

# **Transaction System Migration**

---

## **Executive Overview**

The new transaction paradigm is to move beyond a single platform implementation and connect to Web Services. In the past, traditional host based transaction sub-systems permitted business to build interactive “terminal based” applications.

These transaction sub-systems are tied to COBOL applications with application interfaces built from text based screen mapping services such as BMS<sup>1</sup>, and MFS<sup>2</sup> permitting a pseudo-conversational style of accessing and updating application information.

The current challenge is to either connect legacy business applications to the web or to migrate business applications for full integration onto Application Servers.

This paper discusses one of the options for propogating Application Servers with legacy applications written in COBOL. It is intended as an overview with references to successful migration projects that have used LegacyJ PERCobol to enable platform independent deployment of COBOL transaction based applications.

### **CONTENTS**

<b>EXECUTIVE OVERVIEW.....</b>	<b>1</b>
<b>MIGRATION GOALS .....</b>	<b>2</b>
<b>MIGRATION PROCESS .....</b>	<b>2</b>
COBOL.....	2
3270 SCREEN INTERFACES .....	2
TRANSACTION COMMUNICATIONS ...	2
<b>MIGRATION.....</b>	<b>3</b>
PHASE 1 .....	3
PHASE 2 .....	3
<b>CICS MIGRATION OPTIONS .....</b>	<b>4</b>
CICS LINK .....	4
CICS XCTL.....	4
CONTROL FLOW.....	4
NEW DEPLOYMENT CONTROL FLOW	5
<b>CICS SCREEN INTERFACES .....</b>	<b>6</b>
SEND AND RECEIVE MAP .....	6
VSAM FILES.....	6

*REVISION: OCTOBER, 2003*

---

<sup>1</sup> Basic Mapping Support (BMS) used in CICS applications

<sup>2</sup> Message Format Service (MFS) used in IMS applications

## Migration Goals

Specific measurable goals of a migration project are critical in producing a quantifiable result. The migration of an existing COBOL business application to a platform independent, Java enabled application requires that changes be made. The traditional COBOL application source may run within CICS/DB2 and 3270 display application while the target may be a web-based application running on an Application Server independent of specific system dependencies.

The preservation of the COBOL source with minimal changes is important to maintaining the integrity of the source code. Ideally, the business processes should continue to execute and process data in the same manner as it once did within the transaction sub-system. It also can assist during the debugging process because both previous and converted systems can be tested in parallel to confirm that the same application behavior exists on both systems.

## Migration process

### **COBOL**

Using LegacyJ PERCobol permits the migration of code with only minimal modifications. PERCobol supports the IBM S/390 COBOL syntax with the exception of the EXEC CICS statements that must be changed for Application Server execution.

Other options do exist to produce CICS and IMS clients with PERCobol. Documentation for this option with samples are included in the LegacyJ PERCobol Programmer's Guide. This guide is include in the PERCobol HELP system and is accessible from the LegacyJ web at [http://www.legacyj.com/perc\\_doc.html](http://www.legacyj.com/perc_doc.html)

### **3270 Screen Interfaces**

The 3270 screen user interfaces are normally in two forms BMS maps or MFS maps. BMS can be converted straight to HTML using the "3270 Bridge Facility" available with CICS TS 1.3<sup>3</sup>. Once converted to HTML, cosmetic changes can be applied to the screens to make them more pleasing or user-friendly.

### **Transaction Communications**

Communication between the HTML screens and the COBOL/Java business logic can use either JSP or ASP in conjunction with SQL stored procedures and XML data streams to pass information between the user interface and the application program.

---

<sup>3</sup> See <http://www-4.ibm.com/software/ts/cics/library/presentations/c82.pdf> for a high level overview of the BMS to HTML conversion process.

# Migration

## Phase 1

In phase one, modifications must be made to the legacy COBOL code to remove mainframe sub-system dependencies. These include:

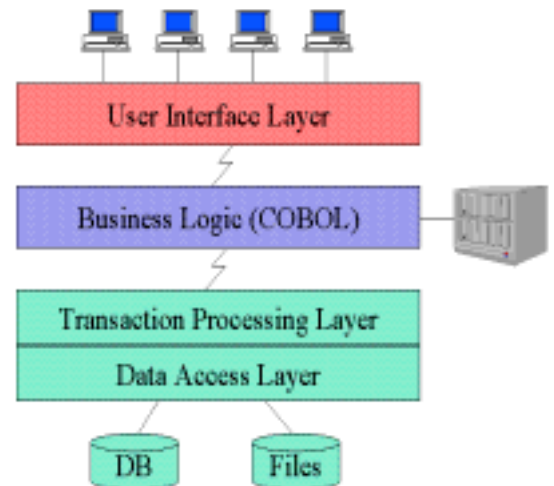
- Removing any CICS API command dependencies the application may have and replacing it with alternative code that “mimics” the CICS process.
- Compile COBOL source code using LegacyJ PERCobol to generate Java byte code. The resulting Java class is then ready to deploy in the Java environment along with the PERCobol deployable runtime routines.
- Convert existing DB2 databases as needed. Any VSAM files are converted to an equivalent indexed file system such as C-ISAM or LegacyJ Indexed files.
- Regenerate User Interfaces using the original CICS 3270 application screen layouts. The application's BMS maps are converted to HTML using the using the "3270 Bridge Facility" available with CICS TS 1.3<sup>4</sup>. Once converted, cosmetic changes for style consistency and readability are applied to the HTML documents.
- Connection of the new application interface screens to back to the application. The data entered into the HTML screens is processed through ASP and sent to the application server using an XML data stream. The XML data stream is received by the server application, parsed, and then sent to the business logic program (formally the CICS program).

By the end of phase one, a working, converted system exists that performs the same functions as it did on the mainframe. At this point, no enhancements have been made to any of the application programs unless they were absolutely necessary.

## Phase 2

Since LegacyJ PERCobol generates Java byte code, the user now has the option to add any requested enhancements in either Java or COBOL. Using PERCobol additional granularity or restructuring of objects or EJBs is permitted to expose more of the business logic for Java extensions. PERCobol may be used to deploy or migrate applications between systems offering alternatives for application execution.

The solution created with the use of PERCobol permits options in the creation of application extensions with either COBOL or Java.



<sup>4</sup> See <http://www-4.ibm.com/software/ts/cics/library/presentations/c82.pdf> for a high level overview of the BMS to HTML conversion process.

# CICS Migration Options

## **CICS LINK**

Linking from one CICS program to another is the same as using a COBOL call statement.

```
EXEC CICS LINK  
  PROGRAM('DP8026')  
  COMMAREA(DP8026-RECORD-AREA)  
  LENGTH(DP8026-LENGTH)  
END-EXEC.
```

## **Equivalent COBOL CALL**

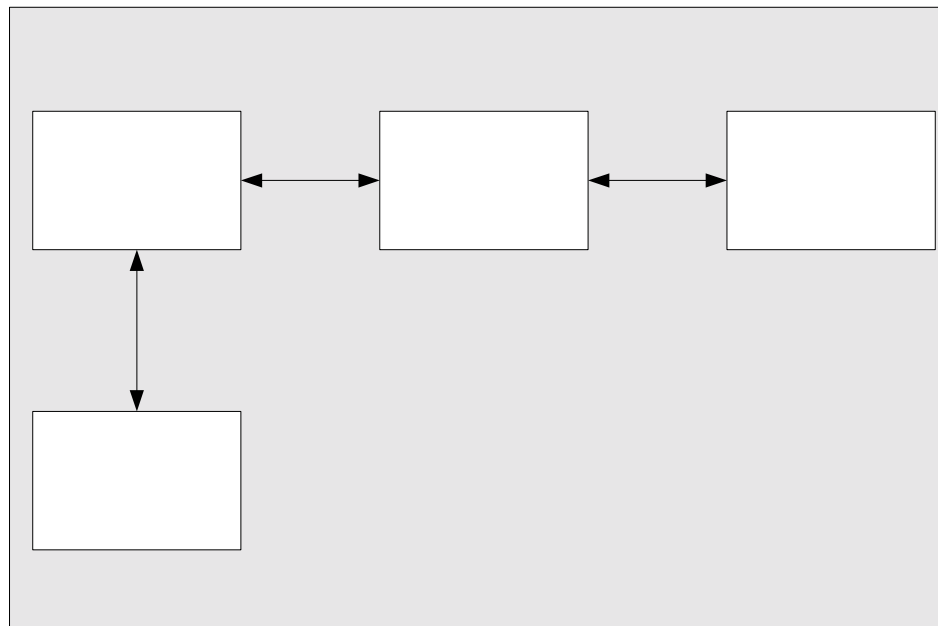
```
CALL 'DP8026' USING DP8026-RECORD-AREA.
```

## **CICS XCTL**

A CICS XCTL command is different from a CICS LINK in that the invoking program and the invoked program are on the same level. A LINKed-to-program on the other hand is one level further down than the program that initiated the link. Also, the initiating program in an XCTL command promptly ends after issuing the command. To overcome this, your application should use a manager program or class that drives all of the other programs. Much like CICS does now. The figures below illustrate this concept.

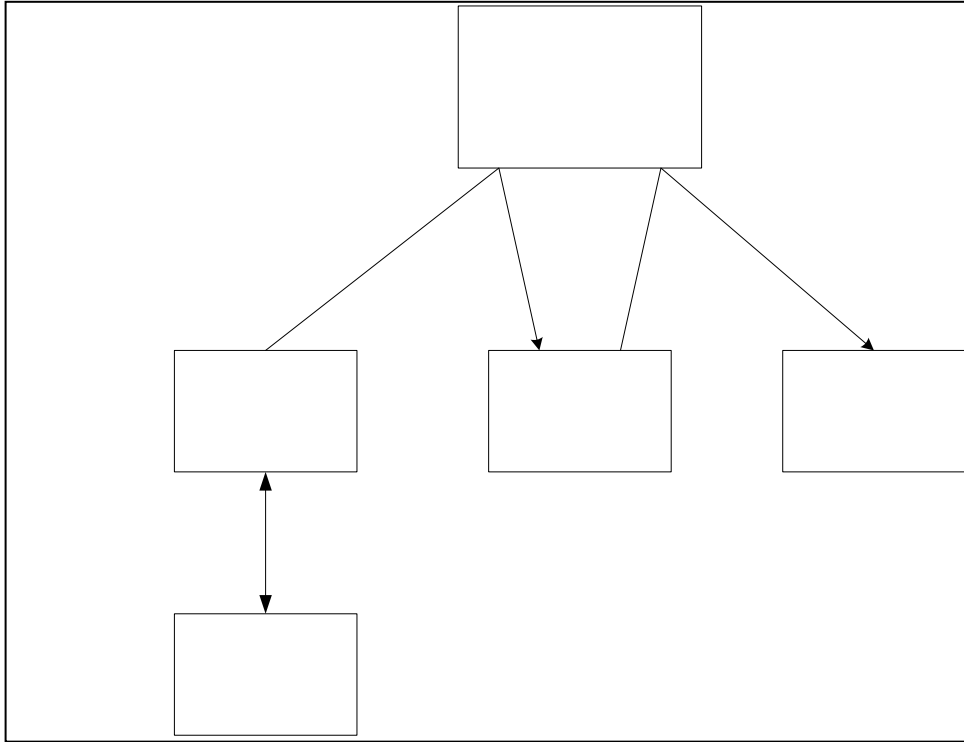
## **Control Flow**

The following diagram illustrates how the CICS sub-system manages the transfer of control from one program to another.



### ***Deployment Control Flow***

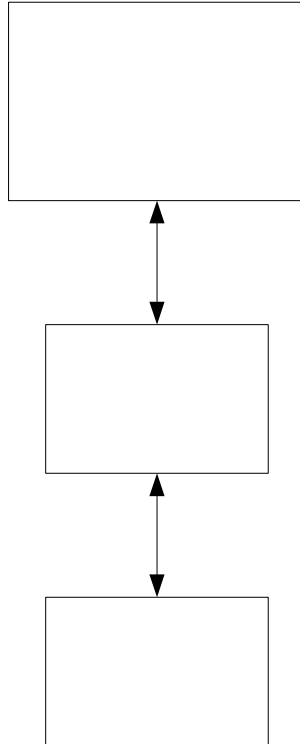
The following diagram illustrates how the new deployment with PERCobol illustrates control transfer from one program to another.



# CICS Screen Interfaces

## ***SEND and RECEIVE MAP***

Depending on the type of user interface, there are several solutions available to help send and retrieve information from a user. One solution is to create a wrapper class or program that intercepts the screen fields and moves them into the symbolic map record layout for the mapset. Then, when the application is called, the screen fields can be accessed through the programs linkage section. Any new data entered into the display fields can then be parsed back out through the wrapper class or program. The figure below illustrates this approach.



## ***VSAM Files***

The READ, WRITE, REWRITE, BROWSE commands used to access and perform operations on a VSAM file will need to be commented out. These CICS commands are replaced with COBOL calls or invocations to classes that can perform those operations. The utilization of standard classes ensures that data is accessed and updated in a uniform manner.