Hello and welcome to the “e-Business access to CICS – Strategic Options” seminar.

My name is Mark Cocker and I have the pleasure of working in the CICS Strategy and Product Management team in Hursley Park, England – home of the development team for CICS Transaction Server and tools.

Lets move to slide 1 to discuss the objectives of this seminar.

Abstract

- Why do successful enterprises want to architect e-business solutions to leverage the value of their CICS applications?
- What choices are available to connect to CICS?
- Which access method is ‘best’ to connect from a particular e-business client or server to program in CICS?
- This presentation discusses the factors that have a bearing on the answer to these questions.
- Over many years CICS has made available a large choice of protocols, interfaces and APIs to connect to and from CICS servers.
- Access methods discussed include standard architectures and interfaces such as SOAP, JCA and Java RMI and standard transports such as WebSphere MQ, HTTP, CICS sockets.

During this presentation we will explore the reasons why successful enterprises architect e-business solutions to leverage the value of their CICS applications, the choices available to access CICS from the many different e-business clients, and the merits of one choice over another. Customers frequently ask which access method is ‘best’ to access CICS. Whilst there is no one answer, we will discuss the environmental, functional and non-functional factors that have a bearing on the answer to these questions.

Over the past 35 years, part of the success of CICS has been due to the fact it does supports many types of clients, and in recent years the support of a choice of protocols, interfaces and APIs to connect to and from CICS servers. SOAP, JCA, Java RMI, WebSphere MQ, HTTP, TCP/IP sockets… the choice seems endless. However, choice can be overwhelming, and not making the right choice can lead to over-taxing demands on clients, less than optimised networks and systems, and reduced flexibility for future access and re-use.

This presentation aims to outline which of these choices are industry best practice – ie. what are the strategic options?

There are a number of redbooks and whitepapers referred to at the end of this presentation which will help you explorer this topic in more detail.

So, lets get started. Moving to slide 2….

Acknowledgements

During this presentation I may use some trademarks or abbreviations. They are shown here.

Moving on to the agenda on slide 3…

Agenda

- What is e-business and CICS relate to it?
- Three Styles of Transformation
- What assets can be transformed
- Link3270 Bridge

- e-business access to CICS programs
- Which architecture should I use to connect to CICS
- Strategic connectivity options
  - SOAP, JCA, JAVA RMI
  - JMS, HTTP, CICS Sockets

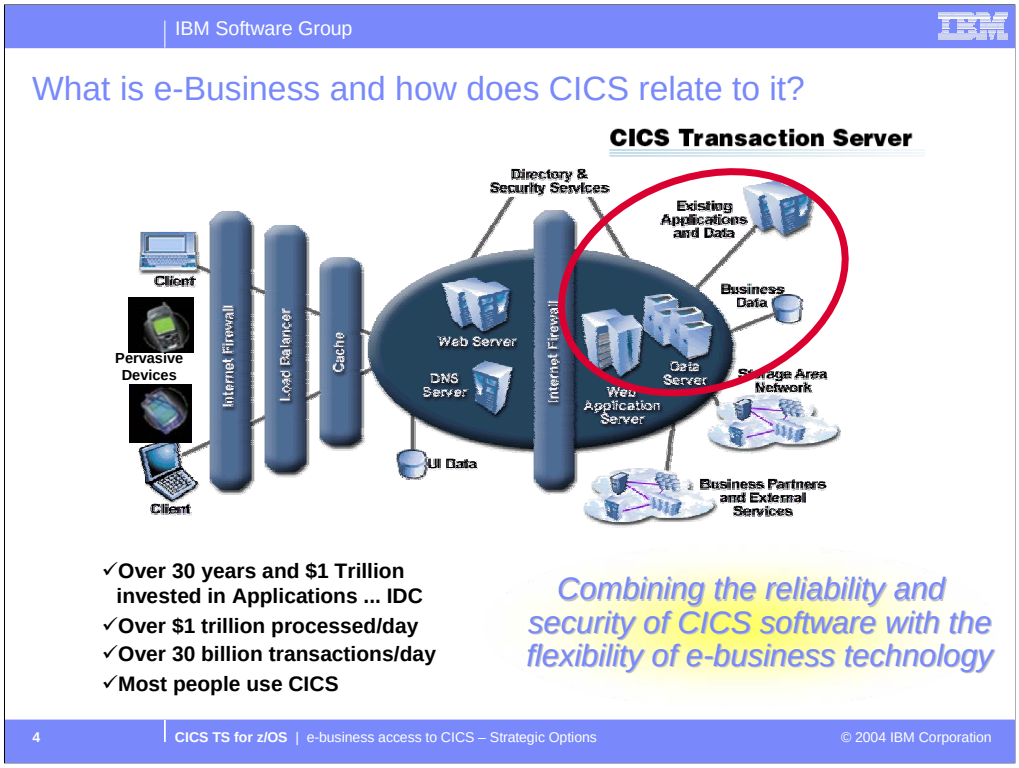
- Strategic options table and summary

- Further reference

We will start off by asking “What is e-business?” and discuss the styles of transforming CICS programs into a form that can be reused.

We then will discuss e-business access to CICS programs, the architectural choices, and the connectivity options, finishing up with a summary and further references.

So if we go to slide 4, I would like to give a brief background on e-business, and why it is so important to business.



The challenges of the fast paced networked economy places pressure on all enterprises. The requirements to grow business with new business models, acquisitions, new value creation, and better collaboration with partners and suppliers to reduce costs and improve margins require extraordinary support from IT resources. Traditionally, IT impeded change, or at best determined the rate of change. Today, using middleware, companies can enable change at the rate and pace required by the networked economy.

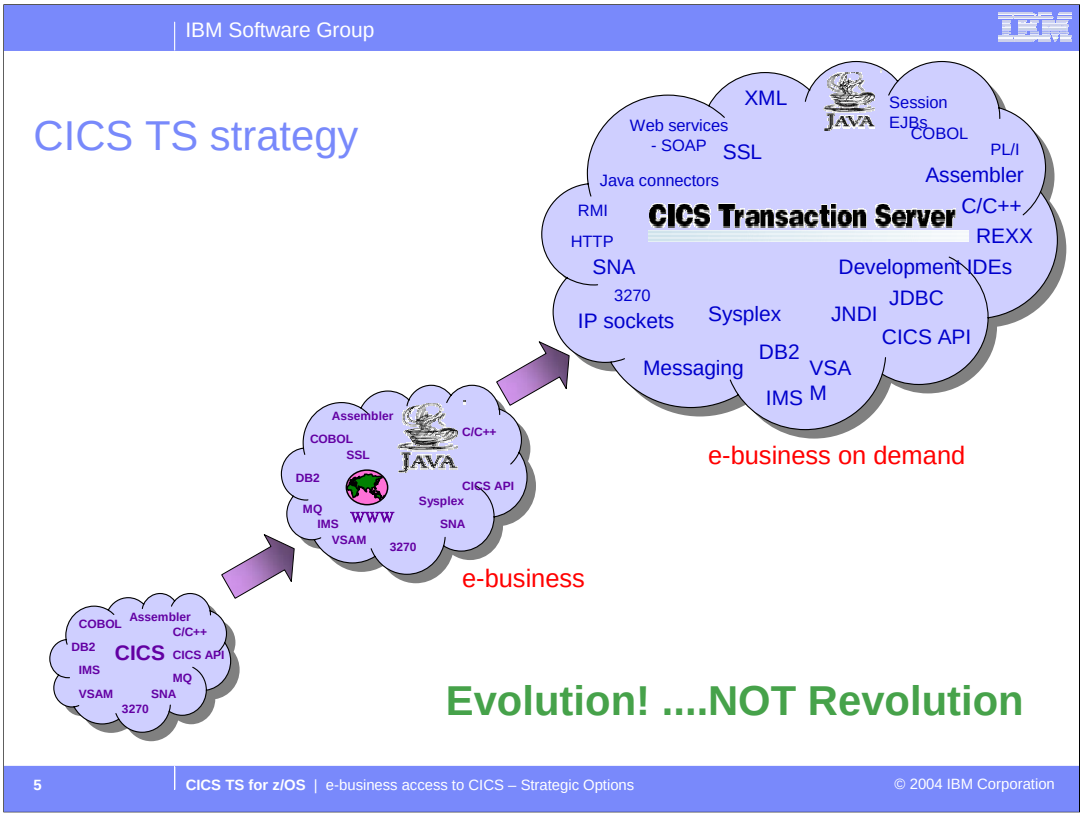
e-business has ushered in the next generation of transaction processing. The promise of e-business is inherently about doing business transactions over the web. CICS is well positioned to leverage its' core competency in transaction processing to deliver on the promise of scalable, secure transactions, a paradigm it pioneered over 35 years.

Dynamic e-business, fuelled by Web services and other service-oriented architecture implementations, represent the next generation of transaction processing. Web services are the building blocks for constructing the distributed Web-based applications. Web services enable the connection of applications within businesses and between businesses to be established quickly and easily; interactions with marketplaces can be established more efficiently; business functions can be delivered to a broader set of customers and partners; and new business models can be pursued by combining applications in new dynamic ways.

But there's still a lot of uncharted territory in e-business, and adopting unproven technologies can be risky. To help you develop a winning business strategy that increases market-share and capitalizes on new revenue sources, IBM supports leading-edge technologies that builds on the strength and reliability of IBM CICS® software.

CICS is IBM's premier transaction processor for the z/OS and OS/390 environments, enabling thousands of enterprises to run business-critical workloads totalling billions of transactions per day, with a financial value of trillions of dollars. Many of these enterprises see significant advantages in building upon their core investment in CICS skills and applications, and extending that investment to provide the basis for their new e-business solutions.

Moving to slide 5…..



The CICS TS strategy and its defining role in enterprise solutions is to:

•Enable efficient and optimized extension and reuse of existing CICS applications and business logic
•Enable enterprises with a strong investment in CICS skills and infrastructure to create applications using new technologies by building on those skills
•To support and manage mixed application workloads within one or more CICS systems
•To reuse of existing DB2, IMS DB, and VSAM data from Java and EJB applications.

Many of the features you can see on this foils were introduced in CICS to support of this strategy.

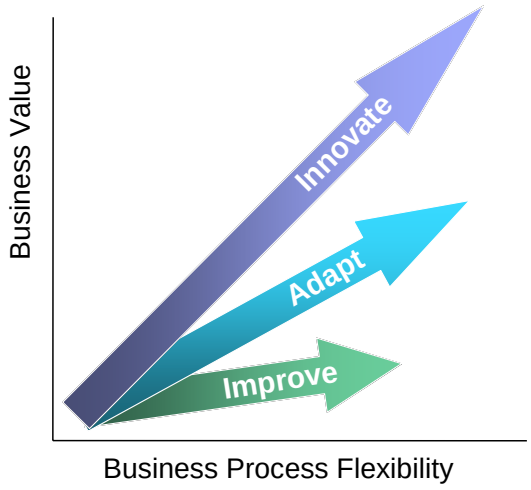
CICS TS V2 provides a platform from which an enterprise can evolve towards the adoption of e-business exploiting all the advantages of an evolutionary approach while capturing new opportunities derived from the latest industry standard e-business technologies.

So, with many reasons to access CICS, let us look at the major styles in which applications in CICS can be accessed.

Moving to slide 6…

Three styles of transformation

1. Improve
   re-face applications to enhance the user experience
2. Adapt
   re-use applications as part of a larger solution
3. Innovate
   re-engineer applications to reflect business processes



In order to move towards an on-demand business, customers need to transform their technical infrastructure from unique, single purpose applications to shared resources. This may be achieved using new software components, which enable transformation to progress through three key stages, as follows:

1 - Improve – re-face applications to enhance the user experience

This first style of transformation focuses on reducing costs and increasing productivity by enhancing the user interface. This method is the most accessible because it requires the ***lowest*** level of investment. Customers can achieve a ***rapid*** return on investment by creating a better user experience, which improves ease of use and employee productivity. This can also result in lower training costs and an increase in overall user satisfaction.

2 - Adapt – re-use applications as part of a larger solution

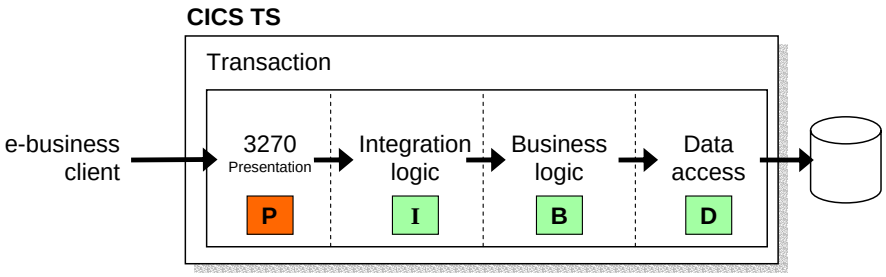
The second style of transformation focuses on extending existing applications beyond their original design to create integrated solutions. This method requires greater investment, but yields ***significantly greater*** business value and process flexibility. Customers can turn existing applications into reusable services, which can be accessed by new users or extended to new front-end business functions. The underlying principle - that existing applications are re-used with little if any change – means this is a lower-risk approach than a replacement strategy, which involves re-writing applications. A key concept of the *adapt* style of transformation is that applications are re-used via programmatic interfaces which enable the application to be invoked locally or remotely, either via a standardized API or via a standardized network protocol.

3 - Innovate - re-engineer applications to reflect business processes

The third style of transformation involves some re-engineering of the original application. Undoubtedly, this method requires ***greater*** investment of resources and time, which should only be attempted after a considered decision, but has the capability to create “components” from existing applications which are fully flexible and configurable for use in new applications. This reuse of business logic is called componentization and may result in significant cost savings when compared with developing new application code.

Moving on to slide 7 we ask, What assets in CICS can be transformed.

What assets in CICS can be transformed?



- Best practice in CICS application design is to separate key elements of the application, in particular:
  - 3270 presentation logic
  - Integration logic
  - Business logic
  - Data access

Typically CICS customers have created programs with two types of interfaces, 3270 terminal-driven programs and COMMAREA programs.

CICS terminal-driven programs – depicted as a “p” in a red box on this foil - are designed to be invoked directly from an IBM 3270 Display Station or similar buffered terminal device. Invocation usually corresponds to a single interaction in an end user dialog, starting with receipt of a message from the terminal and ending with transmission of a response message to the same device. Terminal-driven programs can be written to analyze and construct device-specific datastreams to interact with the terminal but this can be tricky. More than likely terminal-driven programs use the CICS **Basic Mapping Support** (BMS) APIs that provide a level of abstraction between the datastreams and the data the program is really interested in**. BMS** simplifies application programming for terminals such as the 3270 devices by enabling the programmer to define the static portions of the screen and dynamic fields to display or accept input in a definition file called a MAPSET which is separate from the terminal-driven program. The terminal-driven program uses simple BMS APIs to deal with the dynamic fields which contain the valuable input and output information and let CICS worry about the datastreams.

CICS COMMAREA programs – shown as green boxes on this foil - are passed a request and send a response via an area of memory called the COMMunications AREA. The format of the data is typically language specific – most notable COBOL record structures. In general, CICS COMMAREA programs are similar to subroutines in that they are unaware of who invoked them and automatically inherit a transactional and security context managed by CICS. These programs usually focus on integration logic, business logic and data access.

The best practice in CICS application design for a number of years has been to separate key areas of responsibility into programs with clearly defined interfaces, in particular:

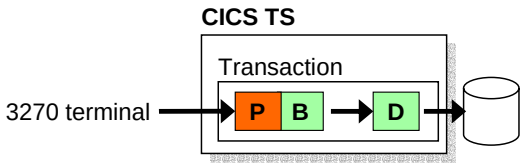
- The 3270 presentation logic,
- Integration logic,
- Business logic, and
- Data access.

This application design ‘pattern’ or framework enables a high-degree of reuse of programs within a larger application, plus re-use with alternative implementations of presentation logic, e.g. for a web browser, a Java client, or SOAP requestor. It also allows each program to be developed and optimized individually for the best return on investment. The good news is that you will find some or many of your application have already been written this way.

However, those programs that were not written to this pattern typically have inter-mixed these responsibilities into, in the worse case, a single program which only has a 3270 terminal-driven interface.

If we turn to slide 8, we see how CICS TS V2 provides functionality that addresses this problem.

Reusing 3270 Presentation logic with the Link3270 Bridge



P B

- However, there remain some programs that combine presentation and business logic
- Link3270 Bridge is a technology in CICS TS that provides a COMMAREA interface to many BMS and terminal-oriented programs
  - Information in the COMMAREA is passed to the BMS application without having to emulate 3270 terminals
  - No changes required to existing BMS application

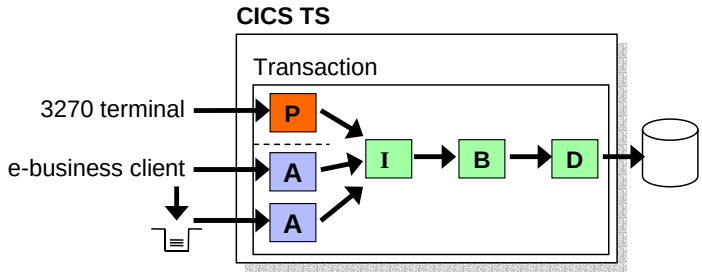
Here we see a program that combines presentation logic (the ‘p’ in the red box) and business logic (the ‘B’ in a green box) into a single program for which there is only a 3270 terminal interface.

CICS TS V2 provides a LINK3270 Bridge function that neatly addresses this problem – it provides an architected COMMAREA interface which may be used to access a terminal-driven program. The LINK3270 Bridge is particularly well-matched to BMS applications. Record formatted data received from BMS is returned as a set of architected “vectors” in a COMMAREA and, similarly, COMMAREA data is mapped to record formatted data for BMS on output. No changes are required to existing application code, and knowledge of 3270 datastreams is not generally needed. Thus the LINK3270 Bridge provides a programmatic interface for an important class of terminal-driven programs, enabling them to be re-used without resorting to a slower networking screen-scraping approach.

It should be noted that this is not a solution to drive all terminal-driven programs and indeed in some circumstances there are better methods to address this situation. The CICS External Interfaces Guide manual gives more information on the capabilities of the LINK3270 Bridge.

If we move to slide 9, we will look at how to gain e-business access to CICS COMMAREA programs or terminal-driven programs that are running with the aid of the LINK3270 Bridge…

e-business access to CICS programs



- Typical e-business clients
  - Web browser
  - Java servlet or EJB
  - Web Services SOAP client
  - C# client in Microsoft .NET
  - WebSphere MQ client
- Adapters
  - An external connector
  - A — An internal adapter (written or generated by tools)
  - A standard IP-based protocol

So, today end-users require the use of a wide range of e-business client types on a wide range of platforms and they wish to access CICS programs. Typical e-business clients include:

•A Web browser,
•A Java servlet or Enterprise Javabean running in a J2EE application server such as WebSphere,
•A Web services SOAP client requester,
•A C# application running in a Microsoft .NET environment, and
•A WebSphere MQ client

The basic principle is that e-business clients access the same integration or business logic used as existing terminal-driven applications. This can involve either;

•An external connector on the client, and / or
•An internal adapter in CICS, and
•A standard IP-based protocol.

For example, a terminal-driven program (‘P’) and a SOAP adapter (‘A’) can access the same CICS integration logic program (‘I’). An adapter is simply a program which accepts the request and converts the data from an external format to the internal COMMAREA format.

An external connector provides a remote call interface and implements a private protocol (equivalent to Distributed Program Link) to invoke an application running under CICS. An external adapter must also be used. This converts data from its external format to the COMMAREA format used by CICS. The most popular example of an external connector for CICS is the CICS Transaction Gateway, which implements a call interface specified by the J2EE Connector Architecture, and is used with external adapters implemented as Java beans.

An internal adapter runs as application code within CICS and receives messages via a standard protocol such as SOAP over HTTP (HyperText Transfer Protocol), or IIOP (Internet Inter-Orb Protocol). The adapter may be implemented in any language supported by CICS and may be independent of the specific protocol used; it de-marshalls data from its wire format and transforms it into the appropriate COMMAREA format.

A standard IP-based protocol - in addition to these techniques, customers may choose to create protocol-specific adapters which exploit a particular transport protocol such as TCP/IP sockets, HTTP or WebSphere MQ. This approach may be the only available option which supports their e-business client and it permits greater flexibility in the functionality which can be implemented. However, this flexibility must be balanced against a loss of generality and flexibility because the adapter can only be used with a specific transport protocol and could require additional code for management and recovery.

Moving to slide 10, we will look at the factors that will influence your choice of access method to connect to CICS…

Which architecture should I use to connect to CICS?

Factors that will likely influence your choice of access method…

- Business factors
  - Eg. Availability of skills
  - Preferred application development environment / tools
- Technical factors
  - Security
  - Transaction commit scope
  - Performance and scalability
  - Architectural limits
  - Synchronous or asynchronous invocation
  - Client to server coupling
  - Data conversion
  - State management

The choice of architectural approach is a *key* decision because it may affect the costs of developing e-business applications and their long term value. However, business factors, such as the availability of skills, may be just as significant as technical factors influencing this decision. It's important to recognize that there is no single right answer, just as there is no right programming language for all applications.

Some technical factors to take into account include…

Security – CICS is used in many sensitive and confidential applications where integrity and accuracy of data are paramount and therefore security measures within an e-business infrastructure are a core business requirement. The most common security requirement is to authenticate end users who access the e-business solution. Simple userid and password authentication is still widely used, although x.501 client certificates, kerberos tickets, and other schemes are becoming popular. Whichever technique is adopted, the user's credentials must eventually be mapped to a RACF userid in order to support the authorization and accounting requirements which normally apply to CICS applications.

Transaction commit scope - This refers to the capability of a given access option to support Local Transactions (one phase commit), enabling a number of updates performed by CICS applications to be processed as a single unit of work, or Global Transactions (two phase commit), enabling an external server to coordinate updates performed by CICS with updates to local resources held by that server.

Performance and scalability - Response time and cost per transaction are important aspects of performance in a production system. CICS seeks to minimize these and is highly optimized for traditional styles of access, such as 3270 terminal access over an SNA network. However, most e-business solutions require additional elements which are less highly optimized, and impose a overhead on the execution of the target business program. This overhead may be characterized by expressing it as a percentage of the base execution cost of the target program, which is normally assumed to be approximately one million CPU instructions for a typical CICS/COBOL/VSAM application.

Architectural limits - Some access options suffer from architectural limits such as the 32 Kilobyte limitation on the COMMAREA or the 4 Megabyte limit in a WebSphere MQ message. At any given time these limits may be relevant to the design of an e-business solution. However, normal IBM practice is to relieve architectural limits when they impact customer applications or operations.

Synchronous or asynchronous invocation - The majority of access options support synchronous invocation, meaning that a client request receives a single reply from CICS and the client waits for the reply. In the alternative approach, known as asynchronous invocation, CICS generates an immediate response confirming that the request has been received, plus one or more deferred responses containing replies to the original request. Typically, the client system continues processing as soon as the confirmation response is received and does not wait for the deferred response(s).

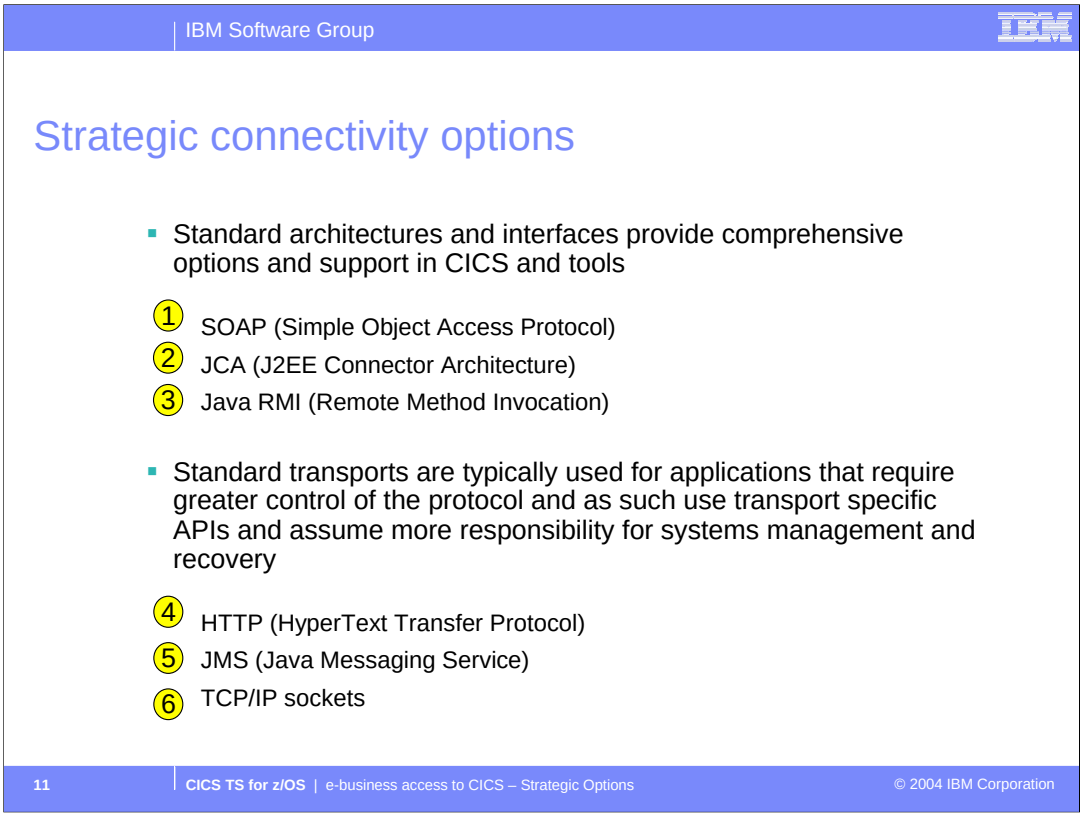
Client to server coupling - Some access options are described as tightly coupled, whilst others are described as loosely coupled. Tight coupling implies that the client must be treated as a single entity for many purposes. Loose coupling is not a precise concept but refers to several possibilities:

•The client may use dissimilar technology from CICS.

•The client need not be maintained or upgraded in step with CICS.

In general, loosely coupled systems are more robust: planned or unplanned outages, software upgrades, and other operational events have less impact on their ability to interoperate.

We are now in a position to examine the strategic connectivity options available as outlined on slide 9…



I have grouped these options into two.

First **standard architectures** and interfaces provide comprehensive options and support in CICS and tools.

- SOAP (Simple Object Access Protocol)
- JCA (J2EE Connector Architecture)
- Java RMI (Remote Method Invocation)

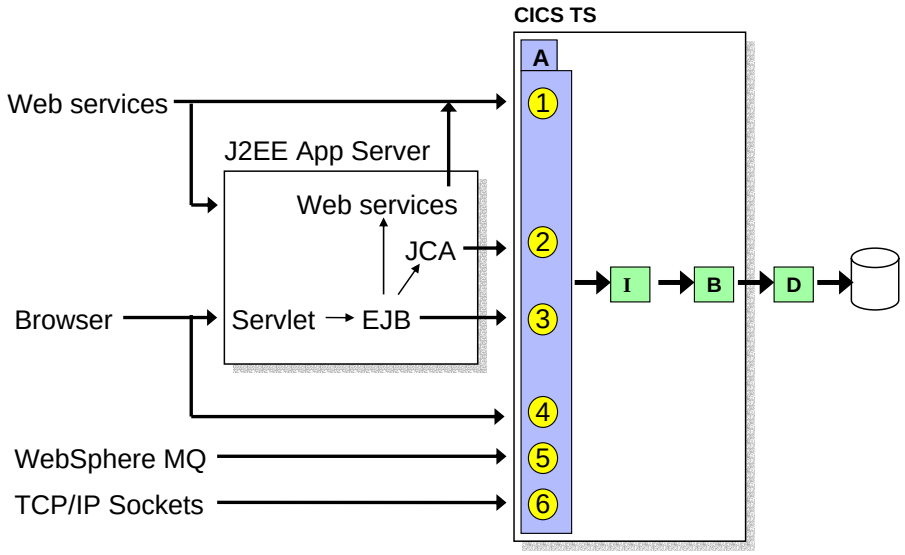
Second, standard transports are typically used for applications that require greater control of the protocol and as such use transport specific APIs and assume more responsibility for systems management and recovery.

- JMS (Java Messaging Service)
- HTTP (HyperText Transfer Protocol)
- TCP/IP sockets

There are others, but those presented here are the prime choices.

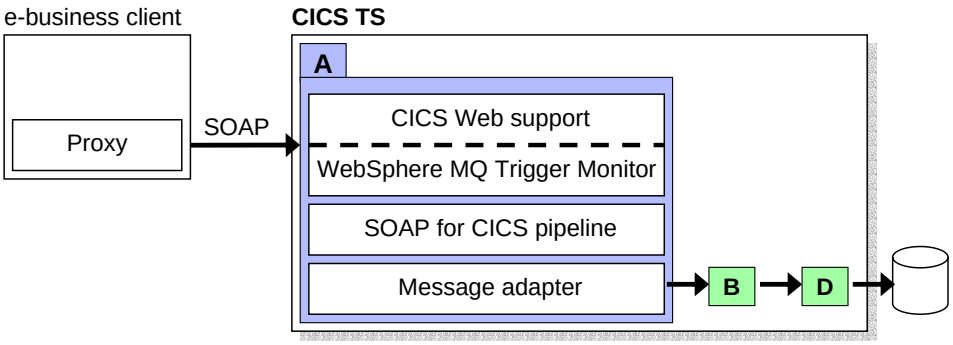
For those who like pictures more than words, slide 12 illustrates these strategic options…

Access to CICS options – strategic connectivity options



OK, so lets take a look at each of these options in a little more detail. On slide 13 we look at Web services and SOAP…

Standard architecture - SOAP



The SOAP for CICS feature enables CICS programs to be a SOAP Web service provider or service requester.

The definition of the SOAP service is defined in a WSDL (Web Services Definition Language) file. Typically tools are used to import the WSDL file and generate a proxy for the e-business client to use to construct and send the SOAP message.

The SOAP request is received by the HTTP listener (called CICS Web support) or the WebSphere MQ trigger monitor for CICS. In both cases the request is passed to the SOAP for CICS pipeline to processes the SOAP architected headers and setup the transaction and security environment and call the target message adapter with the XML message. The message adapter transforms the XML message into a COMMAREA before calling the business logic. Once the business logic completes, the message adapter creates an appropriate XML message which is passed back to the pipeline to be wrapped in SOAP headers and returned to the e-business client.

If you wish to derive the XML schema directly from the COMMAREA layout of the business logic program, the message adapter and the WSDL file can be generated by tools such as the IBM WebSphere Studio Enterprise Developer.

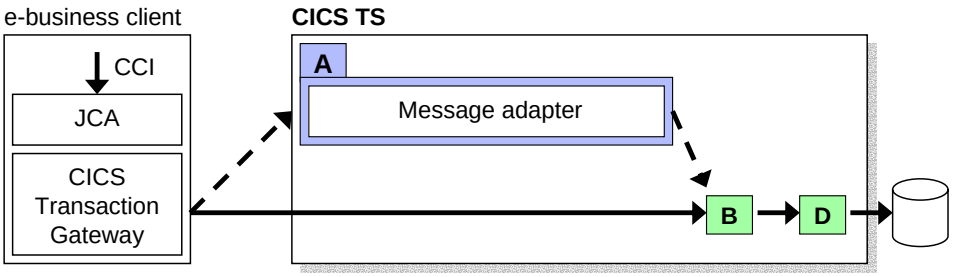
SOAP for CICS allows CICS programs to interact securely and reliably with Web services, independently of platform, environment or application language. Developers can rapidly build open standards based applications independently of the CICS business logic program they will interact with. Hence SOAP is deemed to be a loose coupling architecture.

Continuing to our second architecture covered on slide 14…

The SOAP for CICS Feature

- Objective
  - Demonstrate commitment to SOAP as a strategic access method
  - Support production use of CICS SOAP capabilities for early adopters
- Function
  - Similar to the Tech Preview, plus...
  - Application access to SOAP headers.
  - Application stage may run in transaction and security context determined within the pipeline (eg via headers)
  - 'User' containers in the pipeline.
  - Programmatic application stage program naming.
  - Elimination of BTS IO overhead for HTTP pipeline.
  - Performance enhancements to memory management
- Availability
  - GA'ed 26th September 2003
  - A free feature for CICS TS V2 (ie 2.2 and 2.3).
  - Fully supported (L2/L3), PMRs, APARs, PTFs etc
  - SMP/E installable

Standard architecture - JCA



The J2EE Connector architecture (JCA) defines a standard for connecting the J2EE platform to heterogeneous enterprise information systems, such as CICS. The CICS Transaction Gateway provides the CICS JCA connector as a resource adapter. The resource adapter plugs into the J2EE application server, providing connectivity between the J2EE application, the application server, and CICS. By using the JCA you are assured of seamless connectivity to multiple heterogeneous enterprise information systems, such as CICS and IMS.

The JCA defines a Common Client Interface (CCI) for the e-business client to use to drive interactions. In most J2EE environments the JCA can support transaction coordination (2 phase commit protocol) and the flowing of a user security identity.

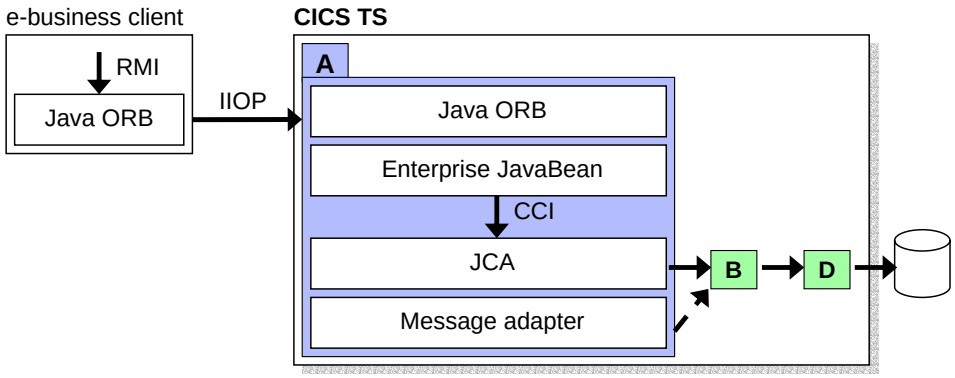
The J2EE application can invoke the CICS business logic program directly if no message transformation is required. In this case WebSphere Studio Application Developer Integration Edition can be used to create a Java bean to represent a COMMAREA formatted as COBOL records, with Java methods for getting and setting fields.

A message adapter in CICS is only required if the message is to be transformed, for example if the request is in XML and the CICS business logic program expects a COBOL record format. The JCA connector provided by the CICS Transaction Gateway is an effective replacement for the CICS ECI Java classes and has a limit of 32K message size.

The JCA is deemed to be a ‘medium’ coupling architecture because messages are typically flowed as COBOL types.

The third architecture, Java RMI, is covered on slide 16….

Standard architecture – Java RMI



Java Remote Method Invocation (RMI) enables e-business clients to invoke methods on remote Java objects across a network.

The e-business client can call a remote object once it obtains a reference to it, either by looking up the remote object in a naming service, such as LDAP, or by receiving the reference as an argument, a return value, or from a cache. RMI uses object serialization to transparently marshal and demarshal parameters supporting true object-oriented polymorphism. CICS supports RMI over IIOP (Internet Inter-Orb Protocol). IIOP optionally supports SSL encryption, full transaction coordination (2 phase commit protocol), and the flowing of a users' J2EE security role.

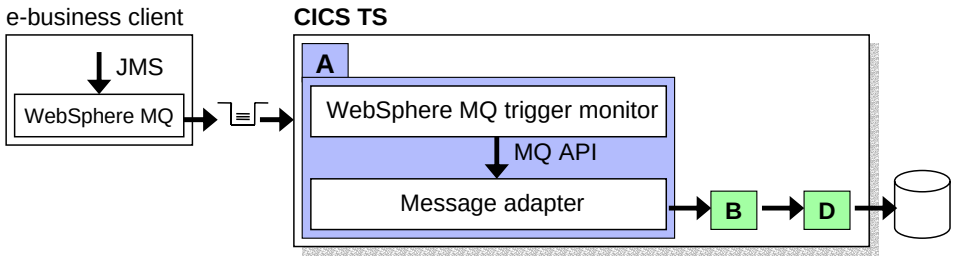
When an IIOP request is received, a Java Virtual Machine (JVM) is started in CICS and the Object Request Broker (ORB) built into the JVM decodes the RMI. The ORB in turn calls the appropriate methods of an Enterprise JavaBean to process the request. All data marshalling is handled by the ORB. If the session bean is to call a business logic program, the J2EE CCI interfaces can be used – the very same CCI as provided by the CICS Transaction Gateway but in this case the JCA is provided by CICS TS and does not involve network flows. If the message needs to be transformed a message adapter could be called prior to calling the business logic.

The session bean in CICS has the option to be stateless between invocations or state-full, in which case CICS will save the session bean data into a VSAM file and restore the state automatically at the next invocation.

The use of RMI is viewed to be a tightly coupled connection because both ends need to be implemented by compatible J2EE technologies and EJB interfaces.

On the next foil, 17, we look at WebSphere MQ and JMS….

Standard transport – JMS/WebSphere MQ (1 of 2)



The Java Messaging Service (JMS) is an integral part of the J2EE platform for enterprise messaging, providing a reliable, flexible service for relaying asynchronous messages and events. The JMS API is a common API and framework that enables the development of portable, message based applications in the Java programming language. The JMS API improves programmer productivity by defining a common set of messaging concepts and programming strategies.

WebSphere MQ provides JMS APIs for use by e-business clients with many options for routing and encrypting messages prior to arriving on z/OS.

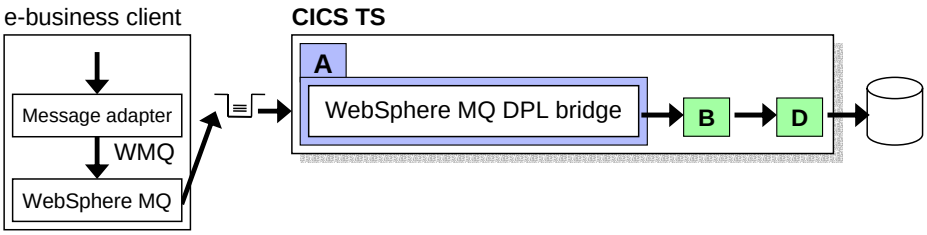
The WebSphere MQ trigger monitor program runs in CICS and, according to the queue definitions, as messages arrive it will start the appropriate message adapter program in CICS as a new transaction. The message adapter will use WebSphere MQ APIs to receive the message, transform it if required, and call the business logic program. The JMS e-business client should not place complex Java objects into the message, but plain text.

A reply message is typically sent using the reply-to queue defined in the message. For efficiency the message adapter program will typically continue to process messages on the inbound queue until it is empty.

Although JMS can be used for pseudo-synchronous messaging, caution needs to be applied to ensure appropriate error handling and compensating business logic programs are in place.

There is another option available that uses WebSphere MQ native APIs that can be seen by turning to slide 18…

Standard transport – WebSphere MQ (2 of 2)

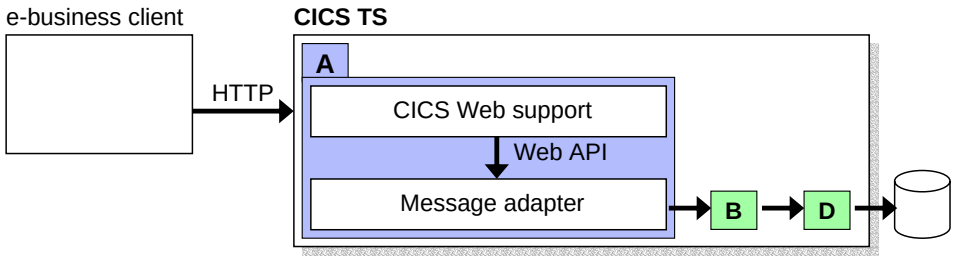


A second option is to use the WebSphere MQ DPL bridge supplied as part of the base WebSphere MQ Server product on z/OS.

This generic adapter will pass a message from a named input queue to a business logic program via the COMMAREA. This is ideal in the situation where the e-business client can format the message into a format acceptable by the business logic program – typically a COBOL record. As you can see, this option involves no programming in CICS.

Progressing to slide 19 we will take a look at HTTP….

Standard transport – HTTP



The adapter component is made up of the HTTP listener in CICS (called CICS Web support) and a message adapter. CICS Web support can accept secure HTTP or SSL encrypted HTTP with client and server certificates.

CICS Web support sets up the transaction and security environment and calls the message adapter. The message adapter uses the CICS WEB APIs to extract the HTTP user data which is typically formatted as an HTML form or MIME data. The message adapter has access to the HTTP and TCP/IP headers if required. The message adapter transforms this information into a COMMAREA and calls the business logic program.

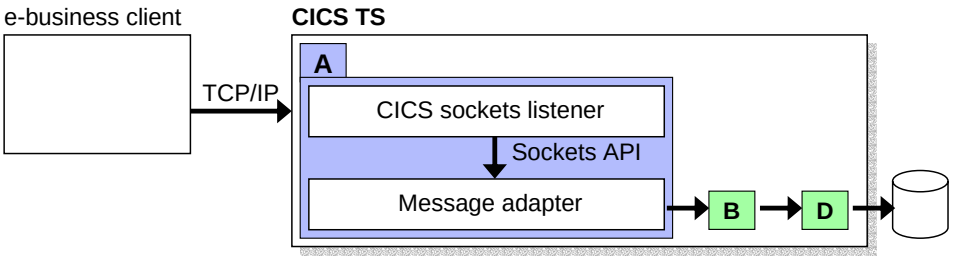
If the e-business client is a web browser, then the response message will typically be formatted as HTML or XML. The message adapter can use the CICS DOCUMENT APIs to easily merge static HTML or XML with dynamic data from the business logic program. The response is returned to the client for display or processing.

The CICS Web support is ideally used together with more traditional forms of static web serving, dynamic servlets and scripting.

HTTP is synchronous and stateless. However, if state management is required CICS does provide a utility for storing state data, indexed by a state management token which the HTTP client will return on subsequent calls in order to retrieve the state.

And last of all we turn to the TCP/IP sockets support in CICS on slide 20…

Standard transport – CICS sockets



The TCP/IP Socket Interface for CICS (referred to as CICS sockets) provided by z/OS Communication Server is primarily intended to support peer-to-peer applications in which both ends of the connection are programmable. CICS sockets provides a variant of the Berkeley Software Distribution 4.3 Sockets interface, which is widely used in TCP/IP networks and is based on the UNIX® and other operating systems.

CICS sockets has a listener which will call the message adapter program when a request is received. The message adapter uses the socket APIs to receive and send data, and perform general communication control functions. The programs can be written in COBOL, PL/I, assembler, or C. Client adapters can be written to create new outbound connections. On the next slide we see a matrix of strategic options for e-business clients to access CICS.

Moving on to slide 21, I have summarised the points raised about the different access methods into a table…

Strategic options Table

e-business client		CICS TS	
Architecture	**Capabilities**	**Interface**	**Coupling[1]**
1. SOAP[4]	SSL HTTP (synchronous) WebSphere MQ (asynchronous)	XML in a CONTAINER	Low
2. JCA	2 phase commit[3] User security context[3] 32K max message size SSL[3] Inbound only Synchronous	COMMAREA[2]	Medium
3. Java RMI[4]	2 phase commit State management User security role SSL Synchronous	Enterprise JavaBean (session bean)	High
4. JMS	SSL Asynchronous	WebSphere MQ API or COMMAREA[2]	Medium
5. HTTP	SSL Secure HTTP Synchronous	CICS WEB API	Medium
6. TCP/IP sockets	Synchronous or Asynchronous	CICS Sockets API	High

This table summarizes the strategic options for e-business clients to access CICS programs. Application requirements should be matched up to the *architecture*, *capabilities, and interface* columns.

In particular the *interface and coupling* columns provide an indication of what is needed in CICS to adapt from the access method to the business logic program. It is recommended for the business logic program to maintain a COMMAREA interface to allow flexibility and in the future to support open standards as they evolve.

Note:

Coupling – ‘low’ indicates endpoints do not share assumptions about implementation technologies, ‘high’ indicates tight integration with many assumptions.

COMMAREA is a message formatted as XML, simple text, or a binary structure such as a COBOL record.

Restriction apply in some environments.

SOAP and Java RMI support is provided in CICS TS V2.

And so if we move to slide 22, I would like to summarize what we have covered…

Summary

CICS is a first class participant in on-demand, e-business applications
- open standard architectures
- Interfaces and transports to maximize reuse
- *Existing* CICS assets can be reused from many e-business clients.

- Why on-demand and e-business is imperative
- Three styles of transformation,
- What can be transformed.
- LINK3270 bridge
- How to architect a solution
- Connectivity options
- Strategic options table

- CICS and WebSphere Application Server are strategic middleware products
  - Interoperate - Web services
  - exploit and complement z/OS qualities of service
  - high availability
  - scalability
  - low cost per transaction,
  - excellent security.

In combination, WebSphere CICS support almost any mission-critical e-business solution.

CICS is a first class participant in on-demand, e-business applications – working with industry open standard architectures, interfaces and transports to maximize the reuse of your mission critical CICS assets. With the right client adapters and interfaces to business logic programs in CICS, your *existing* CICS assets can be reused from many e-business clients.

CICS has lead the way in return on investment whilst minimizing risk. We have covered why it is imperative that businesses adopt the on-demand philosophy and why e-business is integral to this goal. We have looked at the three styles of transformation, and what can be transformed. We have seen how the LINK3270 bridge can assist in this transformation and how we can gain access via e-business to CICS programs. Having embraced this concept, we show how to architect such a solution and the various connectivity options that are available. Finally we have the strategic options table to help provide a matrix of the best options to serve business and allow flexibility for future open standards as they transpire.

CICS and WebSphere Application Server are strategic middleware products that interoperate well using technologies such as Web services and J2EE, to support end-to-end on demand systems. They exploit and complement z/OS qualities of service, such as high availability and scalability at low cost per transaction, with excellent security. In combination, WebSphere and CICS support almost any mission-critical e-business solution.

If you are looking for more information on strategic options for e-business access to CICS the next slide has some pointers to more detailed information…

Further information

- IBM CICS TS – www.ibm.com/cics
  - SOAP for CICS Feature - http://www-306.ibm.com/software/htp/cics/soap/
  - e-business Access to CICS - Using the JCA to integrate WebSphere and CICS
- Redbooks
  - Revealed! Architecting Web Access to CICS
    - www.redbooks.ibm.com/redbooks/pdfs/sg245466.pdf
- OS/390 Guide to e-business Connectors
  - ibm.com/servers/eserver/zseries/library/whitepapers/pdf/gf225124.pdf
- IBM zSeries and S/390 Web-Enablement Overview - Blending your Core Business with the Power of the Internet (August 2001. GF22-5138-03)
  - ibm.com/servers/eserver/zseries/library/whitepapers/pdf/gf225138.pdf
- WebSphere Application Server
  - ibm.com/software/webservers/appserv/was/
- WebSphere MQ
  - ibm.com/software/integration/mqfamily/index.html
- WebSphere Studio Enterprise Edition
  - ibm.com/software/awdtools/studioenterprisedev/

I hope this seminar has been useful and thank you for listening.