CICS Modernization & Integration
Modernization – easier than thought

“The irony is that host applications are probably better suited for exposure as part of an SOA than many applications based on more modern 4GL object-oriented languages.

When folks wrote screen-based transactions many months ago, they wrote it at a business function viewpoint: I add a customer, I add an order for that customer, I check backlogs for that customer, etc. So in many respects, those CICS screens of 15 years ago are better suited to service orientation than a lot of the newer, distributed code that’s been written over the last several years, because of their affinity with a business function, what did the object-oriented guys do? They took those screens and they broke them down into a thousand different objects.”

Phil Murphy, Forrester Research
CICS and SOA is big!

Chart 11: Which System z middleware have you enabled, or do you plan to enable, with web services?

Source: Arcati Limited - The Arcati Mainframe Yearbook 2007
Three Styles of Application Transformation

1. **Transform User Experience**
   - Enhance user interface and workflow for quick return on investment
   - gui-fication
   - Low Risk – no change to app
   - Quick return on investment
   - Affected by change

2. **Transform App Connectivity**
   - Supply new interfaces to existing application based on open standards like JCA, JMS, J2EE or Web Services
   - Low Risk – little or no change
   - Medium Return on investment
   - Medium affected by change

3. **Transform App Architecture**
   - Break off monolithic applications and create flexible, reusable modules
   - componentization
   - High risk → modification
   - High Return on investment
   - Increases application flexibility
Best practice in CICS application design is to separate key elements of the application, in particular:

- Presentation logic eg. - 3270, HTML, XML
- Integration or aggregation logic - Menu, router, tooling
- Business logic - Reusable component
- Data access logic - VSAM, DB2, IMS, ...

Provides a framework for reuse and facilitates separation of concerns, clear interfaces, ownership, and optimization

Allows **callable business logic** – parameters passed via COMMAREA
3270 Based Program Reuse

- Some programs combine presentation, integration, and business logic
- Service Flow Modeler (SFM) and Link3270 Bridge provide a callable, COMMAREA interface to many BMS and terminal-oriented programs
  - Information in the COMMAREA is passed to the BMS application
  - Does not use VTAM or screen scraping
  - No changes required to existing BMS application
Connectivity to CICS

- **Typical clients...**
  - Web service requester
  - Java servlet or EJB running in a J2EE app server
  - C# application running in a Microsoft .NET VM
  - Web browser
  - Messaging middleware

- **Transforming technologies...**
  - External connectors
  - Internal adapters (user written or generated by tools)
  - Standard IP-based protocol
Host Integration – CICS Strategies

Web services

J2EE App Server

Web services

JCA

Servlet → EJB

Browser

WebSphere MQ

TCP/IP Sockets

CICS TS

A

1

2

3

4

5

6

I

B

D

© 2006 IBM Corporation
Factors Influencing your Integration Choices

- **Business factors**
  - Agreed company standard or reference frameworks
  - Preferred application development environment and tools
  - Availability of skills

- **Technical factors**
  - Security
  - Transactional scope (1pc, 2pc)
  - Performance
  - Granularity
  - Reliability, availability and scalability (RAS)
  - Synchronous or asynchronous invocation
  - Inbound and outbound capability
  - Client/server coupling
  - Data conversion
  - State management

- **Applications today are typically delivered across several e-business clients**
CICS Web Services Support

- **Web Services Clients (examples):**
  - Another program in CICS (invoke web service)
  - BPEL process (Process Choreography – WPS/WID)
  - WebSphere Web Services Gateway
  - .NET assembly
  - WebSphere MQ client
  - Anything that can invoke a Web Service
CICS Web services support (overview)

- The pipeline is a set of message handlers that are executed in sequence
- Message handlers perform ‘infrastructure’ processing on request and response messages and can be used for security, auditing, monitoring etc.
CICS Web Services Support (Requester)

- **Invoke Web Services from CICS programs**
  - Any Language (COBOL, Assembler, PL/I, C, C++, Java)
  - EXEC CICS INVOKE WEB SERVICE …

- **Web Service Could be (examples):**
  - A CICS based Web Service
  - BPEL process (Process Choreography – WPS/WID)
  - WebSphere Web Services Gateway
  - .NET assembly
  - Any Web Service (SOAP over HTTP or MQ)
CICS Transaction Gateway

WAS
Servlet, JSP, EJB
Applet, Any Java Pgm
Many Languages, Many Platforms

CICS TG
CICS Transaction Gateway

CICS TS
CICS Program
Business logic

Other/Any
Web Service Client

WAS
WebSphere Application Server Web Service Support

CICS TG
CICS Transaction Gateway

CICS TS
CICS Program
Business logic

WAS=WebSphere Application Server
CTG Topologies

Topology 1. Application Server and the CICS Transaction Gateway are both deployed on a distributed (non-zSeries) platform.

Topology 2. Application Server is deployed on a distributed platform and the CICS Transaction Gateway is deployed on z/OS.

Topology 3. Both Application Server and the CICS Transaction Gateway are deployed on zSeries.
Enterprise JavaBeans

CICS Transaction Server V3.1

© 2006 IBM Corporation

session bean in CICS (local / remote)

session + entity bean in other EJS

CICS-Pgm(DPL) über Java Connector (CTG)

JCICS: Link / Start + CICS Ress.Zugriff

JDBC, SQLJ

"any Java", z.B. XML, java.net, etc.

Other/Any

WebSphere App Server

CICS TS

Web Service Client

WebSphere Application Server Web Service Support

EJB Business Logic

EJB Business Logic

EJB Business Logic

EJB Business Logic

EJB Business Logic

© 2006 IBM Corporation
# CICS Interoperability Summary

<table>
<thead>
<tr>
<th>Standard architecture</th>
<th>Capabilities</th>
<th>Security to zSeries</th>
<th>Transactional scope</th>
<th>Interface</th>
<th>Coupling</th>
</tr>
</thead>
</table>
| **1. Web Services**   | Synchronous (HTTP)  
                        | Asynchronous (WebSphere MQ)  
                        | Inbound and outbound | User ID + password  
                        | SSL | Local CICS transaction  
                        | Global transaction | CONTAINER  
                        | COMMAREA XML | Loose |
| **2. JCA**            | 32KB max message size  
                        | Inbound only  
                        | Synchronous and Async | User ID + password  
                        | Thread identity  
                        | SSL | Local transaction  
                        | Global transaction | COMMAREA | Medium |
| **3. Enterprise JavaBeans** | EJB state management  
                        | Inbound and outbound  
                        | Synchronous | EJB security roles  
                        | SSL | CICS transaction  
                        | Global transaction | Enterprise  
                        | JavaBean session bean | Tight |

**Standard transport**

| **4. WebSphere MQ**   | Inbound and outbound  
                        | Asynchronous  
                        | Assured delivery | User ID + password  
                        | SSL | CICS transaction | WebSphere MQ API or  
                        | COMMAREA | Medium |
| **5. HTTP**           | Inbound and outbound  
                        | Synchronous | User ID + password  
                        | SSL | CICS transaction | CICS WEB API | Medium |
| **6. TCP/IP sockets** | Inbound and outbound  
                        | Synchronous and Async | User ID + password | CICS transaction | CICS sockets API | Tight |
Any more questions?