

SOA Integration for CA Mainframe Resources

CA Ideal™ for CA Datacom®, CA Telon®, CA Gener/OL™, CA ADS™ for IDMS™

A HostBridge® White Paper



866-965-2427

info@HostBridge.com

www.HostBridge.com



Copyright Notice

© 2005-2010 by HostBridge Technology. All rights reserved. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without prior written permission. You have limited permission to make hardcopy or other reproductions of any machine-readable documentation for your own use, provided that each such reproduction shall carry this copyright notice. No other rights under copyright are granted without prior written permission. The document is not intended for production and is furnished "as is" without warranty of any kind. All warranties on this document are hereby disclaimed including the warranties of merchantability and fitness for a particular purpose.

Revision: 20100430

Trademarks

HostBridge and the HostBridge logo are registered trademarks of HostBridge Technology. CA Ideal, CA Datacom, CA ADS, CA IDMS, CA Telon, CA Gener/OL, and ca smart are trademarks or registered trademarks of CA Inc. CICS is a registered trademark of IBM. All other trademarks mentioned are property of their respective owners.

SOA Integration for CA Mainframe Resources

CA Ideal™ for CA Datacom®, CA Telon®, CA Gener/OL™, CA ADS™ for IDMS™

Service-oriented architecture (SOA) and web services integration continue to provide organizations with opportunities to do more with existing information resources while reducing cost and complexity. This is especially true for mainframe resources. Today, many organizations that rely on the mainframe are deploying web services to provide more and newer users with mainframe access; modernize the mainframe for today's web-centric customers, partners, and employees; integrate mainframe data and business logic with other enterprise systems and applications; and make the mainframe a valued participant in SOAs.

Integrating mainframe resources into SOAs still poses some challenges, however, particularly when legacy applications are involved. CA mainframe resources – CA Ideal™ for CA Datacom®, CA Telon®, CA Gener/OL™, and CA ADS™ for IDMS™ applications and data – have unique characteristics that must be accommodated to ensure high-performance, high-fidelity integration of CA resources.

This white paper explains how HostBridge allows you to employ SOA and web services as the most effective means to integrate CA resources with other applications.

HostBridge® is mainframe integration software that runs under CICS®, transforming mainframe data to integration-ready XML, HTML, or web services. With HostBridge installed on the mainframe, distributed systems and applications using standard interfaces can invoke CA transactions. HostBridge returns the transactions as XML documents or web services ready for integration with any distributed application or system, including web applications, other enterprise systems, and SOAs.

Unique among integration solutions for CA environments, HostBridge has specially designed panel mapping tools for navigating and integrating CA resources. Where other products rely on row/column coordinates from screen scraping, HostBridge uses field names from panel definitions. This unlocks screen scrapers' problematic dependence on presentation logic for panel interaction. As a result, HostBridge integrations are highly reliable and replicate CA data and business logic with much higher fidelity.

Accessing Terminal-Oriented Applications

CA mainframe development environments allow you to develop terminal-oriented applications quickly and easily. Each screen of an application derives from panel definitions that act as templates for screen layout. These panel definitions contain names for every input field, but these names are lost when applications generate terminal output.

The loss of field names means that when integration software receives output from these applications, there is no metadata with which to construct a meaningful XML, SOAP, or other service document. For this reason, organizations have traditionally

relied on screen scraping gateway technologies to generate web interfaces or to integrate mainframe applications with other applications.

Screen Scraping

Screen scrapers typically run on a physical or logical middle tier. They are tightly coupled in that the screen scraper interacts specifically and only with data available in the source application’s presentation layer. Any changes to the application that alter presentation – i.e., locations of fields on screen – will break the integration.

When screen scrapers communicate with terminal-oriented applications they open a terminal session with the application, send a request to the application, receive the terminal data stream, use HLLAPI to capture the screen data, process the screen data, convert the contents to XML, and ship the XML document to the requester. (A variation of the model is to use FEPI on the mainframe instead of a middle-tier terminal emulation client. This simply moves the screen scraping onto the mainframe.) The most common components of the screen scraper model appear in the figure below.

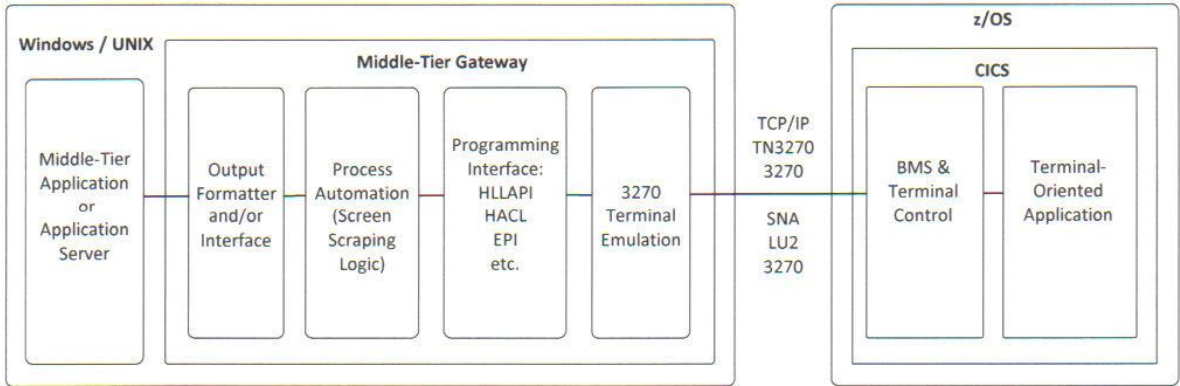


Figure 1. Typical middle-tier screen scraper data path.

Screen scrapers allow you to get to your terminal-oriented mainframe applications, but they create performance bottlenecks and multiple points of failure between the application and the web service.

HostBridge for CA Mainframe Solutions

The diagram below shows the relatively simple data path required for conducting transactions using HostBridge. Fewer moving parts reduce latency and increase reliability. This configuration provides sub-second response times and as many concurrent transactions as the host system allows.

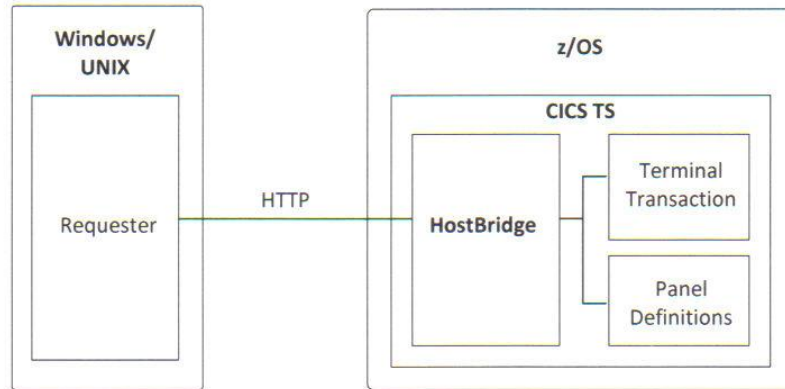


Figure 2. Typical HostBridge data path.

HostBridge Integration Architecture

HostBridge runs on the mainframe under CICS. Exploiting CICS technologies, HostBridge intercepts mainframe data before presentation logic is applied and then, in the case of CA data, uses panel definitions and field names as metadata to control application interactions. This architecture enables HostBridge to deliver enterprise levels of performance and high-fidelity replication of source data and applications.

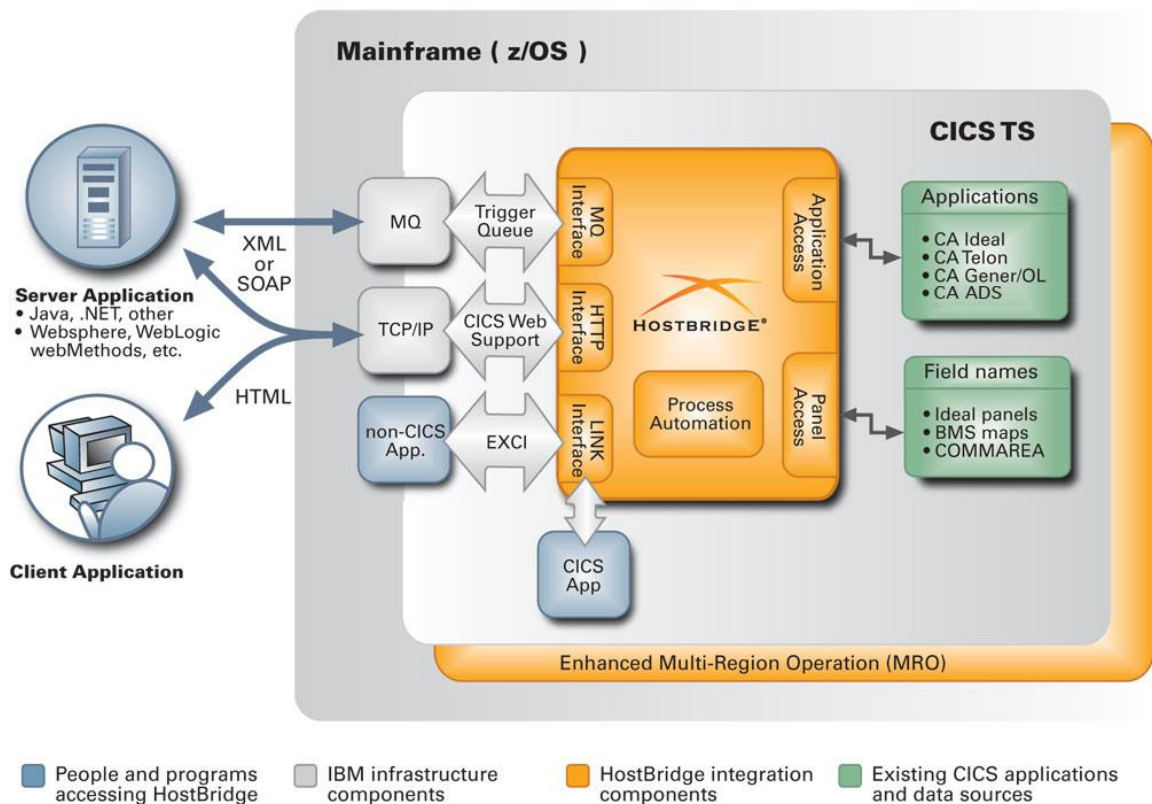


Figure 3. HostBridge and CA Ideal architecture.

HostBridge works with standard CICS components to allow distributed applications to access existing CA applications. For CA Ideal, HostBridge stores panel definition information in a VSAM file, which it later uses to identify panels and field names for each transaction before generating XML output. For CA Telon, HostBridge accesses field names using information directly from the BMS maps. For CA Gener/OL, HostBridge accesses field names using a CA-approved API. HostBridge handles CA ADS panels in the same way it handles CA Ideal.

Using HostBridge with CA Ideal

This section describes how you can use HostBridge to access existing CA Ideal applications.

Creating the Panel Data File

As mentioned above, HostBridge uses a VSAM file as a repository to store and retrieve panel data. To create the repository, use the CA Ideal export utility to export the CA Ideal panel definition dataset. HostBridge includes a panel processor that analyzes the exported panel and stores the panel information in a VSAM file.

HostBridge reads this VSAM file with each transaction to identify the current panel and extract the field names from the dataset. HostBridge then maps the field data on the screen to the field names and returns an XML document to the requesting application. The XML tags identify the field names and contain the corresponding field data.

Invoking HostBridge

Assume you need to access information contained on a CA Ideal screen such as the one shown below.



Figure 4. Sample CA Ideal application screen.

In the screen, notice the “HSII#” field with the value “_____.” Figure 5 shows how HostBridge displays the CICS data in XML after an external application sends an HTTP request with the following format:

[http://company.com:\[port\]/hostbridge?HB_ENTRY=\[transaction\]](http://company.com:[port]/hostbridge?HB_ENTRY=[transaction])

In the URL, the “port” is a port you assign to HostBridge under CICS Web Support (CWS) and “transaction” is the name of the transaction you want to execute. For our purposes, assume the transaction name is “MGET.” Below is the HostBridge output sent back to the requesting application after executing the transaction.

```

<?xml version="1.0" standalone="yes" ?>
<hostbridge sysid="TPT1" build_date="Mar 28 2005 10:56:06">
  <token>3664ef41</token>
  <timestamp>20050511211042</timestamp>
  <status>
    <response>0</response>
    <desc>ok</desc>
  </status>
  <parameters>
    <tranid>MGET</tranid>
    <entry>mget</entry>
    <userid>RACFUSR</userid>
  </parameters>
  <fields count="178">
    <field name="field_r1_c2">
      <name len="11">field_r1_c2</name>
      <value maxlen="79" len="0" />
      <attr byte="e8" justify="n" disp="y" prot="y" num="n" int="y" mdt="n" />
    </field>
    <field name="field_r2_c2">
      <name len="11">field_r2_c2</name>
      <value maxlen="6" len="5">HSII#</value>
      <attr byte="e8" justify="n" disp="y" prot="y" num="n" int="y" mdt="n" />
    </field>
    <field name="field_r2_c9">
      <name len="11">field_r2_c9</name>
      <value maxlen="6" len="6">_____</value>
      <attr byte="40" justify="n" disp="y" prot="n" num="n" int="n" mdt="n" />
    </field>
    ...
    <field name="field_r24_c2">
      <name len="12">field_r24_c2</name>
      <value maxlen="79" len="76">F2Deps F4Reas F5=Card F11PBFS F14Grp/Ctr
        F20MCIN F22MCIL F24Quit</value>
      <attr byte="f0" justify="n" disp="y" prot="y" num="y" int="n" mdt="n" />
    </field>
  </fields>
  <cursor type="pos" pos="166" row="3" col="7">166</cursor>
  <copyright info="HostBridge Copyright 20002005 HostBridge Technology, U.S. Patent
    Pending">
  </copyright>
</hostbridge>

```

Figure 5. Sample XML output (without field rename).

Because MGET is a non-BMS transaction, HostBridge assumes the transaction is a standard CICS 3270 transaction. Notice that in this case, HostBridge identifies the HSII# field using its row/column location (<field name="field_r2_c9">) in the 3270 data stream. To alert HostBridge that panel definitions exist for the transaction, simply add the command HB_FIELD_RENAME=1 to the URL:

[http://company.com:\[port\]/hostbridge?HB_ENTRY=MGET&HB_FIELD_RENAME=1](http://company.com:[port]/hostbridge?HB_ENTRY=MGET&HB_FIELD_RENAME=1)

Now, the XML output includes the field names instead of the row/column values as shown in the figure below.

```

<?xml version="1.0" standalone="yes" ?>
<hostbridgesysid="TPT1" build_date="Mar 28 2005 10:56:06">
  <token>3664ef41</token>
  <timestamp>20050511211707</timestamp>
  <status>
    <response>0</response>
    <desc>ok</desc>
  </status>
  <parameters>
    <trandid>MGET</trandid>
    <entry>mget</entry>
    <userid>RACFUSR</userid>
  </parameters>
  <fields count="118">
    <field name="P1HSII#" index="0">
      <name len="8">P1HSII#</name>
      <value maxlen="6" len="6">_____</value>
      <attr byte="40" justify="n" disp="y" prot="n" num="n" int="n" mdt="n" />
    </field>
    ...
    <field name="P1MESSAGE" index="0">
      <name len="10">P1MESSAGE</name>
      <value maxlen="79" len="62">Consumer number 000MGET invalid, please
        enter a valid number.</value>
      <attr byte="f8" justify="n" disp="y" prot="y" num="y" int="y" mdt="n" />
    </field>
  </fields>
  <cursor type="pos" pos="166" row="3" col="7">166</cursor>
  <copyright info="HostBridge Copyright 20002005 HostBridge Technology, U.S. Patent
    Pending" />
</hostbridge>

```

Figure 6. Sample XML output (with field rename).

Now, that same HSII# field appears as <field name="P1HSII#" index="0">, which is the name assigned to the field in the CA Ideal panel definition. If you invoke the initial transaction using HB_FIELD_RENAME, each subsequent transaction during a session will honor the command.

To continue interacting with the application, let us look at a URL that you could send to the application to change the contents of the P1HSII field from “_____” to “012345.”

http://company.com:4041/hostbridge?HB_TOKEN=3664ef41&P1HSII=012345

This part of the URL...	Does this...
http://company.com:4041/	Identifies the address and port where HostBridge runs. The HTTP listener passes URLs on this port to the HostBridge analyzer.
hostbridge?	Identifies this as a HostBridge session. Everything that follows the “?” in the URL is the command string that controls HostBridge behavior.
HB_TOKEN=3664ef41	After an initial connection to the host application, HostBridge returns a session ID called a state token. Seen in the XML above as: <p style="text-align: center;"><token>3664ef41</token></p> Subsequent transactions with the host must include this token in the URL so CICS will recognize the transaction as part of an existing session.

P1HSII=012345

Sends the field name and the data value we want to enter in the transaction. HostBridge interprets any name/value pair sent along the URL as input to a field unless the name starts with "HB_" (e.g., HB_TOKEN).

HostBridge can also receive requests in the form of XML documents that conform to a fixed schema. The following XML document submits the same transaction data as the URL above.

```
<?xml version="1.0" ?>
<hostbridge>
  <transaction>
    <parameters>
      <hb_token>3664ef41</hb_token>
    </parameters>
    <fields>
      <P1HSII>012345</P1HSII>
    </fields>
  </transaction>
</hostbridge>
```

Figure 7. Sample XML input document.

For companies integrating applications as web services, HostBridge supports SOAP messages as input to invoke HostBridge. The following SOAP message submits the same transaction data as the XML and URL requests above.

```
<?xml version="1.0" ?>
<SOAP-ENV:Envelope>
  <SOAP-ENV:Body>
    <hostbridge>
      <hb_token>3664ef41</hb_token>
      <P1HSII>MICHAEL</P1HSII>
    </hostbridge>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure 8. Sample SOAP request.

CA Telon

The method HostBridge uses for integrating CA Telon depends on whether or not you generated your applications using BMS.

BMS Applications

If you generate your CA Telon applications to use BMS, HostBridge uses the Link3270 Bridge facilities to access the information in the BMS maps. This method allows HostBridge to generate XML output based on the field names within the maps before CICS creates a 3270 data stream. This is the same method HostBridge uses to access all other CICS BMS applications and is the preferred method for integrating CA Telon.

Non-BMS Applications

If you generate your CA Telon applications without using BMS, you can generate BMS maps that HostBridge uses to determine field names for each panel of an application. (When using HostBridge with non-BMS CA Telon applications, HostBridge uses the START Bridge facilities rather than the Link3270 Bridge facilities.)

To use HostBridge with non-BMS CA Telon applications,

- Include the CICSBMS statement in the generation job to export BMS maps. (This does not generate your application as a BMS application; it merely creates the BMS maps for HostBridge to use.)
- Use our HB_TELON directives to identify CA Telon applications that do not use BMS.
- Interact with your CA Telon applications as described in the previous section on CA Ideal.

CA Gener/OL

CA and HostBridge worked together to develop an API that provides access to the field names used within CA Gener/OL applications. The API provides access to panel names, field names, and field locations for each screen of an application, while HostBridge includes a program (HBR\$GENL) that receives control from CA Gener/OL and processes the information before the application generates a 3270 data stream.

To use HostBridge with CA Gener/OL applications:

- Contact CA to receive and install the Zap patch to the CA Gener/OL SG#CE module. (This patch provides the API for HostBridge support.)
- Use the HostBridge HB_GENEROL_RENAME=1 directive to indicate a CA Gener/OL transaction.
- Interact with your CA Gener/OL applications as described in the previous section on CA Ideal.

CA ADS

HostBridge maps CA ADS panels in the same way it handles CA Ideal panels, using a VSAM file as a repository to store and retrieve panel data. To create the repository, use the CA ADS export utility to export the CA ADS panel definition dataset. HostBridge includes a panel processor that analyzes the exported panel and stores the panel information in a VSAM file.

HostBridge reads this VSAM file with each transaction to identify the current panel and extract the field names from the dataset. HostBridge then maps the field data on the screen to the field names and returns an XML document to the requesting application. The XML tags identify the field names and contain the corresponding field data.

HostBridge Process Automation

A key performance component of HostBridge integration software is our Process Automation engine. This mainframe-resident, JavaScript-based development and runtime facility automates the many transactions inherent in terminal-oriented application processes, which may involve dozens or even hundreds of screens. Scripting tools installed in the middle tier require an HTTP request/response for each transaction, thus generating large numbers of requests/responses and increasing latency. In contrast, HostBridge Process Automation orchestrates complex transaction processes on the mainframe and automates them as a single web service. The result is an increase in response speed of as much as 20 times.

Conclusion

CA Ideal for CA Datacom, CA Telon, CA Gener/OL, and CA ADS for CA IDMS continue to offer powerful capabilities for mainframe application development. HostBridge allows organizations with CA mainframe resources to protect their investments by integrating these resources with any distributed application or SOA.

Using industry standards such as XML, SOAP, and JavaScript, HostBridge allows developers to leverage existing skills and knowledge to build and deploy integration frameworks more rapidly and reliably. Requiring no changes to mainframe applications, HostBridge removes mainframe programming from the integration equation.

HostBridge does recognize that mainframe application are sometimes changed. In such cases, our panel mapping capabilities protect you against integration breakage. Because HostBridge relies on field names for mainframe application interaction, not screen coordinates, changes that alter screen geometry do not affect integration. This saves time and money for development staff and reduces the chances that users will receive errors when they interact with the mainframe through distributed interfaces.

HostBridge Technology

100 East 7th Ave.

Stillwater, OK 74074

www.HostBridge.com

