HostBridge and BizTalk: Integrating CICS with eBusiness Processes

---

A HostBridge™ White Paper

Toll-free: (866) 965-2427
Email: info@hostbridge.com

Copyright Notice

Revision date: 6/13/2002

Trademarks

Table of Contents

List of Figures and Screen Shots

HostBridge and BizTalk: Integrating CICS with eBusiness Processes

Overview: BizTalk, CICS and HostBridge

Microsoft's BizTalk Server 2000 enables you to build and deploy integrated business processes within your organization and with your trading partners. It includes a suite of tools and services that make building business processes and integrating applications faster. Organizations can quickly implement secure, reliable trading partner relationships independent of operating system, programming model, or programming language.

BizTalk represents an emerging category of eBusiness integration software based upon the use of XML. However, the existing category of mainframe CICS applications certainly is not dead. In fact, according to statistics from IBM and others, CICS has never been more successful:

- 30 years and $1 trillion (per IDC) invested in CICS applications
- 14,000+ CICS customers worldwide
- 20,000+ CICS/390 licenses worldwide
- CICS is used by 490+ of IBM's top 500 customers
- 30 million end users of CICS applications
- 150,000+ concurrent users/system
- 5,000 CICS software packages from 2,000 ISVs
- 950,000 programmers earn their living from CICS
- CICS handles >30 billion transactions/day valued at >$1 trillion/week

For companies with large investments in mainframe CICS applications, the ability to integrate these applications with products like BizTalk is imperative.

HostBridge is a patent pending software product that XML-enables a broad class of existing CICS applications. HostBridge does this without requiring modification to the existing applications, and without screen-scraping. As a result, HostBridge is an ideal tool for integrating CICS applications with BizTalk.

This White Paper presents a case study on how you can use HostBridge to integrate existing CICS applications with BizTalk.

Types of CICS Applications

Not all CICS transactions operate the same way. As a result, the integration approach will depend on how the CICS transaction operates. The following diagram shows a high-level taxonomy of CICS transactions.

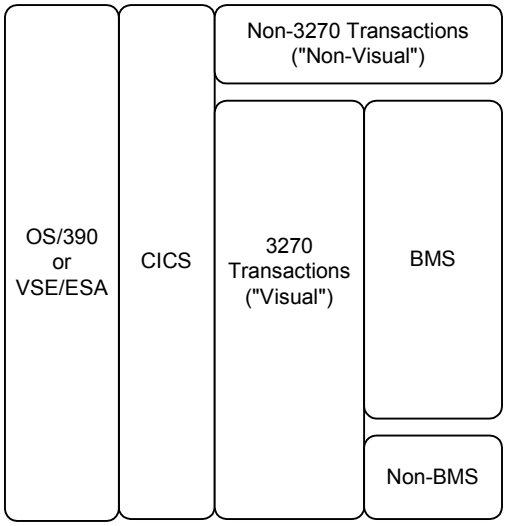


Figure 1. CICS Application Access Taxonomy

"Visual" vs. "Non-Visual" Transactions

CICS transactions fall into two broad categories: "visual" and "non-visual." A "visual" transaction is one that expresses a presentation interface to an end-user at a terminal. You could also refer to a "visual" transaction as a "terminal-oriented" transaction. In contrast, "non-visual" transactions do **not** interact with an end-user. Instead, another program invokes these transactions. (This type of transaction is also referred to as "COMMAREA transaction" because the input/output parameters are passed to/from the transaction using an area of storage referred to as the "communication area," or COMMAREA.)

The distinction between non-visual and visual transactions is important because integration possibilities exist for non-visual transactions that do not exist for visual transactions. As you might imagine, non-visual transactions are far easier to integrate with other programs than are visual transactions. However, visual transactions are far more common than non-visual transactions.

Understanding CICS "Visual" Transactions

In order to understand the integration possibilities for visual transactions, we need to further define this category.

CICS application developers have always had a number of choices in how to design their transaction to interact with an end-user at a terminal. The majority of applications use a component of CICS called Basic Mapping Support (BMS). BMS essentially handles the presentation logic of the transaction and relieves the application developer from having to encode and decode 3270 terminal data streams. The minority of applications that do not

use BMS either include code to process 3270 data streams, or rely upon a non-IBM solution to handle presentation logic.

The distinction between BMS and non-BMS applications is important because integration possibilities exist for BMS applications that do not exist for non-BMS applications. For non-BMS applications, integration is based upon the terminal-oriented data stream generated by the application. As described elsewhere, this approach can have serious limitations.

What is HostBridge?

HostBridge is a patent pending software product that XML-enables existing CICS transactions. HostBridge runs under CICS on the mainframe. With HostBridge, any CICS transaction can be XML-enabled automatically. HostBridge does this without requiring modification to your existing applications, and without screen-scraping. As a result, HostBridge is the perfect tool for integrating CICS applications with BizTalk.

Integration Architecture

BizTalk provides multiple ways to develop eBusiness solutions that integrate with external applications. One approach is to control all aspects of the interaction with the external application within the BizTalk process or "schedule." Another approach is for BizTalk to rely upon an application "adapter" that handles the detailed interaction with the external application. Technically, such an adapter is a request object that integrates with BizTalk.

Our experience with BizTalk suggests that, for optimum performance, the use of an application adapter is preferable to controlling application integration within the BizTalk schedule.

The basic architecture of this approach looks like this:

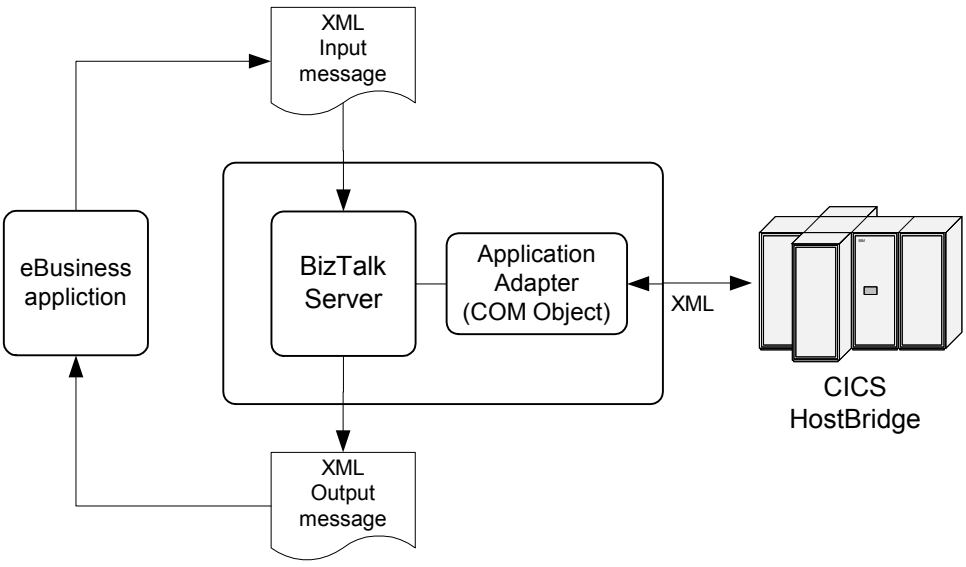


Figure 2. HostBridge/BizTalk Solution Architecture

The eBusiness application initiates the process by sending an XML message to the BizTalk server. The message triggers the activation of a BizTalk schedule. As part of the processing directed by the schedule, BizTalk calls the

application adapter. The adapter interacts with the CICS transaction through HostBridge (HostBridge runs within CICS on the mainframe). Communication between BizTalk, the adapter, and HostBridge continues until the required CICS data elements are collected. The BizTalk schedule then sends the output message back to the originating application.

Sample Application

The sample application described in this section illustrates how to integrate BizTalk and CICS using HostBridge and an application adapter. See Appendix A for a description of the CICS transaction used in this sample application.

BizTalk Process and Data Flow

The following BizTalk schedule controls this sample application:

1. A message flows into a BizTalk channel that activates the schedule.
2. The message contains a quote request that BizTalk sends to a request object
3. The request object runs a series of CICS transactions using HostBridge and returns the requested data to BizTalk.
4. BizTalk sends the data as output to the application that requested it.

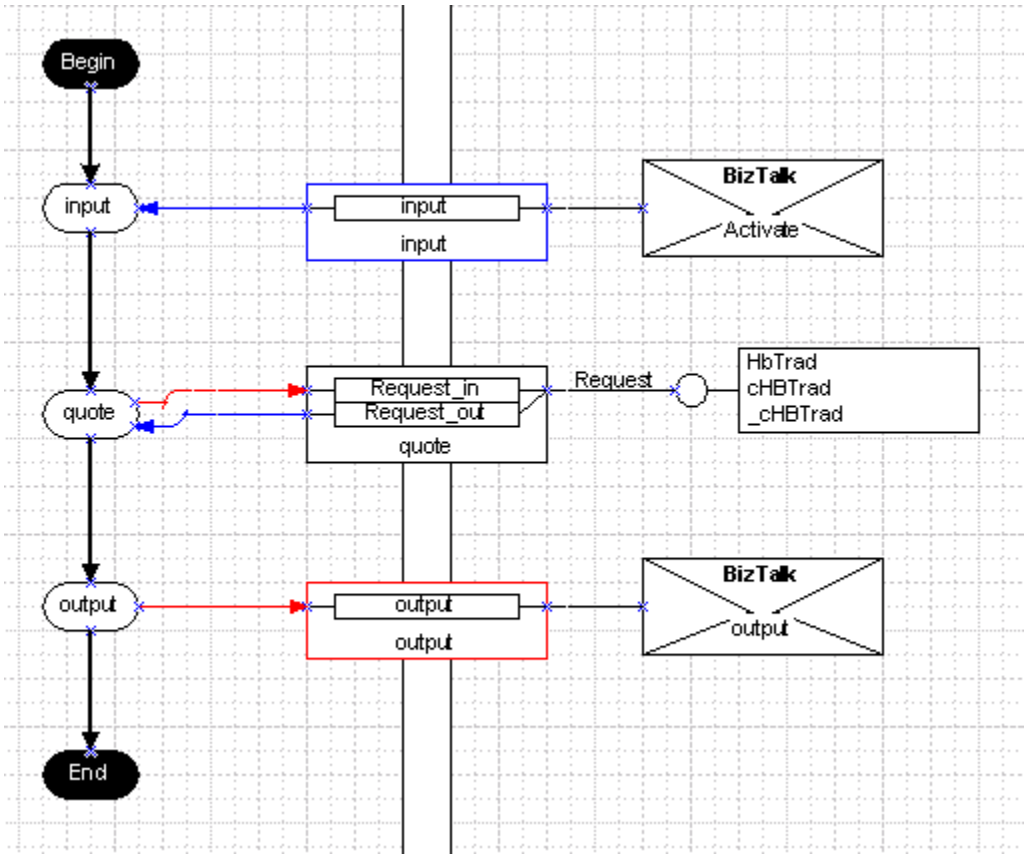


Figure 3. BizTalk Schedule for Sample Application

The following diagram shows the data flow BizTalk uses to carry out the process above.

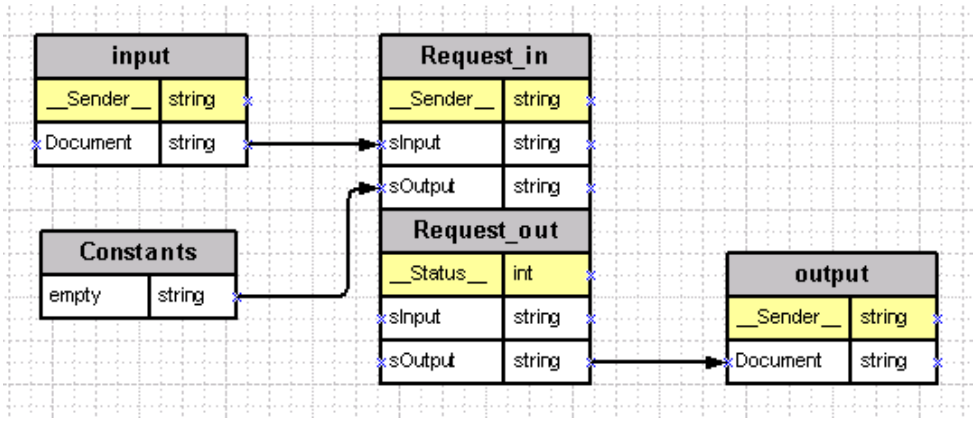


Figure 4. BizTalk Data Flow for Sample Application

This data flow diagram directs BizTalk to forward the entire XML document (received from the eBusiness application) to the application adapter/request object. The request object returns the output to BizTalk, which in turn sends it to the eBusiness application. The next section describes what the request object does in between the point where it receives the input XML document and the point where it returns the output XML document to BizTalk.

Sample Request Object

The following Visual Basic program receives the XML input document from BizTalk, runs through a series of transactions with HostBridge, and returns the requested data as output in an XML document.

```
Public Sub Request(ByRef sInput As String, ByRef sOutput As String)
On Error GoTo ErrHandler
Dim sResponse1 As String, sResponse2 As String, sResponse3 As String
Dim sURL As String, sToken As String, sPrice As String, sShares As String
Dim sStock as String, iStocknum As Integer
Const BASEURL = http://hostaddress:port/hostbridge?HB_TRANID=trad&HB_AID=enter"

'Request1

sURL = BASEURL

'Send request1
sResponse1 = GetRequest(sURL)

'Get stock value from sInput
sStock = GetElementText(sInput, "stock")

'Lookup stock number
iStocknum = StockLookup(sStock)

'Request2

'Get token
sToken = GetElementText(sResponse1, "token")

'Build request string
sURL = BASEURL & "&HB_TOKEN=" & sToken & "&option=" & iStocknum

'Send request2
```

```
sResponse2 = GetRequest(sURL)

'Request3

'build request string
sURL = BASEURL & "&HB_TOKEN=" & sToken & "&opt2=1"

'Send request3
sResponse3 = GetRequest(sURL)

'Get current price
sPrice = GetField(sResponse3, "SHRNOW")

'Get number of shares held
sShares = GetField(sResponse3, "HELD")

'Build output document
sOutput = "<output><price>" & sPrice & "</price><shares>" & sShares &
"</shares></output>"

End Sub
```

Figure 5. Sample request object

The request object is responsible for building and sending the HTTP requests to HostBridge and processing the XML documents returned by HostBridge. A typical URL sent to HostBridge always has certain repeating elements, such as:

```
http://hostaddress:port/         IP address and port assigned to HB
            hostbridge?          indicates this is a hostbridge request
            HB_TRANID=trad&      the CICS transaction name
            HB_AID=enter"        AID key used with a transaction
```

Within the subroutine, we will assign this string to a variable and refer to the as the "base" URL.

Invoking the request Object

The BizTalk schedule passes the input XML document as a string to the request object in the `sInput` variable. The XML input document uses the following form, where the `<stock>` element contains the name of the stock for a requested stock quote.

```
<input>
     <stock>ibm</stock>
</input>
```

Figure 6. Input XML sent from BizTalk to the request object

Request 1: Building and Sending a URL to HostBridge

The first request sent to HostBridge is the base URL shown above. This tells HostBridge to start a CICS transaction named "`trad`." In response, HostBridge returns an XML document to the request object that represents the first screen of the application (Screen 1 in Appendix A). Figure 7, below, is an abbreviated version of the XML document.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!--HostBridge Copyright 2000, 2001 HostBridge Technology, U.S. Patent Pending-->
<hostbridge>
  <token>86d946b5</token>
  <fields>
    <field name="COMP1">
      <value>Casey_Import_Export</value>
    </field>
    <field name="COMP2">
      <value>Glass_and_Luget_Plc</value>
    </field>
    <field name="COMP3">
      <value>Headworth_Electrical</value>
    </field>
    <field name="COMP4">
      <value>IBM</value>
    </field>
    <field name="OPTION">
      <value></value>
    </field>
    <field name="MESS2">
      <value></value>
    </field>
  </fields>
</hostbridge>
```

Figure 7. XML document returned by HostBridge to Request 1 (abbrev.)

While the input request specifies a stock by name, the CICS application expects the end-user to use a numeric value to select the company from a list of company names. Thus, in order for our program to select the company, it must determine the corresponding item number from the list. Our program uses the `GetElementText` function to obtain the name of the stock from the input XML document. It then calls the `StockLookup` subroutine to determine the position of the company in the list and assign the corresponding integer value to the `iStockNum` variable.

```
Private Function StockLookup(ByVal sName As String) As Integer
    'function to return the stock number for a given stock name
        Select Case LCase(sName)
        Case "casey_import_export"
            StockLookup = 1
        Case "glass_and_luget_plc"
            StockLookup = 2
        Case "headworth_electrical"
            StockLookup = 3
        Case "ibm"
            StockLookup = 4
    End Select
End Function
```

Figure 8. StockLookup Function

Once the StockLookup function correlates the company name to an item number in the list, the program is ready to send the next request to HostBridge.

Request 2: Managing State and Submitting Data

The CICS transaction used in this example is a “pseudo-conversational” transaction. Without going into a detailed explanation of what this means, whenever HostBridge is used to invoke a pseudo-conversational transaction it returns a “token.” HostBridge must receive this token on the next request.

In preparation for sending the next request to HostBridge, the program uses the `GetElementText` function to retrieve the `<token>` value from the first XML document returned by HostBridge. In our example, the token value is `86d946b5` and is saved in the variable `sToken`. Next, the program appends the string `"HB_TOKEN="` and the value of `sToken` to the URL. This will tell HostBridge to associate the next transaction with the previous transaction.

Next, the program appends the string `"&option="` and the value of `iStockNum` to the URL. This specifies the stock number to the CICS transaction. Note that the input field `option` was included in the first XML document returned by HostBridge.

The request object sends the resulting URL to HostBridge as follows:

```
http://hostaddress:port/hostbridge?HB_TRANID=trad&
HB_AID=enter&HB_TOKEN=86d946b5&option=4
```

This request tells HostBridge to execute the next leg of the `trad` transaction, selecting company number 4. In response, HostBridge returns an XML document to the request object that represents the second screen of the CICS application (Screen 2 in Appendix A). Figure 9 is an abbreviated version of the XML document.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!--HostBridge Copyright 2000, 2001 HostBridge Technology, U.S. Patent Pending-->
<hostbridge>
  <token>86d946b5</token>
  <fields>
    <field name="OPT2">
      <value></value>
    </field>
    <field name="MESS3">
      <value></value>
    </field>
  </fields>
</hostbridge>
```

Figure 9. XML document returned by HostBridge to Request 2 (abbrev.)

Request 3: Building and returning an XML document to BizTalk

After selecting a company, the request object needs to tell the CICS application what to do: get a quote, buy shares, or sell shares. The request object selects from a list of three items where the order of these items in the list is fixed. Selecting item 1 from the list indicates that we want a quote. Thus, the program appends the string `"&opt2=1"` to the base URL (along with the token). Note that the input field `opt1` was included in the second XML document returned by HostBridge.

The request object sends the resulting URL to HostBridge as follows:

```
http://hostaddress:port/hostbridge?HB_TRANID=trad&
HB_AID=enter&HB_TOKEN=86d946b5&opt2=1
```

This request tells HostBridge to execute the next leg of the `trad` transaction, selecting action item 1 (get quote).

In response, HostBridge returns an XML document to the request object that represents the third screen of the application (Screen 3 in Appendix A). Figure 10, is an abbreviated version of the XML document.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!--HostBridge Copyright 2000, 2001 HostBridge Technology, U.S. Patent Pending-->
<hostbridge>
  <token>86d946b5</token>
  <fields>
    <field name="SHRNOW">
      <value>00163.00</value>
    </field>
    <field name="HELD">
      <value>6033</value>
    </field>
  </fields>
</hostbridge>
```

Figure 10. XML document returned by HostBridge to Request 3 (abbrev.)

The XML document returned by HostBridge contains the share price and number of shares held. In the BMS map used by the TRAD transaction, the fields containing that information are SHRNOW and HELD. HostBridge uses these same names in its XML document. Our program uses a function called `GetField` to obtain the content of the `<value>` element related to the `<field>` element whose name matches the second parameter passed to the function.

After extracting the values of `SHRNOW` and `HELD` out of the XML document, our program formulates an XML document and returns it to BizTalk (as the `sOutput` variable). In our example, the XML output would be:

```
<output>
     <price>163.00</price>
     <shares>6033</shares>
</output>
```

Figure 11. Output XML returned to BizTalk

The BizTalk schedule, in turn, sends this XML document back to the requesting eBusiness application.

Summary

Microsoft's BizTalk Server 2000 enables you to build and deploy integrated business processes within your organization and with your trading partners. While BizTalk represents an emerging category of eBusiness integration software, the existing category of mainframe CICS applications is alive and well within large organizations. For these companies, integrating these applications with products such as BizTalk is imperative.

Using BizTalk to handle the business processes and an application adapter to handle the interaction with HostBridge and the CICS transaction is a preferable approach. It yields higher performance levels because the detailed interaction with the CICS transaction happens outside of the BizTalk schedule. Working together, BizTalk and Host Bridge provide a simple, scalable and secure solution for integrating existing CICS BMS applications with eBusiness processes.

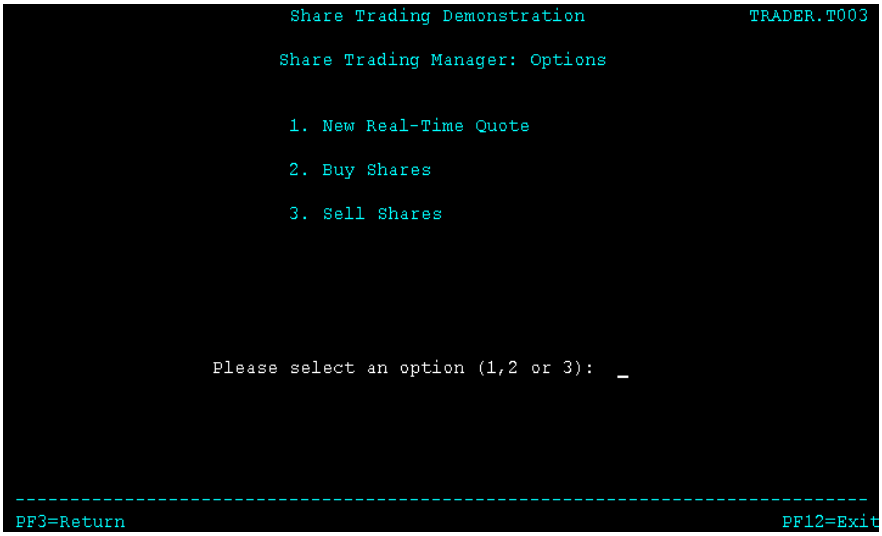
Appendix A: CICS Trader Application

IBM provides a sample CICS BMS application that simulates a stock trading application. The Share Trading Demonstration, or TRADER, consists of only a few screens that allow you to choose a company, get a stock quote, or buy/sell shares. The case study presented in this White Paper uses this application. The application transactions are pseudo-conversational and use the BMS commands SEND MAP and RECEIVE MAP to communicate with the end-user. The application is simple to use. The example below shows how to get a stock quote for IBM using the demonstration application.

1. Logon to CICS.
2. Enter TRAD to start the application. (The following screen should appear.) This screen allows you to select the company whose stock you want to act upon.

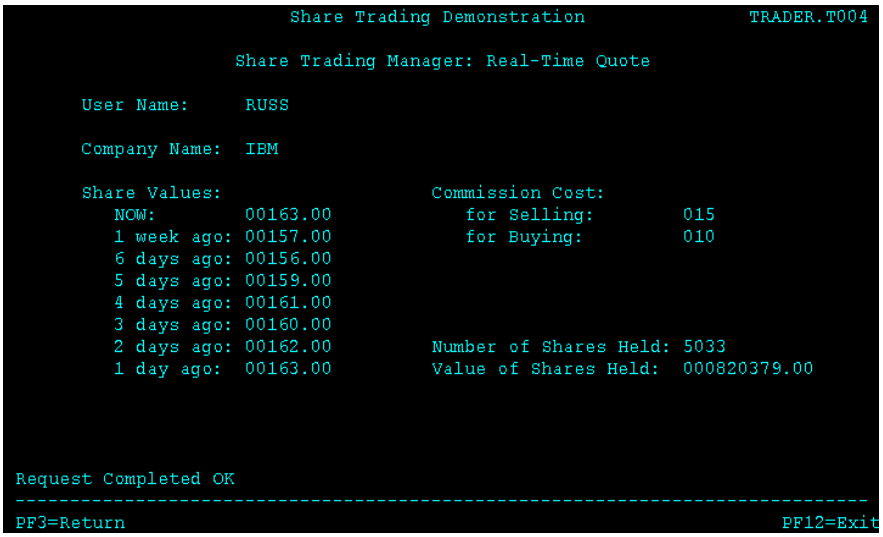
Screen 1. Company selection screen

3. Enter 4 to select IBM. (The following screen should appear.) This screen allows you to get a stock quote, buy shares, or sell shares.



Screen 2. Trading options screen

4. Enter `1` to retrieve the stock quote. (The following screen should appear.) This screen displays the stock quote for the selected company.



Screen 3. Real-time Quote screen

5. Press PF12 to exit the application.