,80(R12)
B   BCTMSTY

NOTMBUTT EQU
SR  R14,R15
STH  R14,COMAQLEN
BAL  R4,COMPRESS

AFTTEXT  EQU
L   R14,COMAQADD
AH  R14,COMAQLEN
MVC  COMAQADD,SAVEQADD

COMPRESS  EQU
THIS CODE REMOVES THE EXTRA BLANKS
LH  R15,COMAQLEN
LTR  R15,R15
BZ  NOAREA
BNE  ERROR
L   R14,COMAQADD

COMPLOOP  EQU
CLI  0(R14),X'00'
A   R0,=F'100
TEST15

NOTFND  EQU
CLI  BEENHERE,

NOTOPEN EQU
ERROR  EQU

SUBSTITUTE SHEET (RULE 26)
APPENDIX C

TSO FOREGROUND HARDCOPY
DSNAME=RA.PATENT
PROGRAM RAM1B00
           (RAM1B00)
           0010000

ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT
MECHANISM AND PROTOCOLS, ARE COPYRIGHT 1998 BY THE
UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED.
SUBROUTINE RAM1B00
FUNCTION:
RETRIEVE SPECIFIED DATA FROM TCP/IP REQUEST
DO NECESSARY DATA FORMATTING
RETURN RESULTS TO CALLING PROGRAM

SASS MENU INTERFACE

COPY RACL0001
COPY RARAREGS

COMMAREA STARTS HERE BASE REG IS R5
COPY RAMICOMM    COPY COMMAREA DSCET
USING COMMAREA,R5

FILE DESECT STARTS HERE  BASE REG IS R9

QLINES    EQU 100
QLINELEN   EQU 140
TSL        EQU 5

USING COMMAREA,R5

WORKING STORAGE

DFHEISTG DSECT    COMMAREA DSECT

SCRATCH PAD AREA

FINDFLDT   DS CL256
DOUBLE3    DS D
           DS OD
           DS F
ANSWER     DC F'0'
DIVIDEND   DS OD

SUBSTITUTE SHEET (RULE 26)
QUOTIENT DS F
REMAINS DS F
DIVISOR DS F
MESS DS CL79
THISSEG DS PL3
CHAR9 DS CL9
FLDLEN DS F
SV01 DS F
SV014 DS F
SV015 DS F
SAVE6 DS F
DEC01 DS F
DEC14 DS F
DEC15 DS F
SIGDIGDS F
FOUNDIT DS C
GOTDEC DC C
GOTSIG DS C

COPY RARASPAD

PROGRAM RAM1B00 STARTS HERE
DFHEIENT CODEREG=(3,8,7),DATARAE=(13)
AMODE 31
RMODE ANY

CLC EIBCALEN.=H'0'
BE RETURN

EXEC CICS HANDLE CONDITION ERROR(0ERROR)
L R5,DFHEICAP

GETDATA EQU
BE BCTLOOP
CP COMAUSEG.,=P'0'
BE BCTLOOP
CLC 14(R6),=C'MDASSEGNT'
BNE BCTLOOP

PACK THISSEG,22(3R6)
CP COMAUSEG,THISSEG
BNE NOTSEG1

MVI FOUNDIT,C'Y'
B BCTLOOP NOT THERE, BUMP A BYTE

NOTSEG1 MVI FOUNDIT,C'N'
B BCTLOOP NOT THERE, BUMP A BYTE

SAVR6 ST R6,SAVE6

CLI FOUNDIT,C'Y'
BNE BCTLOOP
CLI COMADEC,P,=C'0'
BNH QUEMATCH ANY DECIMAL PLACES?
NO, SKIP

ST R1,DEC01 SAVE
ST R14,DEC14 REGS
ST R15,DEC15

PACK DOUBLE2,COMADECPUT DEC PLACES IN DOUBLE?
CLC COMAOLEN,NUMZONE LENGTH
BE DECEND OK?
MVZ NUMCHECK(2),COMAOLEN
CLC NUMCHECK(2),NUMZONE
BNE DECEND NO, DONE
PACK DOUBLE,COMAOLEN PUT LENGTH IN DOUBLE

CP
DOUBLE,DOUBLE2 DEC PLACES BIGGER THAN LENGTH
BH LOK
MVI COMASERR,C'Y'
B SAEYERROR ERROR

LOK
ZAP SIGDIG,DOUBLE GET NUMBER OF
SP SIGDIG,DOUBLE2 SIG NUMERIC DIGITS

CVB R1,DOUBLE PUT DEC PL IN R1
ZAP FWORD,=P'0' ZERO COUNTER
LR R14,R1 GET DEC PL
BCTR R14,0 SUB FOR EX
EX R14,MOVECH16 MOVE ZONES
ZAP DOUBLE3,FWORD GET INPUT COUNTER
CVB R1,DOUBLE3

FDZON
CLI 0(R15),C' '
BE FDBCT
CLI 0(R15),C'0'
BL FDERR

FDBCT
LA R15,1(R15,)
BCT R1,FDZON
B FDOK

FDERR
MVI COMASERR,C'Y'
B SAEYERROR

FDOK
CVB R1,DOUBLE FIND
AR R14,R1 DECIMAL
CVB R1,DOUBLE2 POINT
SR R14,R1 AREA
BCTR R14,0

CP
FWORD,SIGDIG INPUT MORE THAN SIG NUMERIC?
BNH USEFW NO USE INPUT COUNTER
ZAP DOUBLE3,SIGDIG GET SIG DIGITS
B GOTS ON

USEFW
ZAP DOUBLE3,FWORD GET INPUT COUNTER
CVB R1,DOUBLE3 POINT
AR R15,R1 TO LAST BYTE
BCTR R15,0 OF INPUT DATA

CVB R1,DOUBLE GET PLEN
CVB R0,DOUBLE2 GET DEC PTS
SR R1,R0 POINT
LR R0,R1 TO
BCTR R1,0 NUMERIC PORTION
EX R1,BLDEC BLANK IT
LR R1,R0 RESTORE REG
MVL
  CLI 0(R15),C'0'     DIGIT?
  BL  BCTDEC       DON'T MOVE
BCTDEC
  MVC 0(1,R14),0(R15) MOVIE IN SIG DIGIT
  BCTR R14.0       BUMP
  BCTR R15.0       BACK
  BCT R1,MVL       KEEP GOIN'
DECEND
  EQU CVB   R1,DOUBLE GET PLEN
  BCTR R1,0       FIX FOR EX
  EX R1,OCDEC   MAKE ALL NUMERIC
  EX R1,MVCDEC   MOVE TO R6
  LA R14,22(R6) BUMP TO
  MVI 0(R2),C'E'     ERROR
SEROOK
  LH R14,HALFWORD RESTORE LENGTH OF TARGET FILED
  BCTR R14.0
  LTR R14,R14 TEST REGISTER
  BM GOBACK IF NEGITIVE DON'T MOVE IT
  CLI COMAFTYP,C'P'
  BE DOPACKED
  CLI COMAFTYP,C'B'
  BE DOBINARY
  EX R14,CLCDATA CHECK IF DATA HAS CHANGED
  BE GOBACK NOCHANGE, GOBACK
  EX R14,Movedata MOVE DATA INTO TARGET FIELD
  BNE FDERR
  CLC COMAOLEN,NUMZONE
  BNE OLOK
  EX R14,Packda
  B DATADONE
DOPACKED
  EQU CLI COMASERR,C'Y'
  BE DATADONE
  MVC CHAR9,NUMZONE
  LA R1,CHAR9
  LA R1.9(R1)
  MVZ NUMCHECK(2),COMAOLEN
  CLC NUMCHECK(2),NUMZONE
  BNE FDERR
  CLC COMAOLEN,NUMZONE
  BNE OLOK
  EX R14,Packda
  B DATADONE
DOBINARY
  EQU CLI COMASERR,C'Y'
  BE DATADONE
  MVC CHAR9,NUMZONE
  LA R1,CHAR9
  LA R1.9(R1)
  MVZ NUMCHECK(2),COMAOLEN
  CLC NUMCHECK(2), NUMZONE
  BNE FDERR
  CLC COMAOLEN,NUMZONE
  BNE OLOK2
  MVC COMAOLEN,COMAPLEN
OLOK2 PACK DOUBLE2, COMAOLEN CVB R2, DOUBLE2 SR R1, R2 EX R2, LOADNUMB PACK DOUBLEE, CHAR9 CVB R1, DOUBLE CLC COMAOLEN, =C'02' BNH DOHALF DOFULL ST R1, (R15) B DATADONE DOHALF STH R1, (R15) B DATADONE DATADONE EQU 01337200 01337300 01337400 01337500 01339300 01339400 01339500 01339600 01339700 01339800 01339900 01340000 01340100 01340200 01340300 01340400 01340500 01340600 01340700 01340800 01340900 01341000 01341100 01341200 01341300 01342000 01343000 01344000 01345000 01346000 01347000 01348000 01349000 01350000 01351000 01352000 01353000 01354000 01355000 01356000 01357000 01358000 01359000 01360000 01361000 01362000 01363000 01364000 01365000 01366000 01367000 01368000 01369000 01370000 01371000 01372000 01373000 01374000 01375000 01376000 01377000 01378000 01379000 01380000 01381000 01382000 01383000 01384000 01385000 01386000 01387000 01388000 01389000 01390000 01391000 01392000 01393000 01394000 01395000 01396000 01397000 01398000 01399000 01400000 01401000 01402000 01403000 01404000 01405000 01406000 01407000 01408000 01409000 01410000 01411000 01412000 01413000 01420000 01430000 01440000 01450000 01460000 01470000 01480000 01490000 01500000

SUBSTITUTE SHEET (RULE 26)
BE TIMOUT1 YES, CONTINUE
CLC COMAVUID,BLANKS HAS THERE BEEN A VALID USER?
BNH GOTMDAS NO, CONTINUE
CLC COMAEKSS(9),COMAVUID NO, IS IT THE SAME ONE?
BE TIMOUT1 YES, CONTINUE
B STARTOVR NO, START OVER

TIMOUT1 EQU
ZAP DOUBLE2,MDASLTIM CHECK TO SEE IF TIMED OUT
MVC MDASLTIM,DOUBLE UPDATE LAST ACCESS TIME
AP DOUBLE2,TIMOUT1 I
CLC DOUBLE,DOUBLE2 I
BNH GOTMDAS IT'S OK
BH PUTCKEY YES, CONTINUE
CLC COMAEKSS,BLANKS WAS COMAKEY THERE ALREADY?
BNH GOTCKEY YES, CONTINUE
MVC COMAKEY(9),COMAEKSS NO, PUT IT IN COMMAREA
B GOTECKEY YES, CONTINUE
PUTCKEY MVC COMAKEY(9),SAVEKSS NO, PUT IT IN COMMAREA

GOTCKEY MVI COMAERRF,C ':
MVI COMACONRF,C ':
MVI COMASGRF,C ':
MVI COMAPRF,C ':
MVI COMATYPF,C ':
MVC COMAEADC,BLANKS
ZAP COMASEG,=P0'0'
MVI COMAPRF,C ':

UNPK COMAMQNM,EIBTASKN COPY TD QUEUE PREFIX
OI COMAMQNM+7,C'0'
MVC COMAPRF,SAVEPROF
MVC COMAMVAR,SAVEMVAR
MVC COMAMVNL,SAVEMVNL
MVC MDASLTIM,DOUBLE
MVC COMACOML,COMPTH

LH R14,=H'11'
LA R15,COMAFICT PUT DATA HERE
MVC COMAFNDF,=C'FACTFICT' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT

LH R14,=H'11'
LA R15,COMAFICT1 PUT DATA HERE
MVC COMAFNDF,=C'FACTFIC1' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT

LH R14,=H'10'
LA R15,SAVETOPC PUT DATA HERE
MVC COMAFNDF,=C'MDASTRAN' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT
CLC SAVETOPC,BLANKS
BNH DGERROR YES, CONTINUE
MVC COMATOPC,BLANKS
MVC COMATOPC(10),SAVETOPC

NOTMDAS EQU

DEBUG DEBUG DEBUG DEBUG DEBUG DEBUG
CLC COMATOPC(10),=C'MI-PINT2'
BNE AFTDBUG1
DEBUG1 B AFTDBUG1
AFTDBUG1 EQU

SUBSTITUTE SHEET (RULE 26)
DEBUG EQU
EXEC CICS UNLOCK DATASET('RAMDAS')
MVC COMAPROC,SAVEPROC
MVI COMAPROF,'C'S' FACTS RESPONSE
B DEQ

DQERROR EQU
MVC MDASKEYY,BLANKS
MVC COMATOPC,BLANKS
B DEQ

DEQ EQU
EXEC CICS DEQ RESOURCE(MIDASSTE) LENGTH(8)
CLI FACTSREQ,'C'Y' FACTS REQUEST?
BNE RETURN NO, RETURN
MVC COMAPROC,SAVEPROC
B RETURN

NORESRC EQU NO RESOURCES AVAILABLE
MVC COMAMESS,BLANKS
MVC COMAMESS(12),='C' NO RESOURCES

ERROR EQU
MVC MDASKEYY,BLANKS

RETURN EQU
EXEC CICS DEQ RESOURCE(MIDASSTE) LENGTH(8) NOHANDLE

RETURN2 EQU
EXEC CICS RETURN

COPY RAMIGETP

BLANKS DC CL130' '
NUMZONE DC 15C'0'
LTORG

END RAMI0300
ALL INFORMATION IN THIS DOCUMENT INCLUDING THE TRANSPORT MECHANISM AND PROTOCOLS, ARE COPYRIGHT 1998 BY THE UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED

SUBROUTINE RAM2A000

FUNCTION:
Read a incoming data and determine what file to read
Read a VSAM data set
Determine if the transaction is update, if so, update record
Format Web Page for browser

PRINT NOGEN
COPY RARAREGS
COPY RAMICOMM
COPY RARAMTEX
COPY RARAGATX
COPY RARAMSTY
COPY RARAMIOB
COPY RARAOLEG
USING COMMAREA,R5

COPY RARAOQUED
QUEUE DSECT

QLINES EQU 45 THESE TWO LINES NEED TO BE IN THE JOB
QLINELEN EQU 200

QUEAREA DSECT
QOUT DS OCL500
QUELINE DS CL500
QUELEN EQU *-QUEAREA
USING QUEAREA,R12

RADSECTO DSECT
INSERTDS
DYNAMIC STORAGE

DFHEISTG DSECT
COPY DFHAID
COPY RARASPAD

INSERTT
UPAD DS C
SCTY DS C
NEWTOPC DS C
AUTOADD DS C
DEFLD DS CL4
ALBS DS CL1
TRECLEN DS CL2
KEYOFF DS CL4
HEDLEN DS CL2
SEGLEN DS CL2
SEGOFF DS CL2
LEVOFF DS CL2
SEGCOUNT DS PL2
NEWSEGS DS PL2
ADDRLEG DS CL4
SADDLEG DS CL4
SCNTOLEG DS CL4
APPID DS CL4
QLINES DS CL3
QLIMIT DS CL3
QLINELEN DS CL3
MAXSEGS DS PL3
FIXED DS C
QINSEG DS CL1
KEYLEN DS CL3
DISPLAY DS C
COURSE DS CL8
SECTION DS CL4
RDONLY DS CL1
SAVE9 DS F
SVE10 DS F
SAVE2 DS F
SVE4 DS F
SAVE7 DS F
SAVE12 DS F
SAVE14 DS F
SAVE15 DS F
RDCTN DS PL3
RDMAX DS PL3
TAGMAX DS PL3
TAGCNT DS PL3
FINDFLD DS CL8
FINDFLTDS CL79
THISSEGS DS PL2
TEMPO3 DS CL3
OUTBLNK DS CL1

PROGRAM RAMI2A00 STARTS HERE

RAMI2A00 DFHEIENT CODEREG=(3,7,8)
RAMI2A00 AMODE 31
RAMI2A00 RMODE ANY

START EQU
CLD EIBCALEN,=H'0'
BE RETURN
L R5,DFHIEICAP
USING COMMAREA,R5
MVI COMACONT,C'

EXEC CICS HANDLE CONDITION DSIDERR(ERVERR)
UNPK KEYLEN,DOUBLE2

SUBSTITUTE SHEET (RULE 26)
OI KEYLEN=2,C'0'
BAL R4,CLEANKEY

INSERTION

PACK DOUBLE,QLINES
CVB R15.DOUBLE
PACK DOUBLE,QLINELEN
CVB R14.DOUBLE
ST R14,FWORD
SR R14,R14
M R14,FWORD
STH R15,COMAQLEN

EXEC CICS GETMAIN LENGTH(COMAQLLEN) INITIMG(X'40') SET (12R)
ST R12,COMAQAADD
LR R14,R12
AH R14,COMAQLEN
PACK DOUBLE,QLINELEN
CVB R15.DOUBLE
STH R15,HALFWORD
SH R14,HALFWORD
SH R14,HALFWORK
ST R14,COMAQ
MVC COMAQLLN,HALFWORD
PACK DOUBLE,QLINES
ZAP RDMAX.DOUBLE
ZAP TAGMAX,DOUBLE
ZAP TAGCNT,=P'3'
ZAP RCNT,=P'0'

MVI COMAERRF,C'1'
MVI COMAENT,C'1'
MVI COMACMD,C'1'
CLI COMASTUD,C'A'
BE CK4ADD

CLC COMATOPL,COMATOPC
BE SAMETOPC
MVI NEWTOPC,C'Y'
B READ2ND

SAMETOPC

LH R14,=H'1'
LA R15,COMACMD
MVC COMAEDTC,BLANKS
MVC COMAEDIO,BLANKS
MVC COMAFNDF,=CMDASNEXT'
MVI COMADECLP,C'
BAL R4,GETDATA
MVI COMAERRF,C'
SR R15,R15
IC R15.0(R14)
LA R15.1(15)
STC R15.0(R14)
STC R15,COMALEVEL
ZAP COMAUSEG,=P'0'

08057500
08058000
08059000
08060000
08067000
08080000
08090000
08100000
08200000
08210000
08220000
08230000
08240000
08250000
08260000
08700000
08710000
08720000
08730000
08740000
08750000
08760000
08770000
08780000
08790000
08791000
08792000
08793000
08794000
08795000
08796000
08797000
08798000
08799000
08800000
08801000
08802000
08803000
08804000
08805000
08806000
08807000
08808000
08809000
08810000
08820000
08830000
08840000
01320000
01330000
01340000
01350000
01360000
01370000
01390000

SUBSTITUTE SHEET (RULE 26)
L R10,ADDRLEG
USING RARAOLEG,R10

ZAP DOUBLE,OLEGSC
CVB R2,DOUBLE
BNP ERROR

LA R10,OLEGHLEN(R10)
USING OLEGS,R10

ZAP DOUBLE2,=P'0'
EQU

CLC P:EGFMA,=C'----'
BE UPBCT

CLI OLEGWHER,C'S'
BE UPSEGS

MVC COMAEDTC,OLEGEDIT
MVC COMAEDIO,OLEGEDIO
MVC COMAFTYP,OLEGFTYP
PACK DOUBLE,OLEGFOFF
CVB R14,COMAFWD3
STH R14,DOUBLE
LR R15,R9
AH R15,COMAFWD3
PACK DOUBLE,OLEGLENG
AP DOUBLE2,DOUBLE
CVB R14,DOUBLE
AR R14,R15 STORAGE VIOLATION PROTECTION
C R14,COMAREN " 
BH BADADDR " 
CVB R14,DOUBLE " 

MVC OMAFNDF(4),APPID
MVC COMAFNDF+4(4),OLEGFNAM
MVC COMAPLEN,OLEGLENG
MVC COMAKEYF,OLEGHEKEY
MVC COMAFTYP,OLEGFTYP
MVC COMAOLEN,OLEGOLEN
MVC COMADECP,OLEGDECP
LH R15,RECLEN
CLI FIXED,C'Y'
BE NOFADD1
AR R15,R14
STH R15,RECLEN
ST R2,SAVE2
EXEC CICS GETMAIN SET(R2) LENGTH(RECLEN) INITIMG(X'40')

NOFADD1

LR R0,R2
L R14,BFILADDR
LH R1,HALFWORD
MVCL R0,R14
LR R9,R2
ST R9,BFILADDR
ST R9,SAVE9
AH R9,HALFWORD
SH R9,SEGLEN
L R2,SAVE2

SUBSTITUTE SHEET (RULE 26)
MVC COMAEDTC, OLEGEDIT
MVC COMAEDIO, OLEGEDIT
MVC COMAFTYP, OLEGFTYP
PACK DOUBLE, OLEGF OFF
CVB R14, DOUBLE
STH R14, COMAFWD3
LR R15, R9
AH R15, COMAFWD3
ST R15, COMAFWD3
PACK DOUBLE, OLEGLENG
AP DOUBLE2, DOUBLE
CVB R14, DOUBLE

AR R14, R15 STORAGE VIOLATION PROTECTION
C R14, COMAREND "
BH BADADDR "
CVB R14, DOUBLE "

MVC COMAFNDF(4), APPID
MVC COMAFNDF+4(4), OLEGFNAM
MVC COMAPLEN, OLEGLENG
MVC COMAKEY, OLEGKEY
MVC COMAFTYP, OLEGFTYP
MVC COMAOLEN, OLEGOLEN
MVC COMADECP, OLEGDECP
BAL R4, GETDATA

CLI ADDFLAG, CY'
BE NODESEG

BCT R4, SEGULOP
SEGSDONE L R9, SAVE9
UPLOVER EQU

CLI COMAENT, CY'
BNE NOENT19

CLI COMAERRF, CY'
BE NOENT19

CLC SEGOFF, HEXZEROS
BE R14, SEGOFF
LH R14, SEGOFF
LR R15, R9
AR R14, R15

CLI SCTY, C'B'
BNE NOTSZB2
LH R15, R(R14)
CVD R15, DOUBLE
AP DOUBLE, NEWSEGS
BH OVERMAX
CVB R15, DOUBLE
STH R15, (2, R14)
NOTSZB2
  CLI  SCTY,C'Z'
  BNE  NOTSZ2
  PACK  DOUBLE,0(2,R14)
  AP  DOUBLE,NEWSEG
  CP  DOUBLE,MXSEG
  BH  OVERMAX
  UNPK  0(2,R14),DOUBLE
  B  NONS19

NOTSZ2
  AP  0(2,R14),NEWSEG
  CP  0(2,R14),MXSEG
  BH  OVERMAX
  EQU
  CLI  UPAD,C'N'
  BE  NOENT19
  EXEC  CICS  REWRITE  DATASET(COMFILE)  FROM(RARSECTO)
  LENGTH(RECLEN)
  MVI  REWRITE,C'R'

LOADSCRN
  EQU
  CLC  COMAS001,BLANKS
  BH  DOS001
  CLI  COMAERRF,C'Y'
  BNE  CKALGO
  CLC  COMAALGN,BLANKS
  BNH  NOALGO
  MVC  COMAIBFN,COMAALGN
  B  DOXXRET

CKALGO
  CLI  NEWTOPC,C'Y'
  BE  NOALGO
  CLC  COMAALGO,BLANKS
  BNH  NOALGO
  MVC  COMAIBFN,COMAALGO
  B  COXXRET

NOALGO
  EQU
  INSERT LOCATION S001

DOS001
  CLC  COMAS001,BLANKS
  BNH  NOS001
  MVC  COMAMSDF,BLANKS
  ST  R12,COMAUP12
  MVC  SPADSKEY,COMAS001
  CLC  COMAS001(4),-C'RAMI'
  BE  NOTOBJ
  EXEC  CICS  HANDLE  CONDITION  NOTFND(NOTOBJ)  NOTOPEN(NOTOBJ)
  DISABLED(NOTOBJ)
  EXEC  CICS  READ  DATASET(RAMIOB)  SET(R1)  R1DFLD(COMAS001)
  LENGTH(HALFWORD)
  USING  RARAMIOB,R1

SUBSTITUTE SHEET (RULE 26)
B TABOVER
CLI OLEGSTABL,C'R'
BNE CKTABD
MVC QOUT(36),=C'\<TR\><TR VALIGNN="TOP" ALIGN="LEFT">'
BNP NOGATX
LA R2,GATXHLEN(R2)
USING GATXS,R2

GATXLOOP
MVC QOUT(79),GATXCOMM
MVI QOUT+79,X'15'
ST R14,FWORD
ST R10,SAVE10
BAL R10,TAGS
L R10,SAVE10
L R14,FWORD
LA R2,GATXSLEN(R2)
BCT R14,GATXLOOP

NOGATXQEXCECCSHANDLECONCONTEXTNOTFND(NOTFND)NOTOPEN(NOTOPEN)
DISABLED(NOTOPEN)
MVC 0(5,R12),=C'\<TD\>'
MVI 5(R12),X'15'
ST R10,SAVE
BAL R10,TAGS
L R10,SAVE10
B TABOVERE

EQU
TABOVERE

EQU
OPBCT
LA R10,OLEGSLEN(R10)
BCT R2,OPLOOP

B CKERRF19

EQU
SLOOP
ZAP THISSEG,=P'0'
ZAP COMAUSEG,=P'1'
ST R10,SADDOLE
ST R2,SCNTOLEG

LH R14,SEGOFF
LR R15,R9
AR R14,R15
CLI SCTY,C'B'
BNE NOTSZB4
LH R15.0(R14)
CVD R15,DONL
ZAP DOUBLE2,DOUBLE
B SKSZ4
CLI SCTY,C'Z'
BNE NOTSZ4
PACK DOUBLE,0(2,R14)
STH R14,COMAFWD3
LR R6,R9
AH R6,COMAFWD3
PACK DOUBLE,OLEGENG
AP DOUBLE2,DOUBLE

SUBSTITUTE SHEET (RULE 26)
CVB R14,DOUBLE
AR R14,R6 STORAGE VIOLATION PROTECTION
C R14,COMAREND
BH BADADDR
CVB R14,DOUBLE
MVC COMAFNDF(4),APPID
MVC COMAFNDF(4),OLEGFNAM
MVC COMAREAS,OLEGHEAD
MVC COMAPLEN,OLEGLENG
MVC COMAKEYF,OLEGKEY
SETB19
MVI OUTBLNK,C'Y'
AP COMAUSEG,=P'1'
B SEGLOOP

CKERRF19 EQU

INSERT1
MVC
AFTER MOVE
CLI COMAERRC,C'Y'
BNE NOERR2
ERR2ND EQU

CLI REWRITE,C'E'
BE DUPE19

MVC SPADSKEY,BLANKS
MVC SPADSKEY(10),=C,'INPUTERROR'
EXEC CICS HANDLE CONDITION NOTFND(N2GATX) NOTOPEN(N2GATX)
RIDFLD(SPADSKEY)

USING RARAGATX,R2
ZAP DOUBLE,GAT
CVB R14,DOUBLE
BNP N2GATX
LA R2,GATXHLEN(R2)
USING GATXS,R2

GATXL002
MVC QOUT(79),GATXCOMM
MVI QOUT+79,X'15'
ST R14,FWORD
ST R10,SAVE10
BAL R10,TAGS
L R10,SAVE10
L R14,FWORD
LA R2,GATXLEN(R2)
BCT R14,GATXL002

N2GATX EXEC CICSHANDLE CONDITION NOTFND(NOTFND) NOTOPEN(NOTOPEN)
DISABLED(NOTOPEN)

B NOUP19

DUPE19
MVC 0(30,R12),=C<">RECORD EXISTS,NOT ADDED</b>'
MVI 30(R12),X'15'
AH R12,COMMAQLLN
B NOUP19

NOERR2 EQU

SUBSTITUTE SHEET (RULE 26)
WO 00/33202

PKLOOP

LA E10,OLEGHLLEN(10)
USING OLEGSR10

ZAP DOUBLE,E=0.0
CLI OLEGKEY,C=''

PKBCT

LA R15,COMAKEY+000
PACK DOUBLE,OLEGFOFF
CVB R14,DDBL
STH R14,COMAFWD3
LA R5,BLANKS
PACK DOUBLE,OLEGLGEN
AP DOUBLE2,E=0.0
CVB R14,DDBL

MVC COMAFCDF(4),APPID
MVC COMAFCDF+4(4),OLEGNAME
MVC COMARED,E=0.0
MVC COMAPLEN,OLEGLGEN
MVC COMAKEY,E=0.0
MVC COMADEP,OLEGDEP
MVC COMAFLR,OLEGFLR
MVC COMARAIN,C=''

BAL R4,PUTDATA

PKBCT

LA R10,OLEGSLEN(10)
BCT R2,PKLOOP

MVC 0(10,R12),C='/<FORM><P>'
MVI 10(R12),X='15'
AH R12,COMAGLLN

MVC QOUT(99),C='/<TR><TR>'
ST R10,SAVE10
BAL R10,TAGS
L R10,SAVE10
MVC QOUT(13),C='/<TR><TABLE>'
ST R10,SAVE10

BAL R10,TAGS
B NOTAG19

ERROR

EQU COMAERRF,C=''

EXEC CICS HANDLE CONDITION NOTFND(ERNOTF) NOTOPEN(ERNOTF)
DISABLED(ERNOTF)

MVC SPADSKEY,BLANKS

SUBSTITUTE SHEET (RULE 26)
MVC SPADSKEY(5) = 'ERROR' 03970000
CLC COMAREGN,BLANKS 03980000
BNE LEAVEEF 03990000
MVC SPADSKEY(8),COMAREGN 04000000
LEVEEF EXEC CICS READ DATASET('RAMYST') SET(R2) LENGTH(HALFWORD) 04010000
USING RARAMSTY,R2 04020000
ZAP DOUBLE,MSTYSC 04030000
CBV R14.DOUBLE 04040000
BNP ERNOTF 04050000
LA R2,MSTYHLEN(R2) 04060000
CKNOTFER
CLC 0(5,R2), = 'FORM' 04070000
BNE CKSTUDER 04071000
MVC 0(79,R12),O(R2) 04072000
MVI 79(R12),X'15' 04073000
AH R12,COMAQLLN 04074000
B NOERBCT 04075000
04076000
CKSTUDER
CLI COMASTUD,C'Y' 04077000
BE NOSTU192 04077002
BE ERNOTF 04077102
CLI COMAUDPDA,C'N' 04078000
BE ERMDASAD 04079000
CLI UPAD,C'N' 04079100
BE ERMDASAD 04079200
B NOEROUT 04079300
ERMDASAD
CLC NOERBCT 04079400
BE NOERBCT 04079500
04079600
NOROUT
MVC 0(79,R12),O(R2) 04079700
AH R12,COMAQLLN 04079800
04079900
NOERBCT
LA R2,MSTYSLLEN(R2) 04080000
BCT R14,CKNOTFER 04080100
B NOSTU192 04080200
04080300
04080400
04080500
04080602
ERNOTF
EQU B NOSTU192 04080700
RETURN
EQU
CLI COMASTUD,C' A' 04080800
BNE DOXXRET 04080900
CLI COMACMD,C' A' 04081000
BNE DOXXRET 04081100
MVC COMAIBFN, = 'C'1' 04081200
B DOXXRET 04081300
04081400
DOXXRET
EQU EXEC CICS RETURN 04081500
COPY RAMIGETP 04081600
COPY RAMIPUTP 04081700
04081800
04081900
04082000
04082100
04082200
04083000
CLEANKEY
LA R10,COMAKEY CLEAR REGISTER 04084000
SR R2,R2 GET LENGTH OF FIELD ENTERED 04085000
IC R2,50 TRANSLATE FIELD TO UPPER 04086000
EX R2,CKTR 04087000
LA R10,COMAKEY 04088000
LA R15,50 04089000
04090000
SUBSTITUTE SHEET (RULE 26)
SUBSTITUTE SHEET (RULE 26)
APPENDIX F

TSO FOREGROUND HARDCOPY
DSNAME=RA.PATENT (RAMI1T00)

PROGRAM RAMI1T00

SASS MENU INTERFACE

ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT
MECHANISM AND PROTOCOLS, ARE COPYRIGHT 1998 BY THE
UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED

COPY RACL0001
COPY RARAREGS

COMMAREA STARTS HERE BASE REG IS R5

COPY RARAGTX COPY COMMAREA DSECT
COPY RAMICOMM COPY COMMAREA DSECT
USING COMMAREA,R5

FILE DSECT STARTS HERE BASE REG IS R9

QLINELEN EQU 140
TSL EQU 5
QOUT DSECT
DS CL79

DFHEISTG WORKING STORAGE
DSECT COMMAREA DSECT

SCRATCH PAD AREA

FOUNDIT DS C
FILENAME DS CL8
TEMPFILD DS CL79
FINDELDT DS CL79
DS OD
DS F

ANSWER D F'O'
DIVIDEND DS OD

QUOTIENT DS F
REMAINS DS F

DIVISOR DS F
MESS DS CL79
THISSEG DS PL3

FLDLEN DS F
AREAEND DS F
AWORD DS F

AREA1 DS F
AREA14 DS F
SV01 DS F
SV014 DS F
SV015 DS F
SIEVE DS F
SAVE6 DS F
SAVE2 DS F

00010003
00020003
00030003
00031037
00032037
00033037
00034037
00035037
00040003
00041003
00042003
00043003
00044003
00045003
00046003
00047003
00047103
00047203
00047303
00047403
00047503
00047603
00047703
00047803
00047903
00048003
00049003
00050003
00060003
00070003
00080003
00090003
01000003
01100003
01200003
01210003
01220003
01230003
01240003
01250003
01260003
01270003
01280003
01290003
01230003
01400003
01410003
01420003
0142103
0142203
0142323
0142423
0142523
0142523
0142723
0142823
0142923
0143023

SUBSTITUTE SHEET (RULE 26)
SAVE12 DS F 00143123
DD2 DS F 00143223
DDO DS F 00143323
DD1 DS F 00143423
DD9 DS F 00143623
DD14 DS F 00143723
DD15 DS F 00143823
SI00 DS F 00143923
SI01 DS F 00144123
SI14 DS F 00144223
SI15 DS F 00145003
MOVEADDR DS F 00146003
MOVELENG DS H 00146103
DDSOFF DS H 00146203
RDMAX DS PL3 00146303
RDINT DS PL3 00146403
TAGMAX DS PL3 00146503
TAGCNT DS PL3 00146603
DECIN DS CL30 00146703
DEGOUT DS CL30 00146803
COPY RASPAD 00147003
PROGRAM RAM1T00 STARTS HERE
CLC COMAEDIO,BLANKS 00150303
BNH EDTOK 00150803
ST R0,SI00 00151003
ST R1,SI01 00152003
ST R14,SI14 00153003
ST R15,SI15 00154003
ST R6,COMAUP01 00155003
ST R12,COMAUP12 00156003
EXEC CICS LINK PROGRAM('RASIS200') COMMAREA(COMMAREA)
LENGTH(COMACOML) 00157003
L R12,COMAUP12 00158003
00159003
L R0,SI00 00160003
L R1,SI01 00170003
L R14,SI14 00180003
L R15,SI15 00190003
B NOSERRS 00200003
EDTOKEQU 00210003
00220003
CLC COMABLZL,NUMZONE 00230003
BNH NOBLZL 00240003
LR R1,R6 00250003
PACK DOUBLE,COMABLZL 00260003
CVB R0,DOUBLE 00261003
CLC GATXCOMM(8),=C'<SELECT' 00261103
BE DOSELC 00261203
CLC GATXCOMM(8),=C'<OPTION' 00261303
BNE CKRADIO 00261403
PACK DOUBLE,COMAPLEN 00261503
CVB R15,DOUBLE 00261603
BCTR R15,0
EX R15,CHEKOPTN
BNE NOTOPT
MVC TEMPFLD(17),=C’<OPTION SELECTED ’
MVC TEMPFLD+17(62),GATXCOMM+8
MVC GATXCOMM,TEMPFLD
00269723
B NOTOPT
00269823

CKRADIO
CLC GATXCOMM(19),=C’<INPUT TYPE="RADIO" ’
BNE CKAREA
00270023
CLC GATXCOMM+26(4),=C’@@@’
BNE NOTOAPI
MVC GATXCOMM+26(4),COMAAPID
00270523

NOTOAPI
PACK DOUBLE,COMAPLENT
CVB R15,DOUBLE
BCTR R15,0
EX R15,CHEKRADI
00271223
BNE NOTOPT
00271323
MVC TEMPFLD(27),=C’<input type="RADIO" CHECKED’
MVC TEMPFLD+27(52),GATXCOMM+19
MVC GATXCOMM,TEMPFLD
00271723
B NOTOPT
00271823

CKAREA
CLC GATXCOMM(16),=C’<textarea name=""’
BE DOAREA
CLC GATXCOMM(12),=C’<textarea>’
BNE NOTOPT
MVC 0(79,R12)GATXCOMM
MVI 79(R12),X’15’
BAL R10,TAGS
B SKTOPT
00272826

DOAREA
CLC GATXCOMM+16(4),=C’@@@’
BNE NOARAPI
MVC GATXCOMM+16(4),COMAAPID
00273126

NOARAPI
MVC 0(79,R12)GATXCOMM
MVI 79(R12),X’15’
BAL R10,TAGS
CLJ COMADECP,C’0’
BNH NODECP
00273226

DOCHARP
MVC DECIN,0(R6)
BAL R9,DECROUT
L R15,FWORD
AR R12,R15
00283926
LA R12,9(R12)
LA R14,1(R14)
AR R14,R0
B MOVEOVER
00284226
NODECP

EX R15,MOVEDATO
B MOVEOVER

DOPACKED

EQU
MVZ NUMCHECK(2),COMAOLEN
CLC NUMCHECK(2),NUMZONE
BNE USEPLEN2
CLC COMAOLEN,NUMZONE
BH OLENOK

USEPLEN2

MVC COMAOLEN,COMAPLEN

OLENOK

EQU
PACK DOUBLE,COMAPLEN
CVB R15,DOUBLE
BCTR R15,0
B SKIPUP
00285926

UNPKPACK

UNPK SPADSKEY(15),0(*,-,R6)
00286026

00286134

SKIPUP

EX R15,UNPKPACK
00286226

OI SPADSKEY+14,C'0'
PACK DOUBLE,COMAOLEN
CVB R15,DOUBLE
BCTR R15,0

LA R2,SPADSKEY
LA R2,15(R2)
SR R2,R15
BCTR R2,0

CLI COMADECP,C'0'
BNH NOPDEC
LA R2,1(R2)
BCTR R15,0
EX R15,UNPKDATP

STH R15,MOVELENG
ST R14,MOVEADDR
MVC DECIN(0,R2)
CLI COMAFYCP,C'B'
BE DIBINARY

ST R15,FWORD
PACK DOUBLE,COMAPLEN
CVB R15,DOUBLE
BCTR R15,0

CLI COMADECP,C'0'
BNH NODICP
STH R15,MOELENG
ST R12,MOVEADDR
MVC DECIN(0,R6)
BAL R9,DECROUT

L R15,FWORD
AR R12,R15
00297126
LA R12,9(R12)
B ENDTD

SUBSTITUTE SHEET (RULE 26)
NODICP
EX R15,MOVEDITO
AR R12,R15
00297626
LA R12,1(R12)
MVI 0(R12),X'15'
LA R12,1(R12)
B NOSERRS

DECROUT
EQU
MVC DECOUT,7(R14)
00298326
PACK DOUBLE,COMADEC2 GET # OF DEC PLACES
00298426
CVB R0,DOUBLE PUT IN DOUBLE WORD
EX R15,MODEDEC2 MOV TO TEMP AREA SPAD24
00298626
SR R15,R0 GET LENGTH OF NON DEC PART
EX R15,MODEDEC2 MOVE IT OUT
LA R14,DECOUT
00299326
LR R1,R14 SETUP REG 1
LA R1,1(R1)
SKIP OVER HEADER
00299126
AR R1,R15
ADD NON DEC LEN
00299226
STH R15,HALFWORD SAVE THIS ADDRESS
00299306
MVI 0(R1),C'-' MOVE IN DEC POINT
00299426
LA R1,1(R1)
POINT TO DEC AREA
00299526
LA R15,SPAD24 DATA IS IN SPAD24
00299626
AH R15,HALFWORD ADD NON DEC LEN
00299726
LA R15,1(R15) SKIP OVER DEC POINT
00299826
ST R14,AWORD SAVE THIS ADDRESS
00299926
LR R14,R0 GET # DEC PLACES
00300026
EX R14,MODEDECO MOVE FROM SPAD24 TO OUTPUT
00300126
L R14,AWORD RELOAD R14
00300226
AR R1,R0 POINT TO END OF AREA
00300326
MVI 0(R1),X'15' LINE FEED
00300426
LH R15,MOVELEN
00300526
LA R15,1(R15)
00300626
LA R15,2(R15)
00300726
L R14,MOVEADDR
00300826
EX R15,MODEMOVE
00300926
DECROVER
BR R9
00301026
MODEMOVE MVC 7('=*..R14).DECOUT
00301126
MODEDEC2 MVC DECOUT(*),SPAD24
00301226
MODEDECO MVC 0('=*..R1).0(R15)
00301326

COLORS
CLI COMACOLR,C'`
00301426
BHN NOCOLR
00301526
MVC QOUT(13),=C'<RONT COLOR="'
00301626
CKRED
CLI COMACOLR,C'R
00301726
BNE CKBLUE
00301826
MVC QOUT+13(05),=C'RED'">
00301926
B COLRD
00302026
CKBLUE
CLI COMACOLR,C'B'
00302126
BNE CKGREEN
00302226
MVC QOUT+13(05),=C'BLUE'">
00302326
B COLRD
00302426
BCTR R15,0
00302526
UNPK SPADSK(9).DOUBLE2
00302626
00302726
00302826

SUBSTITUTE SHEET (RULE 26)
OI SPADSKEY+8,C'0'
LA R2,SPADSKEY
LA R2,9(R2)
SR R2,R15
BCTR R2,0
EX R15,UNPKDITO
00315226
AR R12,R15
00315326
LA R12.1(R12)
MVI 0(R12),X
LA R12.1(R12)
B NOSERRS
NOMOVER LA R14,7(R14)
B AMOVER
MOVER AR R14,R15
00316326
LA R14,8(R14) OFFSET OF 7 + ADJ OF 1
AMOVER MVC 0(2,R14),=C' *'
BAL R10,TAGC
MVC 00(25,R12),=C'SIZE="--", MAXLENGTH="--"'
MVZ NUMCHECK(2),COMAOLEN
CLC NUMCHECK(2),NUMZONE
BNE USEPLEN
CLC COMAOLEN,NUMZONE
BH OLENOK2
USEPLEN MVC 06(2,R12),COMAPLEN
MVC 21(2,R12),COMAPLEN
B LENDONE
OLENOK2 MVC 06(2,R12),COMAOLEN
MVC 21(2,R12),COMAOLEN
LENDONE MVI QOUT+25,'X'15'
CLC 06(2,R12),=C'01'
BNE NOT01
MVC 06(2,R12),=C'02'
NOT01 EQU
MVI 25(R12),X'15'
BAL R10,TAGC
MVC 0(5,R12),=C'/&TD'
BAL R10,TAGC
BAL R10,ERROUT
CLI COMAERET,C'E'
BNE NOSERRS
MVC QOUT(38),=C'<td><IMG SRC="/ourXXXXXXX.GIF"/></td>'
MVC 0(09,R12),=C'&OPTION>'
MVI 9(R12),X'15'
L R12,SAVE12
BAL R10,TAGC
LA R14,SPADSKEY
00329826
PACK DOUBLE.19(4,R4)
CVB R1.DOUBLE
AR R14,R1
BCTR R14,0
IC R15.0(R14)

SUBSTITUTE SHEET (RULE 26)
LA R15,1(R15)
STC R15,0(R14)
B DDREAD

OPTOVER EQU 00331626
ENODOPT EQU
L R0,DD0
L R1,DD1
L R14,DD14
L R15,DD15
BR R2

MOVEVAL MVC 0(('.*',R12),0(R9)
MOVEDISP MVC 0(('.*',R12),0(R9)
MOVEOKEY MVC SPADSKEY('.*'),0(R9)
CHECKDS CLC 0(('.*',R14,R6)
ERRROUT EQU
MVI COMASERR,C', '
MVC COMAÆRET,BLANKS

L R14,COMAUP14
L R15,COMAUP15
L R1,COMAUP
ZAP THISSEG,

STH R14,HALFWORD
R6,COMAMVAR POINT TO BEGINNING OF INCOMING DATA
L R1,COMAMVLEN GET TOTAL LENGTH OF THE DATA
MVC FINDFLDT,BLANKS

QUELLOOP EQU
CLC COMAUSEG,=X'000000'
BE SAVR6F
CP COMAUSEG,=P'0'
BE SAVR6F
B CKFNDF
SAVR6F
MVI FOUNDIT,C'Y'

TAGC EQU
TAGS EQU 0072003

NORES C R12,COMAQEND
BNL TAGENDC
AP TAGCNT,=P'1'
AH R12,COMAQLLN
B NOTAGC

LH R14,COMAQLLN
CLI 0(R12),X'25'

SUBSTITUTE SHEET (RULE 26)
BE RESDONE 00810003
CLI 0(R12),X'15' 00820003
BE RESDONE 00830003
CLI 0(R12),C' ' 00840003
BH RESDONE 00850003
BCTR R12,0 00860003
BCT R14,RESR12 00870003
B NOTAGC 00880003
RESDONE EQU * 00890003
LA R12,1(R12) 00900003
NOTAGC BR R10 00910003
00920003
TAGENDC AH R12,COMAQLLN 00930003
MVI COMACONT,C' ' 00940003
EXEC CICS LINK PROGRAM('RAM10100')COMMAREA(COMMAREA) 00950003
LENGTH(COMACOML) 00960003
MVI COMACONT,C' ' 00970003
00980003
01000003
L R0,COMAQLADD POINT REG 0 TO SEGMENTS 01010003
LH R1,COMAQLLEN LOAD REG 1 WITH SEGMENT LENGTH 01020003
SR R15,R15 SET LENGTH TO ZERO 01030003
IC R15,=C' ' 01040003
SLA R15,24 SET PAD CHARACTER TO BLANK 01050003
LA R14,* MOVE PAD CHARACTER TO BITS 1-8 01060003
MVCL R0,R14 R14 MUST BE A VALID ADDRESS 01070003
01080003
CLI COMAQERR,C' ' SUCCESSFUL CREATE? 01090003
BE ERROR 01100003
01110003
L R12,COMAQLADD 01120003
MVI COMACONT,C' ' 01130003
01140003
01150003
01160003
B NOTAGC 01170003
01180003
ERROR EQU 01190003
MVI COMAERRF,C' ' 01200003
MVI COMASERR,C' ' 01210003
BR R10 GOBACK 01220003
01230003
RETURN EQU 01240003
SFW001
MVC COMAOLEN,BLNKNS 01241003
MVC COMAKEYE,BLNKNS 01242003
MVI COMACOLR,C' ' 01243003
MVI COMADECP,C' ' 01244003
ZAP COMARDMX RDMAX 01245003
ZAP COMARDCT,RDCNT 01245103
ZAP COMATGMX,TAQMAX 01245203
ZAP COMATGCT,TAGCNT 01245303
ST R12,COMAUP01 01245403
01245503
01245603
EXEC CIS RETURN 01245703
CONSTANTS

ASIS DC C'N' Y=NO UPPER/LOWER CASE TRANSLATION
BLANKS DC CL133' 
HEXZERO DC 30'00'
NUMZONE DC 24'C'0'
HEXFF DC 30'FF'

CKTR
UPCASE TR 0(*-.R6).UPCASE TRANSLATE TO UPPER CASE

DC X'000102030405060708090A0B0C0D0E0F'
DC X'101112131415161718191A1B1C1D1E1F'
DC X'202122232425262728292A2B2C2D2E2F'
DC X'303132333435363738393A3B3C3D3E3F'
DC X'404142434445464748494A4B4C4D4E4F'
DC X'505152535455565758595A5B5C5D5E5F'
DC X'606162636465666768696A6B6C6D6E6F'
DC X'707172737475767778797A7B7C7D7E7F'
DC X'808182838485868788898A8B8C8D8E8F'
DC X'909192939495969798999A9B9C9D9E9F'
DC X'A0A1A2A3A4A5A6A7A8A9AAABACADAEAF'
DC X'B0B1B2B3B4B5B6B7B8B9BBBCBCBDCBDEBF'
DC X'C0C1C2C3C4C5C6C7C8C9CACBCDCDCECF'
DC X'D0D1D2D3D4D5D6D7D8D9DADDADCDDDDDDEDF'
DC X'E0E1E2E3E4E5E6E7E8E9EAEBECEDDEEEF'
DC X'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFDFEFF'

DS OH

LTORG
END RAM1T00
APPENDIX G

ENTRY PONT IS:  I1500

ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT MECHANISM AND
PROTOCOLS, ARE COPYRIGHT 1998 BY THE UNIVERSITY OF FLORIDA. ALL RIGHTS
RESERVED.

COMMAREA DSECT
EVENTUAL COMMAREA FOR THIS MODULE
COPY RARAMDAS
USING RARAMDAS,R9
COPY RAMICOMM
USING COMMAREA,R5
COPY RARAMIDP
COPY RARATCPB

DEFINITION OF REGISTERS
COPY RARAREGS
CICS EXEC INTERFACE DYNAMIC STORAGE AREA DSECT
DFHEISTG
COPY RARASPA

SAVE TRAN DS CL4
NO RESPND DS C

WORK AREAS
WORKD DS D -- DOUBLE WORD CITATION NUMBER
WORKD2 DS D -- DOUBLE WORD MESSAGE NUMBER
WORKF DS F -- FULLWORD WORK AREA
TEMPL TEN DS F -- FULLWORD WORK AREA

TEMPORARY STORAGE CONTROL
DSTSTIM# DS H B ITEM NUMBER OF ENTRY IN QUEUE
DSTSTLEN DS H B TEMPORARY STORAGE DATA AREA LENGTH
DSTSTPL DS H B ITEM NUMBER OF BIB 4 PACLINK
DSTSTDAAD DS A ACON TEMPORARY STORAGE DATA AREA ADDRESS
DSTSNAM E DS CL8 AN TEMPORARY STORAGE QUEUE NAME

PROGRAM CONTROL
DSPCCALN DS H B COMMUNICATIONS AREA LENGTH
DSPCNAME DS CL8 AN PROGRAM NAME
DSPCSAVE DS 15F AN REGISTER SAVE AREA
SAVER14 DS A SAVE R14 ADDRESS
SAVER14A DS A SAVE R14 ADDRESS
DSR14SAV DS A SAVE R14 ADDRESS

STORAGE CONTROL
DSSCLE N DS H B STORAGE LENGTH
DSSCAD DS A ACON STORAGE ADDRESS
DSSNXTPOS DS A A(NEXT POSITION IN BUFFER)
DSSNDADD DS A A(SEND BUFFER)
DSSNDEAD DS A A(END OF BUFFER)
DSRECEND DS A A(END OF RECORD)
DSSNDLEN DS H H LENGTH(SEND BUFFER)
DSDEST DS 255C DESTINATION
DSDSTLN DS H H LENGTH(DESTINATION)

NEW DATE STUFF
DSUTIME DS CL8 TIME FOR ASKTIME CALL
DSDOM DS F DAY OF MONTH
DSDOW DS F DAY OF WEEK
DMSOY DS F MONTH OF YEAR
DSYEAR DS F YEAR
DSDATE DS CL10 DATE
DSAPPLI DS CL8 APPLID
DSTIME DS CL10 TIME
DSJC DS CL72 JOB STATEMENT
<table>
<thead>
<tr>
<th>Variable</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSTRYCNT</td>
<td>DS</td>
<td>RETRY COUNT FOR HOST RESPONSE</td>
</tr>
<tr>
<td>DSCRFLAG</td>
<td>DS</td>
<td>CARRIAGE RETURN LINE FEED FLAG</td>
</tr>
<tr>
<td>FTEMP</td>
<td>DS</td>
<td>TEMPORARY FULL WORK</td>
</tr>
<tr>
<td>DSNCOMM</td>
<td>DS</td>
<td>COMM AREA FOR EZACIC25(GETHOSTBYNAME)</td>
</tr>
<tr>
<td>DSRETCOD</td>
<td>DS</td>
<td>RETURN CODE FROM GETHOSBYNAME CALL</td>
</tr>
<tr>
<td>DSERRO</td>
<td>DS</td>
<td>ERROR NUMBER</td>
</tr>
<tr>
<td>HOSTENT</td>
<td>DS</td>
<td>ADDRESS OF HOSTENT STRUCTURE</td>
</tr>
<tr>
<td>DSNMND</td>
<td>DS</td>
<td>REQUESTED OPERATION (GHBN)</td>
</tr>
<tr>
<td>DSNMLEN</td>
<td>DS</td>
<td>LENGTH OF NAME TO LOOK UP</td>
</tr>
<tr>
<td>DSNQTYPE</td>
<td>DS</td>
<td>TYPE OF QUERY</td>
</tr>
<tr>
<td>CSHFIRST</td>
<td>EQU</td>
<td>USE CACHE FIRST THEN GETHOSTBYNAME</td>
</tr>
<tr>
<td>DSNONLY</td>
<td>EQU</td>
<td>DO GETHOSBYNAME CALL ONLY</td>
</tr>
<tr>
<td>CHONLY</td>
<td>EQU</td>
<td>ATTEMPT QUERY USING CACHE ONLY</td>
</tr>
<tr>
<td>DSNHNAME</td>
<td>DS</td>
<td>HOST NAME TO LOOK UP</td>
</tr>
<tr>
<td>DSNCOMMS</td>
<td>EQU</td>
<td>*DSNCOMM</td>
</tr>
<tr>
<td>TCPINPUT</td>
<td>DS</td>
<td>INPUT FROM THE TCP/IP LISTENER</td>
</tr>
<tr>
<td>SOCKDESC</td>
<td>DS</td>
<td>SOCKET DESCRIPTOR</td>
</tr>
<tr>
<td>MVSADDR</td>
<td>DS</td>
<td>MVS ADDRESS SPACE IDENTIFIER</td>
</tr>
<tr>
<td>TCPTASK</td>
<td>DS</td>
<td>TCP/IP TASK IDENTIFIER</td>
</tr>
<tr>
<td>DATAAREA</td>
<td>DS</td>
<td>CLIENT DATA AREA</td>
</tr>
<tr>
<td>FILLER</td>
<td>DS</td>
<td>FILLER</td>
</tr>
<tr>
<td>SSOCKADDR</td>
<td>DS</td>
<td>SOCKET ADDRESS</td>
</tr>
<tr>
<td>PORT</td>
<td>DS</td>
<td>TCP/IP ADDRESSING FAMLY</td>
</tr>
<tr>
<td>ADDRESS</td>
<td>DS</td>
<td>PORT DESCRIPTION</td>
</tr>
<tr>
<td>DZERO</td>
<td>DS</td>
<td>IP ADDRESS</td>
</tr>
<tr>
<td>TCPINS</td>
<td>EQU</td>
<td>XL8 RESERVOED (MUST BE ZEROS)</td>
</tr>
<tr>
<td>PARMLIST</td>
<td>DS</td>
<td>30A</td>
</tr>
<tr>
<td>SOCFUNC</td>
<td>DS</td>
<td>SOCKET FUNCTION NAME</td>
</tr>
<tr>
<td>SOCRECV</td>
<td>DS</td>
<td>SOCKET DESCRIPTOR</td>
</tr>
<tr>
<td>PROTO</td>
<td>DS</td>
<td>SOCKET PROTOCOL</td>
</tr>
<tr>
<td>SCOTYPE</td>
<td>DS</td>
<td>SOCKET TYPE 1=STREAM,2=UDP</td>
</tr>
<tr>
<td>RETCODE</td>
<td>DS</td>
<td>RETURN CODE</td>
</tr>
<tr>
<td>ERRNO</td>
<td>DS</td>
<td>ERROR NUMBER</td>
</tr>
<tr>
<td>NBYTES</td>
<td>DS</td>
<td>SIZE OF BUFFER FOR SOCKET READ</td>
</tr>
<tr>
<td>FLAGS</td>
<td>DS</td>
<td>FLAGS FOR SOCKET CALLS</td>
</tr>
<tr>
<td>NAME</td>
<td>DS</td>
<td>SOCKET ADDRESS</td>
</tr>
<tr>
<td>NFAMILY</td>
<td>DS</td>
<td>TCP/IP ADDRESSING FAMLY</td>
</tr>
<tr>
<td>NPORT</td>
<td>DS</td>
<td>PORT DESCRIPTION</td>
</tr>
<tr>
<td>NADDRESS</td>
<td>DS</td>
<td>IP ADDRESS</td>
</tr>
<tr>
<td>NDZERO</td>
<td>DS</td>
<td>XL8 RESERVOED (MUST BE ZEROS)</td>
</tr>
<tr>
<td>CLNTHNDL</td>
<td>DS</td>
<td>SOCKET DESCRIPTOR OF CLIENT MACHINE</td>
</tr>
<tr>
<td>MAXSOC</td>
<td>DS</td>
<td>MAX. NUM OF SOCKETS OPEN AT ONE TIME</td>
</tr>
<tr>
<td>MAXNOS</td>
<td>DS</td>
<td>HIGHEST SOC NUM ASSIGNED TO APP.</td>
</tr>
<tr>
<td>AF</td>
<td>DS</td>
<td>ADDRESSING FAMILY (MUST = 2)</td>
</tr>
<tr>
<td>SUBTASK</td>
<td>DS</td>
<td>SUBTASK IDENTIFIER</td>
</tr>
<tr>
<td>BUF</td>
<td>DS</td>
<td>BUFFER FOR SOCKET WRITE</td>
</tr>
<tr>
<td>INBUF</td>
<td>DS</td>
<td>BUFFER FOR SOCKET READ</td>
</tr>
<tr>
<td>OPTVAL</td>
<td>DS</td>
<td>OD</td>
</tr>
<tr>
<td>OPTVONOF</td>
<td>DS</td>
<td>F</td>
</tr>
<tr>
<td>OPTVLEN</td>
<td>DS</td>
<td>F</td>
</tr>
<tr>
<td>OPTNAME</td>
<td>DS</td>
<td>F</td>
</tr>
<tr>
<td>TIMEOUT</td>
<td>DS</td>
<td>OD</td>
</tr>
<tr>
<td>TIMEOUTS</td>
<td>DS</td>
<td>F</td>
</tr>
<tr>
<td>TIMEOUTM</td>
<td>DS</td>
<td>F</td>
</tr>
<tr>
<td>RSENDMSK</td>
<td>DS</td>
<td>F</td>
</tr>
</tbody>
</table>

_**SUBSTITUTE SHEET (RULE 26)**_
APPENDIX D

TSO FOREGROUND HARDCOPY
DSNAME=RA.PATNET
RAMI0300 TITLE 'STATE MANAGEMENT FOR MIDAS'

ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT MECHANISM AND
PROTOCOLS, ARE COPYRIGHT 1998 BY THE UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED.

SUBROUTINE RAMI0300

FUNCTION:
EVALUATE DATA FROM TCP/IP REQUEST
CREATE A NEW STATE KEY OR VALIDATE THE ONE THAT IS IN DATA

DSECTS
  DSECT
  COPY RAMICOMM
  COPY RARAMDAS
  COPY RARMIDP
  USING RARMIDP.R12
MAXTRIES
  EQU 10
  COPY REGISTER DEFINITION
  COPY RAREGS

DYNAMIC STORAGE
TEMP STORAGE HERE
  DFHEISTG
  COPY RARS PAD
COMLTH
  DS H
CMLENGTH
  DS H
SAVETRY
  DS H
NUMCHEK
  DS CL15
UNPKTIME
  DS CL15
MIDASSTE
  DS CL8
UPDATE
  DS C
TIMEOUT1
  DS PL4
TIMEOUT2
  DS PL4
TCPDSTAT
  DS OCL16
TCPDSTTN
  DS CL8
TCPDSSTTE
  DS CL8
FINDFLD
  DS CL8
FINDFLDT
  DS CL79
UPDatem
  DS C
FACTSREQ
  DS C
SAVETOpc
  DS CL10
SAVEPProf
  DS C
S A V E M V A R
  DS CL4
SAVEMVLN
  DS CL4
SAVEEXSS
  DS CL10
SAV E P R O C
  DS CL10
STUID
  DS CL20

PROGRAM RAMI0300 STARTS HERE
RAMI0300 DFHEI E NT CORDEREG=R,8)
RAMI0300 AMODE 31
RAMI0300 RMODE ANY
EXEC CICS HANDLE CONDITION DSIDERR( ERROR )
ERROR( ERROR ) TERMIDERR( ERROR ) SYSIDERR( ERROR )
ISCINVREQ( ERROR ) INVREQ( ERROR ) IOERR( ERROR )
DISABLED( ERROR ) ILLOGIC( ERROR ) LENGERR( ERROR )

SUBSTITUTE SHEET (RULE 26)
EXEC CICS HANDLE CONDITION NOTFND(NOMDAS) NOTOPEN(RESULT)

START EQU
CLC EIBCALEN,=H'0'
BNH RETRNV2
L R9,DFHEICAP
USING RARAMDAS,R9
LA R5,MDASLEN
GET LENGTH FOR TOP OF MDAS
AR R5,R9
ADD IT TO R9
USING COMMAREA,R5

MVC COMLTH,COMACOML SAVE COMACOML
SR R14,R14
CLEAR REG
LA R14,MDASLEN
GET LENGTH FOR TOP OF MDAS
AH R14,COMLTH
ADD TO GET TOTAL NEEDED
STH R14,CMLLENGTH
SAVE IT

SPACE
LH R14,=H'10'
LA R15,SAVEPROC
PUT DATA HERE
MVC COMAFNDF,=C'PROCESS' DATA IDENTIFIER
BAL R4,GETDATA
GO GET IT
CLC SAVEPROC,BLANKS
BNH NOSTATE
YES, CONTINUE

EXEC CICS HANDLE CONDITION NOTFND(NOTPROC) NOTOPEN(NOTPROC)

LH R14,=H'20'
LA R15,STUID
PUT DATA HERE
MVC COMAFNDF,=C'STUID' DATA IDENTIFIER
BAL R4,GETDATA
GO GET IT
MVI COMAFYP,C'U'
LA R14,9
PUT DATA HERE
LA R15,SAVEEXSS
PUT DATA HERE
MVC COMAFNDF,=C'MDASSTID'
BAL R4,GETDATA
GO GET IT

MVI COMAFYP,C'U'
LA R14,9
PUT DATA HERE
LA R15,SAVEEXSS
PUT DATA HERE
XC NUMCHECK,NUMCHECK

!!! HARD CODED TIMEOUT1 FOR NOW AT 5 MINUTES!!!
CLI TCP2STTE,C'M'
BNE NOMDAS
-------------READ FILE FOR STATE CHECK-------------

EXEC CICS READ DATASET('RAMDAS') SET(R2)
LENGTH(HALFWORD) RIDFLD(TCP2STTE)UPDATE
CLC CMLENGTH,HALFWORD MAKE SURE LENGTH IS OK
BL ERROR ERROR IF TOO LONG FOR AREA
CLC DOUBLE,DOUBLE2 I
BH TIMEOUT1 IT HAS BEEN TOO LONG, START OVER
BNH CKCONTN
EXEC CICS START TRANSID('M106')
B STARTOVR
IT HAS BEEN TOO LONG, START OVER

CKCONTN EQU
CLC SAVEEXSS,BLANKS WAS A NEW SSN ENTERED?
BNH NONSSN NO, CONTINUE
MVC COMAEEXSS(9),SAVEEXSS YES, PUT IT IN COMMAREA
NONSSN CLC COMASIGN,=C ADMINISTRATIVE USER?
WSNDMSK   DS   F   WRITE SEND MASK
ESNDMSK   DS   F   EXCEPTION SEND MASK
RRETMASK  DS   F   READ RETURN MASK
WRETMASK  DS   F   WRITE RETURN MASK
ERETMSK   DS   F   EXCEPTION RETURN MASK
SMAXSOC   DS   F   LARGEST SCK. DES. + 1
SR1       DS   F   SAVE R1
SAVEQLEN  DS   H   QUEUE LENGTH
SAVEQITE  DS   H   QUEUE ITEM NUMBER
CODE      DS   H   STARTCODE
COMMLLEN  DS   H   INCOMING COMMAREA LENGTH
FROMTCP   DS   C   STARTED FROM TCP/IP REQUEST
GOTPROC   DS   C   GOT A PROCESS READ
SAVER6    DS   F   SAVE REG 6
SAVER15   DS   F   SAVE REG 15
DATALEN   DS   F   LENGTH OF DATA FIELDS
GOTDATA   DS   C   LENGTH OF DATA FIELDS
PLACE     DS   PL1  LENGTH OF DATA FIELDS
WKLEN5    DS   CL5  CHARACTER FIELD FOR BUFFER LENGTH
WKLEN6    DS   CL6  CHARACTER FILED FOR BUFFER LENGTH
* EMAILTO1  DS   CL79  1ST RECP EMAIL ADDRESS
EMAILTO2  DS   CL79  2ND RECP EMAIL ADDRESS
EMAILTO3  DS   CL79  3RD RECP EMAIL ADDRESS
EMAILTO4  DS   CL79  4TH RECP EMAIL ADDRESS
EMAILTO5  DS   CL79  5TH RECP EMAIL ADDRESS
EMAILFROM DS   CL79  EMAIL ADDRESS OF SENDER
EMAILSUB  DS   CL79  EMAILSUBJECT LINE
MAINLINE  DS   OH   
EXEC      CICS ASSIGN STARTCODE (CODE)
CLC       CODE = 'CS'D'
BNE       STARTXCT
EXEC      CICS RETRIEVE SET(R5) LENGTH(COMMLLEN)
MVC       EIBCCLLEN, COMMLLEN UPDAT EIBCALEN
           B   GOTARE

STARTXCT  EQU                     LOAD ADDR COMM.AREA
           R5,DFHEICAP     COMMAREA GREATER THAN 0
           L   EIBCALEN = 'H'0'
           BNH  RETURN

GOTARE    EQU                    COPY TD QUEUE PREFIX
           EIBALEN, COMACOML
           CLC   RETURN
           BL    UNPK
           OI    COMAMQNM, EIBTASKN
           BNH  RETURN
           BAL   R14, IPAD2BIN

CKRPORT   LA                      PUT DATA HERE
           R14,5
           LA                      DATA IDENTIFIER
           R15SAVEPORT
           MVC                  GO GET IT
           SAVEPORT, BLANKS
           MVC                  DATA IDENTIFIER
           COMAFNDF = 'C' PORTNUMB
           BAL                  GO GET IT
           R4, GETDATA

           CLC   SAVEPORT, BLANKS
           BNH  RETURN
           PACK   DOUBLE, SAVEPORT
           CVB   R14, DOUBLE
           STH   R14, NPORT

SUBSTITUTE SHEET (RULE 26)
WO 00/33202

72

CKRINIT EQU *
XC SAVEINIT,SAVEINIT
LA R14,50
ST R14,SAVEINIT INIT THE INIT STRING TO 50
LA R15,SAVEINIT PUT DATA HERE
MVC COMAENDE,'=C' 'RETNODATA' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT
CLC SAVEINIT,(50),BLANKS WAS SOMETHING SENT?
BL RETURN NOT AT LEAST BLANKS, GET OUT
BH GOTINIT NOT AT LEAST BLANKS, GET OUT
XC SAVELINI,SAVELINI NOTHING THERE, ZERO LENGTH

GOTINIT EQU
BAL R14,GETBUG GETMAIN A BUFFER
B ATFGINFO

NOTM12E EQU

BAL R14,GETBUF GETMAIN A BUFFER
BAL R14,GETINFO GET THE NECESSARY CALL INFO

AFTGINFO EQU

CONNECT TO COMMUNICATIONS SERVER
MVI DSCRFLAG,C ' ' CLEAR<CR><LF> FLAG
BAL R14,CONNHOST

CLC NPORT,'=H'25'
BE DOEMAIL

BAL R14,SENDHDR

BNE CUTOFF NO, BYE

MVC COMAI$BFN,'=C'3' SEND REGULAR MESSAGE
MVI SAVEWAIT,'C'N'
MVC COMAPTAR,COMATOPC SAVE TARGET REQUEST

LR R0,R9 ADDRESS OF DATA
AH R0,=AL2(MDASLEN) BUMP TO WHERE COMMAREA GOES
LR R14,R5 ADDRESS OF COMMAREA
LH R15,COMACOML LENGTH OF COMMAREA
LR R1,R15

MVC R0,R14 MOVE THE DATA
LH R15,COMACOML LENGTH OF COMMAREA
AH R15,=AL2(MDASLEN) ADD LENGTH OF COMMAREA
STH R15,HALFWORD SAVE LENGTH OF DATA

REWRMDAS EQU
EXEC CICS REWRITE DATASET('RAMDAS')
LENGTH(HALFWORD) FROM(RARAMDAS)

CLC RETCODE,,DFHRESP(NOTFND) WAS IT NOT FOUND
BE GOCLOS
CLC RETCODE,DFHRESP(NOTOPEN) WAS NOT OPEN
CLC RETCODE,DFHRESP(DISABLED) WAS IT DISABLED
BE GOCLOS
CLC RETCODE,DFHRESP(NORMAL) WAS IT NORMAL
BNE RETURN
B GOCLOS

PUTCONT MVC COMAMESS(8),='C' 'CONTINUE'
EXEC CICS UNLOCK DATASET('RAMDAS')

SUBSTITUTE SHEET (RULE 26)
GOCLOS
EQU
EXEC CICS DEQ RESOURCE (MIDASSTE) LENGTH(8) NOHANDLE
BAL R14,SENDCLOS
B RETURN

DOEMAIL
BAL R14,SENDEHEL0
BAL R14,SDMFROM
BAL R14,SDMTO
BAL R14,SDMDATA
SR R15,R15
END TEST

IPAD2BIN
EQU
ST R14,SAVER14A
LA R14,SAVEIPAD
LA R15,NADDRESS
ZAP PLACE='P0'

MADDLOOP
CLI 1(R14),C:'
BE ZAP1
CLI 1(R14),C:'
BE ZAP1
CLI 2(R14),C:'
BE ZAP2
CLI 3(R14),C:'
BE ZAP3
CLI 3(R14),C:'
BE ZAP3
B NOADDR

ZAP1
EQU *
PACK HALFWORD,0(1,R14)
LA R14.2(R14)
B ZAPADDR

ZAP2
EQU *
PACK HALFWORD,0(2,R14)
LA R14.3(R14)
B ZAPADDR

ZAP3
EQU *
PACK HALFWORD,0(3,R14)
LA R14.4(R14)
B ZAPADDR

ZAPADDR
EQU *
ZAP DOUBLE,HALFWORD
CVB R1.DOUBLE
STH R1,HALFWORD

PUTADDIN
EQU *
MVC 0(1,R15),HALFWORD+1
LA R15.1(R15)
AP PLACE='P1'
CP PLACE='P4'
BL MADDLOOP

NOADDR
EQU *
L R14,SAVER14A
BR R14

ROUTINE TO GETMAIN AN AREA TO BE USED TO BUILD MAIL MESSAGE

SPACE 1

SUBSTITUTE SHEET (RULE 26)
GOTMONTH
MVC 8(R2),4(R14) MOVE IN THE MONTH TEXT
MVC 12(R2),DSDATE+6 MOVE IN THE YEAR
MVC 17(R2),DSTIME MOVE IN THE TIME
SPACE 1
AP
L
LA
MVC
MVC
MVC
MVC
MVC
ED
L
LA
LC
LA
BCTR
EX
AH
MVI
L
LA
LC
LA
BCTR
EX
BR
WORKD2, =PL8'1'
R2,DSNXTPOS START OF BUFFER
R2,SUBJ(R2) WHERE TO MESSAGE NUMBER
1(10,R2),DSDATE DATE FOR SUBJECT LINE
12(R2),DSTIME TIME FOR SUBJECT LINE
21(5,R2), =C'Part'
25(4,R2), =X'4021202020' SET MASK
25(4,R2),WORKD2+6 CURRENT MESSAGE NUMBER
R2,DSNXTPOS START OF BUFFER
R2,DEST1@(R2) WHERE TO WRITE DESTINATION
R15,DSDSTLN LENGTH OF EMAIL ADDRESS
R4,DSDEST A(DESTINATION)
R15,0 DECREMENT FOR EXECUTE
R15,MOVEDEST MOVE EMAIL ADDRESS
R2,DSDSTLN LENGTH OF EMAIL ADDRESS
0(R2),C'>' CLOSE RCPT TO FIELD
R2,DSNXDEST START OF BUFFER
R2,DEST2@(R2) WHERE TO WRITE DESTINATION
R15,DSDSTLN LENGTH OF EMAIL ADDRESS
R15,0 DECREMENT FOR EXECUTE
R15,MOVEDEST MOVE EMAIL ADDRESS
R14

CONNHOST
ROUTINE TO CONNECT TO SMTP, NERSP, NERDC PORT 25 VIA TCP/IP
EQU
ST
MVC
MVC
MVC
MVC
CALL
CLC
BE
R1,SAVER14
SOCFUNC, =CL16'SOCKET' SOCKET FUNCTION TO CALL
AF, =F'2' ADDRESSING FAMILY
SOCTYPE, =F'1' SET SOCKET TYPE TO STREAMS
PROTO, =F'0' USE DEFAULT PROTOCOL FOR STREAMS
EZASOKET,(SOCFUNC,AF,
SOCTYPE,PROTO,ERRNO,RETCODE),
VL,MF, = (E, PARMLIST)
RETCODE, = F'-1' ERROR
LEOS
TSO FOREGROUND HARDCORE

MVC SOCRECV. RETCODE+2
CLC NADDRESS = F'0'
BH GOTIPADR

LOOKUP THE IP ADDRESS OF EMAIL HOST (GETHOSTBYNAME)

PACK Double, COMATLEN
CVB R14, DOUBLE
ST R14, TEMPLEN
A R14 = A(TCPBLEN)
ST R14, TEMPLEN
CVD R14, DOUBLE
UNPK COMATLEN, DOUBLE
OI COMATLEN+5, C'0'
CLI COMAPROF, C'Q'
BE BLDRQST
L R12, COMACNTA
MVC TCPBLEN, COMATLEN
B BLDRESP

BLDRQST

SET ADDRESS OF AREA FOR SEND
LA R12, CNTRLBUF
MVC CNTRLBUF, BLANKS
MVC TCPBLEN, COMATLEN
MVC TCPBTRAN, SAVETRAN
MVC TCPBSYST, =C'FED1'
MVI TCPBCOMM, C'0'
MVI TCPBENC, C'0'
MVC TCPBREQS, =C'FLACENTSERV'
MVC TCPBREQN, =C'01'
MVC TCPBMES, =C'01'

BLDRESP

EQU

GETQ4SND

EQU

MVC COMAQLLEN = AL2(MAXSNDLN)
EXEC CICS LINK PROGRAM('RARASOXY') COMMAREA(COMMAREA)
LENGTH(COMACOML)

CLI COMAQERR, C'Y'
BE SOCKERR
LH R14, COMAQLLEN
ST R14, NBYTES
L R15, TEMPLLEN
SR R15, R14

SUBSTITUTE SHEET (RULE 26)
C R15,=F'0' IS THIS THE LAST ONE?
BNE NOTLAST NO, DO NOT SET FLAG
MVI LASTSEND,C'y' SET LAST FLAG
ST R15,TEMPLEN SAVE WHAT IS LEFT
BAL R14,SENDHOST SEND TO THE HOST
LH R14,COMAQTIE GET ITEM
AH R14,=H'1' ADD ONE
STH R14,COMAQTIE PUT NEXT ITEM NUMBER
CLI LASTSEND,C'y' WAS IT THE LAST ONE
BNE GETO4ND NO, GET THE NEXT ONE
BCT R14,ENDLOOP

MOVEEND MVI 1(R15),C'> END BRACKET
BR R1

BLANKR12 EQU MOVE BLANKS TO AREA
ST R1,SR1 |
LA R0,* |
LR R14,R12 |
SR R1,R1 |
ICM R1,8,=C'. |
LH R15,=AL2(MAXSNIDL) |
MVCL R14,R0 |
L R1,SR1 |
BR R1 |

NULLR12 EQU MOVE BLANKS TO AREA
ST R1,SR1 |
LA R0,* |
LR R14,R12 |
SR R1,R1 |
ICM R1,8,=X'00' |
LH R15,=AL2(MAXSNIDL) |
MVCL R14,R0 |
BR R1 |

SNDMTO EQU LENGTH OF MESSAGE
ST R14,SAVER14 |
CLC EMAILTO1,BLANKS |
BNH SNDMTO2 |
BAL R1NULLR12 |
MVC NBYTES,=F'88' |
MVC 0(09,R12),TOMSG |
MVC 09(79,R12),EMAILTO1 |
BAL R1,FINDEND |
MVI DSCRFLAG,C'y' |
BAL R14,SENDHOST |
BAL R14,HOSTRESP |

SNDMTO2 EQU WAIT FOR RESPONSE
CLC EMAILTO2,BLANKS |
BNH SNDMTO3 |
BAL R1.NULLR12 |
MVC NBYTES,=F'88' |
MVC 0(09,R12),TOMSG |
MVC 09(79,R12),EMAILTO2 |
BAL R1,FINDEND |
MVI DSCRFLAG,Y' |
BAL R14,SENDHOST |
BAL R14,HOSTRESP |
SEND THE MESSAGE HEADER AND GET THE RESPONSE FROM THE HOST |
SENDHDR MVI NBYTES,=F'80' |
BAL R14,SENDHOST |
S R10,=F'80' |
SUBTRACT FROM LENGTH
BAL R14,HOSTRESP  
LA R3,80(R3)  
BCT R4,SENDDNR  
SPACE 1

SEND THE REST OF THE MESSAGE WITHOUT GETTING A RESPONSE FROM THE HOST
SENDREST  
EQU  
ST R10,NBYTES  
MVC SOCFUNC,=CL16'SEND'  
MVC FLAGS,=F'0'  
CALL EZACIC04,(R3,NBYTES),VL,MF=(E,PARMLIST) TRANS, TO ASCII  
SPACE  
SCALL EZASOKET,(SOCFUNC,SOCCRCV,FLAGS,  
NBYTES,(R3),ERRNO,RETCODE),  
VL,MF=(E,PARMLIST)

SPACE  
L R1,RETCODE  
CLC R1,=F'-1'  
BE LEOS  
BAL R14,HOSTRESP  
SPACE  
L R14,SAVER14  
SEND QUIT COMMAND TO HOST
SENDQUIT  
EQU  
ST R14,SAVER14  
MVC 0(1,R3),=Q'UIT'  
MVC NBYTES,=F'4'  
MVI DSCRFAG,=F'Y'  
BAL R14,SENDHOST  
BAL R14,HOSTRESP  
L R14,SAVER14  
BR R14

SEND CHUNK OF THE MESSAGE TO THE HOST (ADDTING CRLF)
SENDHOST  
EQU  
ST R14,DSR14SAG  
MVC SOCFUNC,=CL16'SEND'  
MVC FLAGS,=F'0'  
MVC BUF,0(R12)  
TRANSEND  
EQU

TRANSLATE TO ASCII WITH EZACIC04
CALL EZACIC04,(R12,NBYTES),VL,MF=(EPARMLIST) TRAN TO ASCII  
SPACE  
CLI DSCRFAG,=F'Y'  
BNE EZASENDO   
<CR><LF> FLAG SET ?  
SPACE  
L R14,NBYTES  
AR R14,R12  
MVC 0(2,R14),=X'0D0A'  
L R14,NBYTES  
SPACE  
CLC RETCODE,NBYTES  
BNL SENTALL  
CLC RETCODE,=F'-1'  
BNH LEOS  
L R1,RETCODE  
AR R12,R1  
L R1,NBYTES  
S R1,RETCODE

SUBSTITUTE SHEET (RULE 26)
LET R1 R1
BNP R1.NBYTES
ST R1.NBYTES
LH R14.HALFWORD
BCTR R14.0
LTR R14.R14
BNP LEOS
STH R14.HALFWORD
B EZASEND
L R14.DSR14SAV
BR R14

READ RESPONSE FROM MAIL HOST
VL, MF=(E, PARMLIST)
SPACE

AFTSELT
EQU
CLC RSNDSMK, RRETMSK
BE READCALL
CLC RETCODE, =F'0'
BH READCALL
CLC RETCODE, =F'-1'
BE LEOS
B NORESPN

READCALL
MVC SOCKFUNCK, =CL16'READ'
XC INBUF, INBUF
MVC NBTES, =ALL(L'INBUF)
SPACE
CALL EZASOKET, (SOCKFUNC, SOCRECV,
NBYTES, INBUF, INBUF, ERRNO, RETCODE),
VL, MF=(E, PARMLIST)
SPACE
CLC RETCODE, =F'0'
BE NORESPN
BH TRANRESP
CLC RETCODE, =F'-1'
BE LEOS
B NORESPN

TRANRESP
EQU *
CALL EZACICOS, (INBUF, RETCODE), VL, MF=(EPARMLIST) TRAN. EBCDIC

AFTRANS
EQU
BE LEOS
B NORESPN

TRANRES
CALL EZACICOS, ((R12), RETCODE), VL, MF=(E, PARMLIST) TRAN. EBCDIC

AFTRAN
EQU
MVC BUF.0(R12)
NORESP
EQU
MVI NORESPND, C'Y'

HOSTREX
L R14.DSR14SAV
BR R14

ERROR ROUTINES

LEOM
EXEC

GMAINERR
EQU *

MVC LEM010RC, LEM010SG GETMAIN ERROR

SUBSTITUTE SHEET (RULE 26)
MVC MESSAGE(13),=C'GETMAIN ERROR'
B RETURN

LEOL ERROR EQU *
LEOS SOCKERR EQU *

MVC LEM010RC,LEM010LG PROGRAM LOGIC ERROR!
MVC MESSAGE(20),=C 'PROGRAM LOGIC ERROR'!
B RETURN

LEOW QUEERR EQU *

RETURN EQU *
EXEC CICS DEQ RESOURCE(MIDASSTF) LENGTH(8) NOHANDLE
EXEC CICS RETURN

CONSTANTS AND EXECUTED INSTRUCTIONS

COPY RAMIGETP
LTORG

SAVEDEST MVC DSDEST(*-*),0(R4)
MOVEDE1 MVC 0(*-*),0(R4)
MOVELINE MVC 0(*-*),R10),0(R4)
LOOKEXEC TRT 0(*-*),CRTBLL LOOK FOR A <CR> (NOTE: CLOBBERS R2)
MOVEBUF MVC 0(*-*),R12),0(R3)
MVBLANKS MVC 0(*-*),R12),BLANKS
MAXSNRLN MVC EQU 32767

BLANKS DC CL256'
NUMZONE DC CL2800000000000000000000000000000000000000
PIN1111 DC CL41111'
SUBJMSG DC CL80'SUBJECT'
FROMMSG DC CL80'MAIL FROM:<'
TMOSG DC CL80'RCPT TO:<'
HELMOSG DC CL80'HL0 NERAVS, NERDC, UFL, EDU'
HEADER DC CL80'MAIL FROM: <isis@nerav.nerdc.ufl.edu>
RCPPTO DC CL80'RCP TO:<'
DEST1@ EQU (*-HEADER)
DC CL70'
DC CL80'DATA'
TOPHEAD# EQU (*-HEADER)/80
DATE DC CL15 DATE:
DATE@ EQU (*-HEADER)
DC CL65'
DC CL80'From: The FUN system <isis@nerav.nerdc.ufl.edu>"
Mail from the system

Subject: CL40'Subject:
EQU (*-HEADER)
DC CL40'
DC CL15'To:
DEST2@ EQU (*-HEADER)
DC CL65'
DC CL80'
HDR# EQU (*-HEADER)/80
ENDER DC CL80'
DC CL80'QUIT'
DC CL80'!!
ENDERS EQU (*-ENDER)
ENDER# EQU (*-ENDER)/80

*CRTBL DC 25X'00'
ORG CRTBL+13
*X'0D <CR> TO FIND
ORG CRTBL+255

MAX SECONDS DC 'F00000005'
FIONBIO DC 'XL4'8004A77E'
SO_REUSEADDR DC 'F00000004'
SO_KEEPALIVE DC 'F00000008'
SO_BROADCAST DC 'F00000032'
SO_LINGER DC 'F00000128'
SO_OOBINLINE DC 'F00000256'
SO_SNDBUF DC 'F000004097'
SO_ERROR DC 'F000004103'
SO_TYPE DC 'F000004104'

DOWTABLE DS OD
DC 'XL4'0', 'C'Sun,'
DC 'XL4'1', 'C'mon,'
DC 'XL4'2', 'C'Tue,'
DC 'XL4'3', 'C'Wed,'
DC 'XL4'4', 'C'Thu,'
DC 'XL4'5', 'C'Fri,'
DC 'XL4'6', 'C'Sat,'
SUNDAY DC 'XL4'6', 'C'Sat,'
DOW$ EQU "-SATURDAY"
DOW# EQU (*-DOWTABLE)/DOW$

MONTABLE DS OD
DC 'XL4'1', 'C'Jan'
DC 'XL4'2', 'C'Feb'
DC 'XL4'3', 'C'Mar'
DC 'XL4'4', 'C'Apr'
DC 'XL4'5', 'C'May'
DC 'XL4'6', 'C'Jun'
DC 'XL4'7', 'C'Jul'
DC 'XL4'8', 'C'Aug'
DC 'XL4'9', 'C'Sep'
DC 'XL4'10', 'C'Oct'
DC 'XL4'11', 'C'Nov'
DECEMBER DC 'XL4', 'C'Dec'
MON$ EQU "-DECEMBER"
MON# EQU (*MONTABLE)/MON$

*STDDATA DS OH

CTRLBLOCK DC CL('TCPBLEN')'
DC CL8'
DC CL6.000010'
SPROCESA       DC 'PROCESS'
              DC 'VERIFY'

SSTATEKA       DC 'STATEKEY'

SIPADDR        DC 'IPADDR'

SPORTNMA       DC 'PORTNUMB'

SSSTUIDA       DC 'STUID'

SPASSWDA       DC 'PASSWORD'

RRETNCA        EQU 'RSTDDATA'

RRETNMA        EQU 'RSTDDATA'

RDATALEN       EQU 'RSTDDATA'

END            RAM1S00
APPENDIX H

DATA SET RAM00

1. RAM0400 FILE RECORD SECT
   ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT MECHANISM AND
   PROTOCOLS, ARE COPYRIGHT 1998 BY THE UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED.
   COPY RARARGDA
   USING RARARGDA,R2
   COPY RARAMMEN
   USING REARARMMEN,R12
   COPY RAMCOMM
   USING COMMAREA,R5
   COPY RARATCPB
   COPY RARAEXLG

2. WORKING STORAGE
   COPY DFHAIID 3270 AID CHARACTERS
   COPY RAREGS REGISTER EQUATES
   EJECT
   DFHEISTG DSECT
   SCRATCH PAD AREA
   COPY RARASPAD
   FIELDS FOR RAMIGETD
   FINDFLD DS CL8
   DINDFDRT DS CL79
   UPDATE DS C

   SAVER12 DS F
   TEMP9 DS CL6
   TEMP06 DS CL6
   RGFLUX DS CL2
   RGAPPT DS C
   RGDAYO DS C
   PHASE DS C
   BADSSNF DS C
   NOSSNF DS C
   CREDITS DS CL2
   CTLKEY DS CL3
   APPT DS CL2
   TIMEAP DS CL2
   DADATE DS 0CL9
   DADUMM DS CL1
   DAMMDD DS 0CL4
   DAMM DS CL2
   DADD DS CL2
   DATEIME DS CL4
   TIMEALL DS OCL5 BHHMM
   TIMEBLNK DS C
   TIMEHH DS CL2
   TIMEMM DS CL2
   SYSSMDD DS CL4
   PTIME DS PL4
   PSTIME DS PL4

SUBSTITUTE SHEET (RULE 26)
ADD24 DS PL4
PRTIME DS PL4
SYSTIME DS QCL6
SYSHMM DS OCL4
SYSHH DS CL2
SYSSM DS CL2
SYSS DS CL2
REGMTH DS CL3
REGDATE DS 0CL4
REGMM DS CL2
REGDD DS CL2
REGTIME DS CL4
PACK3 DS PL3
DIVIDEND DS OD
QUOTIENT DS F
REMAINS DS F
DIVISOR DS F
SAVETRID DS F
SAVER6 DS F
SAVESEL DS F
SAVEVAR DS F
SAVEMVLN DS H
TEMPAREA DS CL30
SAVESTID DS CL20
SAVEPSWD DS CL10
SAVEPASS DS CL4

EJECT

PROGRAM RAM0400 STARTS HERE

RAM0400 DFHEIENT CODERE=(3,8)
RAM0400 AMODE 31
RAM0400 RMODE ANY

L R5,DFHEICAP
CLC EIBCALEN,=H'0'
BE RETURN

EXEC CICS HANDLE CONDITION DISIDERR(NEW)
ERROR(NEW) TERMIDERR(NEW) SYSIDERR(NEW)
ISINVREQ(NEW) INVREQ(NEW) IOERR(NEW)
DISABLED(NEW) ILLOGIC(NEW) LENGERR(NEW)
NOTAUTH(NEW)

EXEC CICS HANDLE CONDITION NOTFND(NOTEND) NOTOPEN(NOTOPEN)

START EQU *

XC NUMCHEC,NUMCHECK
MVC COMAFNDF,=C'MDASPSWD' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT

XC SAVEPSWD,SAVEPSWD INIT FIELD

LH R14,H'10' LENGTH OF DATA
LA R15,SAVEPSWD PUT DATA HERE
MVC COMAFNDF,=CMDASPSW2' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT
CLC SAVEPSWD,BLANKS
BNH NO2ND
CLC SAVEPSWD,COMASELC
BNE NOTSAME
MVI COMASELC,+4,C'.'
MVC COMASELC,+5(4),SAVEPSWD
NO2ND
  MVC SAVESEL.C,COMASEL.C
  MVC COMAIBFN,BLANKS

  EXEC CICS LINK PROGRAM('RAMI1S00') COMMAREA(COMMAREA)
  LENGTH(COMACOML)
  CLI COMAERRF,'Y'
  BE RETURN
  CLI COMAIBFN, 'Y'
  GH RETURN

  MVC COMAIBFN, 'Y'
  GH GETRCOD

  MVC COMAIBFN, 'Y'
  GH PUTERROR

NOTSAME
  EQU *
  MVC COMAIBFN, 'Y'
  GH RETURN

BAD
  EQU *
  MVC COMAIBFN, 'Y'
  GH PUTERROR

IBFN16
  EQU *
  MVC COMAIBFN, 'Y'
  GH PUTERROR

FORCE
  MVC COMAIBFN, 'Y'
  MVC COMAPASS,SAVESEL.C
  GH RETURN

ERROR
  EQU *
  NOTOPEN
  EQU *
  NOTFND
  EQU *
  GH PUTERROR

PUTERROR
  EQU *
  MVC 'COMAMESS(8)=C'CONTINUE''
  MVI COMAMFRRF, 'Y'
  GH RETURN

PINOK
  EQU *
  MVC COMAUID(9),COMAEXSS
  MVC COMATOPC,COMATARG
  MVC COMAEXSS+3(6),SAVESTID+13

  CLC COMAEXSS,BLANKS
  BH CKSSNNUM
  MVI NOSSNF, 'Y'
  GH READMMEN

  CKSSNNUM
  MVC NUMCHEC,COMAEXSS
  CLC NUMCHEC(9),NUMZONE
  BE SSNOK
  MVI BAEDSSNF, 'Y'
  GH READMMEN

  SSNOK
  EQU *
  MVC COMASELC(4),SAVESTID+3

  CKFORCE
  CLC COMADATA(5),='FORCE'
  BNE DOPIN

  MVI COMASELC+4,'Y'
  MVC COMASELC+5(4),SAVESTID+3
LH R14,=H'4' LENGTH OF DATA
LA R15,SAVEPASS PUT DATA HERE
MVC COMAFNDF,=C'PASSWORD' DATA IDENTIFIER
BAL R4,GETDATA GO GET IT
CLC SAVEPASS,BLANKS
BH DOPIN
MVC COMPASS,SAVEPASS

DO PIN ROUTINE
DOPIN

    EQU *
    MVC SAVETRID,EIBTRNID
    MVC IEBTRNID,=C'RA56'
    MVC COMASAVP,=C'RAMI0500'
    EXEC CICS LINK PROGRAM('RAGA3000') COMMAREA(COMMAREA)
    LENGTH(COMACOML)
    MVC EIBTRNID,SAVETRID

    CLC COMASELC(5)M=C'+0050 EXPIRED?
    BNE PUTFIX
    MVI COMASELC+4MC'1' EXPIRED?
    PUTFIX

    MVI COMASELC,C'0'
    ←------TEMP FIX!!!!!

    MVC COMARCOD(5)MCINASELC
    MVC COMARMES,BLANKS

    B RETURN

*GOBACK EQU *
RETURN EXEC CICS RETURN

PUTDATAE MVC 0(('*',R12),0(R6)
BUMPADD LA R12,('•')(R12)

COPY RAMIGETP
NUMZONE DC 12C'0' ZONES
BLANKS DC CL120 '
MAXWORK DC HL22000;
SPACE EJECT
LTORG RETURN TO CICS

END RAMI0400

*****************************************************************************

EROBBIN
S3030

JOBID
JOB NAME:
USERID
SYSOUT CLASS:
OUTPUT GROUP:
TITLE:
DESTINATION:
NAME:

TSU09089
EROBBIN
EROBBIN
A
4 .00001 .00001
UFG3820
ROBBINS, EARL

SUBSTITUTE SHEET (RULE 26)
EROBBIN
S303

JOBID: TSU0989
JOB NAME: EROBBIN
USERID: EROBBIN
SYSCOUT CLASS: A
OUTPUT GROUP: 5
TITLE:

DESTINATION: UFG3820
NAME: ROBBINS, EARL
ROOM
BUILDING:
DEPARTMENT:
ADDRESS:

PRINT TIME: 11:37:24 AM
PRINT DATE: 11 AUG 1998
PRINTER: PRT6
SYSTEM ID: NER1

TSO FOREGROUND HARDCOPY

DSNAME=RA.PATENT (RAMI0600)

DATA SET RAMI0600 00010000

ALL INFORMATION IN THIS DOCUMENT, INCLUDING THE TRANSPORT
MECHANISM AND PROTOCOLS, ARE COPYRIGHT 1998 BY THE
UNIVERSITY OF FLORIDA. ALL RIGHTS RESERVED.

COPY RARAMDAS 00030006
COPY RARAMDAS,R9 00030105
COPY RARARGDA 00030205
COPY RARARGDA,R12 00030306
SPACE 00030406
SPACE 00031000
SPACE 00040000
SPACE 00041000
SPACE 00042000
SPACE 00050000
SPACE 00060000
SPACE 00070000
SPACE 00080000
SPACE 00090000
SPACE 00100000
SPACE 00110000
SPACE 00120000
SPACE 00130000
SPACE 00140000
SPACE 00150000
SPACE 00170000

2. WORKING STORAGE

COPY DFHAID 3270 AID CHARACTERS
SPACE REGISTER EQUATES
DFHEISTG COMMAREA DSECT
SPACE SCRATCH PAD AREA

TMPKEY DS 0C16
TMPTIME DS PL8

SUBSTITUTE SHEET (RULE 26)
TMPMDASK DS CL8
LINE DS CL80
MIDASSTE DS CL8
COUNT DS PL3
ABCODE DS CL4
COPY RARASPAD
EJECT

RAM0600 DFHIENT CODEREG=(3,8)
RAM0600 AMODE 31
RAM0600 RMODE ANY
SPACE

REQUIREMENT CHECKING AND EVALUATION ROUTINES

EXEC CICS HANDLE CONDITION DSIDERR(EROOR)
ERROR(EROOR) TERMIDERR(TERMIDERR) SYSIDERR(EROOR)
ISCINVREQ(EROOR) INVREQ(EROOR) IOERR(EROOR)
DISABLED(EROOR) ILLOGIC(EROOR) LENERR(EROOR)
NOTAUTH(EROOR) DUPREC(EROOR)

EXEC CICS HANDLE ABEND LABEL (HNDABND)

EXEC CICS HANDLE CONDITION NOTOPEN(NOTOPEN)
MVC MDSASKEYY,TMPMDASK
MVT MDSASKEYY,C'0'
EXEC CICS WRITE FILE('RAMDAS') FROM(RARAMDAS)
LENGTH(HALFWORD) RIDFLD(MDSASKEYY)

AP COUNT,'=P'1'

----DEQ RESOURCE----

EXEC CICS DEQ RESOURCE(MIDASSTE) LENGTH(8)
B NEXT

ERROR EQU *
MVC LINE(41)=C'THERE WAS AN ERROR PROCESSING THE REQUEST'
SENDLIN
TRMIDERR EQU
B RETURN

NOTOPEN EQU
MVC LINE(38)=0TATE FILE OR ALTERNATE INDEX NOT OPEN
SENDLINE

NOTFND EQU *
MVC LINE(30)=C'REQUESTED RECORD WAS NOT FOUND'
SENDLINE

STARTAGN EQU *
EXEC CICS START TRANSID('MI06')

ALLDONE EQU
MVC LINE,MESSMII06
UNPK LINE+39(3),COUNT
OI LINE+41,C'0'

SENDLINE EQU
EXEC CICS HANDLE CONDITION INVREQ(RETURN) NOTALLO C(RETURN)
EXEC CICS SEND FROM(LINE)
RETURN
EQU *
EXEC CICS DEQ RESOURCE(MIDASSTE) LENGTH(8) NOHANDLE
RETURN2
EQU *
HNDABND
EQU *
EXEC CICS ASSIGN ABCODE(ABCODE)
EXEC CICS ABEND ABCODE(ABOCED) NODUMP
B RETURN
BLANKS
DC CL80'
MESSM06DCCL80'MI06 COMPLETED SUCCESSFULLY, PROCESSED RECORDS
LTORG
END RAMI0600

SUBSTITUTE SHEET (RULE 26)
APPENDIX J

nirvana.c—EAGLE transfer client for Web server.
Written by Michael Lucas for the University of Florida
November 11, 1998 http://www.req.ufl.edu

Incorporates client.c by Tom Kelliher and getcgvars.c from NCSA
Ver 1.1.6 build 0 Robustified(tm)

Program flow

This program is started by a call from a Web page which contains a form. Data is passed via a CGI string which is then parsed via the standard NCSA getcgvars.c program into a tow-dimensional array of strings: name of the filed paired with the value for the field. (Note that the program must be compiled in a directory where getcgvars.c exists, or is in the path.)

This program then sends a message to the appropriate CICS server and port (both defined near the top of this program) to start the transaction MI00.

When CICS receives this call it responds with OKTOSEND to indicate readiness to receive data. This program then sends all the CGI date in the form
  8-byte-field-name field-data carriage-return
without the spaces (eg, MYFIELDNTHISISTHEDATAIn).

When CICS has received the data it then does magic and returns a 50-byte data descriptor and a fully-marked-up Web page. This program parses the stream and writes the page to standard out, where it is picked up by the CGI form and sent as a Web page to the browser that originally called it.

RECENT CHANGE LIST
11/03/1998 ver 1.1.4 "Robustified!"
  Repaired code that sent garbage to Web browser on dead socket call, causing random text boxes or garbage to appear
APPENDIX J

11/17/1998 ver 1.1.5
  Added alarms piped to pdie, to prevent child processes forked by the SP from running forever if they fail to receive the needed information from the data client (e.g. CICS abend or online application hork by an AOL user).

11/19/1998 ver 1.1.6
  Modified die and pdie to accept the socket address so they can close the socket on err. Modified all calling functions pass sock to die and pdie.

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>

SUBSTITUTE SHEET (RULE 26)
```c
#include <stdio.h>
#include <signal.h>
#include "getgivar.c"

#define SERVER_PORT 9999
#define BUFFER_SIZE 10240
#define MAX_STRING_LENGTH 20480
#define CTL_BUFFER_SIZE 60
#define UF_CICS "mvs your site"
#define NUMBER_OF_XFER_ATTEMPTS 5
#define TRANS_TYPE "M100"
#define DIAG_MSG_SIZE 8
#define OK_MSG "GOTDATAOK"
#define SPACES "

prototypes

void die (const char *, int);
void die (const char *, int);
void print_top(void);
void zero_pad(int, char *);

main

int main (int argc, char *argv[])
{
    int sock;
    struct sockaddr_in server;
    struct sockaddr_in client;
    in clientLen;
    struct hostent *hp;
    char buf [BUFFER SIZE];
    char html_out[MAX STRING LENGTH];
    register int l;
    int byte_out = 0;
    int byte_in = 0;
    ()
    int count = 1
    int byte_tct = 0;
    register int rec_size = 0;
    register int rec_size = 0;
    char **cgvars;
    char output[CTL BUFFER SIZE];
    char output[BUFFER SIZE];
    char zero_stg[6];
    int var_size = 0;
    char holder[25];
    
    fd for socket connection
    Socket info. for server
    Socket info. about us
    Length of client socket struct.
    Return value from gethostbyname()
    Received data buffer
    Build string for output
    loop counter
    Total bytes sent on a socket write
    Total bytes received on a socket read
    miscellaneous counter
    Data written to the socket
    Data written to the socket
    Holds the length of the cgi variable

    Install the signalling system to abort processing in case of error.
    signal(SIGALRM, die("No response from data client. Nirvana terminating.", sock));
```
alarm(600);

Parse the cgi string and pull out a 2d array of data.

cgivars = getcgvars();

Open a TCP socket and send message.
if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
  pdie("Couldn’t open stream socket", sock);

Prepare to connect to server.

bzero((char *) &server, sizeof(server));
server.sin_family = AF_INET;
if ((hp = gethostbyname(UF_CICS)) == NULL)

  {
    printf("%s: unknown host\n", UF_CICS);
    die(buf, sock);
  }

bcopy(hp->h_addr, &server.sin_addr, hp->h_length);
server.sin_port = htons((u_short) SERVER_PORT);

Try to connect.

if (connect(sock, (struct sockaddr *) &server, sizeof(server)), 0)
  pdie("Couldn’t connect stream socket", sock);

Determine what prot client’s using.

clientLen = sizeof(client);
if(getsockname() overwrote name structure", sock);

Prepare to send to CICS to kick off the transaction.

if (byte_out = write(sock, TRANS_TYPE, sizeof(TRANS_TYPE)) < 0)
  pdie("Couldn’t write TRANS_TYPE on stream socket", sock);

Check that CICS is happy and ready to receive data.

if (byte_in = read(sock, buf, DIAG_MSG_SIZE))    // If we got data

  {
    if(stemp(buf,"OKTOSEND")
      Check for ready
      die("CICS did not reply ready.", sock);
      These brackets are needed to define the scope of the IF ELSE
      condition.

else
  pdie("No answer from socket reading ready message from CICS.",
       sock);

Start formatting the HTML page.

print_top();

Flush the strings.

bzero(html_out, sizeof(html_out));
bzero(outbuf2, sizeof(outbuf2));
Copy the Web page variables to the output string in this format:

8 spaces, variable length(6), variable name, variable data, newline.
Copy until out of data.

```c
for (i=0; cgivars[i]; i++)
{
    bzero (holder, sizeof(holder));

    Pull any button data

    if (!strcmp(cgivars[i], "MIDASBTN"))
        i += 2;
    var_size = strlen(cgivars[i + 1]);
    zero_pad(var_size, zero_stg);
    strcat(outbuf2, zero_stg);
    sprintf(holder, "%%d", var_size);
    strcat(outbuf2, holder);
    strcat(outbuf2, cgivars[i + 1]);
    strcat(outbuf2, "\n");
}

Move the length info to the beginning of the output stream.
It will be the length of outbuf PLUS the information itself.

rec_size = (CTL_BUFFER_SIZE + strlen(outbuf2));

zero_pad(rec_size, zero_stg);

sprintf(html_out,"%s%d", zero_stg_rec_size);
strcat(html_out,"
0 1 2 3 4 5");

Add the output buffer to the transmission string.
strcat(html_out, outbuf2);

Write html_out to the socket.

count = 1;
byte_tot = 0;
while (byte_tot < rec_size)
{
    if (!write(sock, html_out, rec_size)))
        count += 1;
    if (byte_out < 0)
        perror("Exceeded number of write attempts on socket", sock);
    byte_tot += byte_out;
}

Clear all buffers and counters to prepare to read CICS response.
count = 1;
rec_size = 0;
byte_tot = 0;
bzero(buf, sizeof(buf));
bzero(outbuf2, sizeof(html_out));

Read the CICS response from the socket.

if (! (byte_in = read(sock, buf, CTL_BUFFER_SIZE)))
{
    if null read, increment bad read counter and test for excess.
    count ++ ;
    if (count == NUMBER_OF_XFER_ATTEMPTS)
        pdie("Excess number of failed reads on initial socket read. Check

communications", sock);
} else
    We got data on the read, continue
    { Move the first 6 bytes of buf into outbuf2
    strncpy(outbuf2,buf,6);
    rec_size = atoi(outbuf2);
    byte_tot = rec_size - byte_in;
    if (byte_in < 0)
        pdie(buf, sock);
    Skip the control info and copy the remainder to the Web output buffer.
    for(i = 0; i < byte_in; i ++ )
        printf("%c", buf[(i + CTL_BUFFER_SIZE)];

    If we didn't get the entire stream in the first pass, loop until we do.
    while (byte_tot > 0)
    {
        bzero(buf, sizeof(buf));
        byte_in = read(sock, buf, byte_tot);
        if (byte_in < 0)
            pdie("Died while in the socket reading loop", sock);
        if(count == NUMBER_OF_XFER_ATTEMPTS)
            pdie("Exceeded number of dead reads on loop socket read",
        sock);
        if(byte_in == 0)  
            If dead read on socket
            count ++ ;
            increment bad-read counter
    } else
    { byte_tot - byte_in;
Send but to webserver
Reset bad-read counter

Send confirmation back to CICS

if(byte_tot <= rec_size) which will be true if
we got the data
    if ( 1 (byte_out = write(sock, OK_MSG, sizeof(OK_MSG))))
        pdie("Died while confirming transmission received",sock);

Write out the end-html string to the Web server
printf("</body></html>");
    Close this connection.
    close(sock);

End of main

pdie — Call perror() to figure out what's going on and die.

void pdie(const char *msg, int sock)
{
    printf("<docy bgcolor=cc0000 text = ffffff><title>Error Report</title><n>");
    printf("<h3><center>Application Error Report </h3><hr><n>");
    printf("<br><br>Problem: %s. Check your data and try
reloading. <br><br><n", msg);
    perror(msg);
    printf("<hr></center></body></html>");
    close(sock);
    exit(1);
}

die —- Print a message and die.

void die(const char *msg, int sock)
{
    printf("<br><br>Error! : %s. Try reloading, or contact support.<n", msg);
    fputs(msg, stderr);
    fputc('n', stderr);
    close(sock);
    exit(1);
}
print_top --- Start output of the HTML page

void print_top(void)
{
    printf("Content-type: text/html\n\n");
    printf("<html>\n")
    printf("<! All information in this document, including the transport>\n");
    printf("<! mechanism and protocols, are copyright 1998 by the >\n");
    printf("<! University of Florida. All rights reserved. >\n");

    Uncomment the following line to enable no-cache for production.

    printf(".META HTTP-EQUIV = " Pragma="Content="no-cache="/i\n");

    zero_pad --- Adds the leading zeros to the data length description

void zero_pad(int rec_size, char *zero_stg)
{
    bzero(zero_stg, sizeof(zero_stg));

    if (rec_size <= 9)
        strcpy(zero_stg, "00000");
    else if (rec_size <= 99)
        strcpy(zero_stg, "0000");
    else if (rec_size <= 999)
        strcpy(zero_stg, "000");
    else if (rec_size <= 9999)
        strcpy(zero_stg, "00" );
    else if (rec_size <= 99999)
        strcpy(zero_stg, "0" );
}
What is claimed is:

1. A method of providing Web access to data, the steps comprising:
   providing a Web server for distribution of data to users;
   providing a database on a database computer operably connected to the Web server, the database having a database program that provides data in a given format;
   upon a user requesting data in the database, the Web server forwarding the request to the database, the database program accessing the data in the given format, the database computer running a Web control program that generates a Web page with the requested data and supplies the generated Web page to the user.

2. The method of Claim 1 wherein the Web control program marries file definition objects and page definition objects to generate Web pages.

3. The method of Claim 1 wherein the Web server is on a Web server computer different from the database computer.

4. The method of Claim 3 wherein the database computer is a mainframe computer.
5. The method of Claim 4 wherein the mainframe computer supplies the generated Web page to the user via the Web server computer.

6. The method of Claim 4 wherein, upon a user requesting data not in the mainframe computer, the Web control program causes the mainframe computer to access data on a remote computer and the Web control program then generates a Web page with the requested data and supplies the generated Web page to the user.

7. The method of Claim 4 wherein, responsive to a user, the Web control program calls a subroutine on the mainframe computer, which subroutine is independent of the Web control program and is a legacy subroutine.

8. The method of Claim 4 further including the steps of: having a user supply an identification code, forwarding the identification code from the Web server computer to the mainframe computer, the Web control program then generating a state key, incorporating the state key into a Web page supplied to that user, and wherein the mainframe computer detects the state key from the Web page upon the user requesting data and decides whether that user can access the requested data based on the state key.
9. The method of Claim 8 wherein the state key is randomly generated for a given access session and the state key loses its ability to authorize data access if it is not sent by the user to the mainframe computer at least once during a time-out interval.

10. The method of Claim 4 wherein the Web control program receives a page change command from a user using a Web browser and, responsive to the page change command, the Web control program changes a given Web page to a changed Web page such that any users accessing the given Web page now receive the changed Web page, the page change command changing the appearance of a given Web page.

11. The method of Claim 1 further including the steps of: having a user supply an identification code, the Web control program then generating a state key, incorporating the state key into a Web page supplied to that user, and wherein the database computer detects the state key from the Web page upon the user requesting data and decides whether that user can access the requested data based on the state key.

12. The method of Claim 1 wherein the Web control program receives a page change command from a user using a Web browser and, responsive to the page change command, the Web control program changes a given Web page to a
changed Web page such that any users accessing the given
Web page now receive the changed Web page, the page
change command changing the appearance of a given Web
page.

13. A method of providing Web access to data,
the steps comprising:
 providing a Web server for distribution of data to
users;
 providing a database on a database computer operably
connected to the Web server, the database having a
database program that provides data in a given
format;
 providing a Web control program on the database
computer;
 having a user supply an identification code to the
Web server, the Web control program then generating
a state key and incorporating the state key into a
Web page supplied to that user, and wherein the
database computer detects the state key from the Web
page upon the user requesting data and decides
whether that user can access the requested data
based on the state key.

14. The method of Claim 13 wherein the state key
loses its ability to authorize data access if it is not
sent by the user to the database computer at least once
during a time-out interval.
15. The method of Claim 14 wherein the Web server is on a Web server computer different from the database computer.

16. The method of Claim 15 further including the step of forwarding the identification code from the Web server to the database computer.

17. The method of Claim 16 wherein the database computer is a mainframe computer.

18. The method of Claim 17 wherein the state key is randomly generated for a given access session.

19. The method of Claim 13 wherein the state key is randomly generated for a given access session.

20. A method of managing a Web site, the steps comprising:

providing a Web control program on a computer;

having the Web control program receive a page change command from a user using a Web browser to access the Web site; and,

responsive to the page change command, the Web control program changing a given Web page to a changed Web page such that any users accessing the given Web page now receive the changed Web page, the
page change command changing the appearance of a given Web page.

21. The method of Claim 20 wherein the Web control program marries file definition objects and page definition objects to generate Web pages.

22. The method of Claim 21 wherein the computer having the Web control program is a mainframe computer and users request data on the mainframe computer via a Web server on a Web server computer.

23. The method of Claim 22 wherein the Web control program generates HTML through a completely table driven process, independent of file definitions and page definitions.

24. The method of Claim 23 further including the steps of: having a user supply an identification code, forwarding the identification code from the Web server computer to the mainframe computer, the Web control program then generating a state key, incorporating the state key into a Web page supplied to that user, and wherein the mainframe computer detects the state key from the Web page upon the user requesting data and decides whether that user can access the requested data based on the state key.
25. The method of Claim 22 further including the steps of:

- providing a database on the mainframe computer, the database having a database program that provides data in a given format;
- upon a user requesting data in the database, the Web server forwarding the request to the database, the database program accessing the data in the given format, and, by operation of the Web control program, generating a Web page with the requested data and supplying the generated Web page to the user.
FIG. 3

VERIFY STUDENT ID
ENTER SOCIAL SECURITY NO.
ENTER YOUR PIN

FIG. 4

YOU HAVE GRADES FOR 8 CREDITS

<table>
<thead>
<tr>
<th>COURSE</th>
<th>CREDIT</th>
<th>GRADE</th>
<th>HIDDEN STATE KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGC2021C</td>
<td>04</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>ECO3100</td>
<td>04</td>
<td>C</td>
<td></td>
</tr>
</tbody>
</table>
FIG. 6

READ PRIMARY FUNCTION TABLE

MOVE IN INDICATED PRIMARY FUNCTION TABLE KEY

PROTECTED PAGE?

PRE-FUNCTION CHECKS?

CONTINUE

PERFORM PRE-FUNCTION CHECKS

PASSED ALL CHECKS?

ERROR

YES

NO

YES

NO
INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
  IPC(7) : G06F 15/00, 17/30
  US Cl. : 707/9, 10; 709/202, 218

According to International Patent Classification (IPC) or to both national classification and IPC.

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
  U.S.: 707/9, 10; 709/202, 218, 200, 203, 217, 219

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched.

Electronic database consulted during the international search (name of database and, where practicable, search terms used)
  EAST, WEST

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category*</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>US 5,793,964 A (ROGERS et al) 11 August 1998, see Figures 9-11, column 3, line 20- column 7, line 53, column 9, line 26- columns 10, line 43, column 17, line 49- column 18, line 60.</td>
<td>1-25</td>
</tr>
<tr>
<td>X, P</td>
<td>US 5,974,441 A (ROGERS et al) 26 October 1999, see the abstract, Figures 9-11, column 19, line 50- column 20, line 62.</td>
<td>1-25</td>
</tr>
<tr>
<td>A, P</td>
<td>US 5,926,180 A (SHIMAMURA) 20 July 1999, see the whole document.</td>
<td>1-25</td>
</tr>
<tr>
<td>A</td>
<td>US 5,715,453 A (STEWART) 03 February 1998, see the whole document.</td>
<td>1-25</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.  
See patent family annex.

* Special category of cited document
  "A": document defining the general state of the art which is not considered to be of particular relevance
  "E": earlier document published on or after the international filing date
  "L": document which may throw doubts on priority claim(s) or which is said to establish the publication date of another claim or other special reason (as specified)
  "C": document referring to an oral discussion, use, exhibition or other means
  "T": document published prior to the international filing date but later than the priority date claimed

Date of the actual completion of the international search: 16 MARCH 2000

Date of mailing of the international search report: 12 APR 2000

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
UYEN LE
Telephone No. (703) 305-4134

Form PCT/ISA/210 (second sheet) (July 1998)