IBM Software Group

# IBM WebSphere Technical Conference

Featuring WebSphere, WebSphere Portal, WebSphere BI, WebSphere MQ and CICS – Nov 29th – December 3rd 2004 - Munich

## CI05C - CICS Transaction Gateway v6 - Product update and strategy

Phil_Wakelin@uk.ibm.com, IBM Hursley

WebSphere software

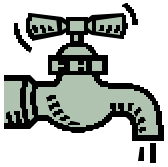
e business on demand software

This version dated: 28-Nov-04

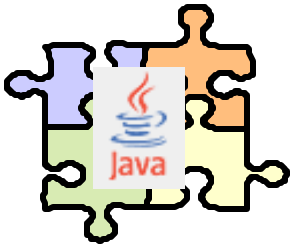
# Agenda

- Connectors and CICS TG
- CICS TG v6 release plan
- CICS TG v6 – what's in the box
  - Performance
  - Systems Management
  - Security (SSL and RACF)
  - Usability
- Future product directions

# What is the CTG?



1. **Plumbing** - Connectivity into CICS
   - Primary inbound connector to CICS
   - COMMAREA (ECI) and 3270 (EPI) based connectors




2. **Interfaces** - Java and non-Java APIs
   - JCA API is strategic and provides enhance QoS
   - Base Java, C, COBOL and COM are stabilised

3. **Integration** - WebSphere and CICS and others
   - All versions of CICS in support (10 currently )
   - 3 versions of WAS on 7 platforms
   - Java + 4 other languages (C, C++, COBOL, COM)
   - 5 SNA servers (AIX, Windows, zLinux)

# CICS Transaction Gateway – Connector Positioning

- Preferred implementation for JCA
  - Access to all CICS servers from WebSphere Application Server.
- Proven characteristics:
  - High performing
  - Secure
  - Scalable
- Ease of installation and flexible configuration
  - Minimal changes to CICS applications
  - No components need to be deployed into CICS runtime
  - Supports all CICS servers and connectivity methods
- Supports a range of Java and non Java clients
  - Including C, C++ COBOL and COM

# Strategic options to access CICS

- *Standard architectures* provide comprehensive development tools and runtime support in CICS

1. SOAP (Simple Object Access Protocol)
2. **JCA (J2EE Connector Architecture) = CICS Transaction Gateway**
3. Enterprise JavaBeans (J2EE EJB)

- *Standard transports* are suitable for use by applications that require greater control of the protocol and do not need the development tools or qualities of service provided by the standard architectures.

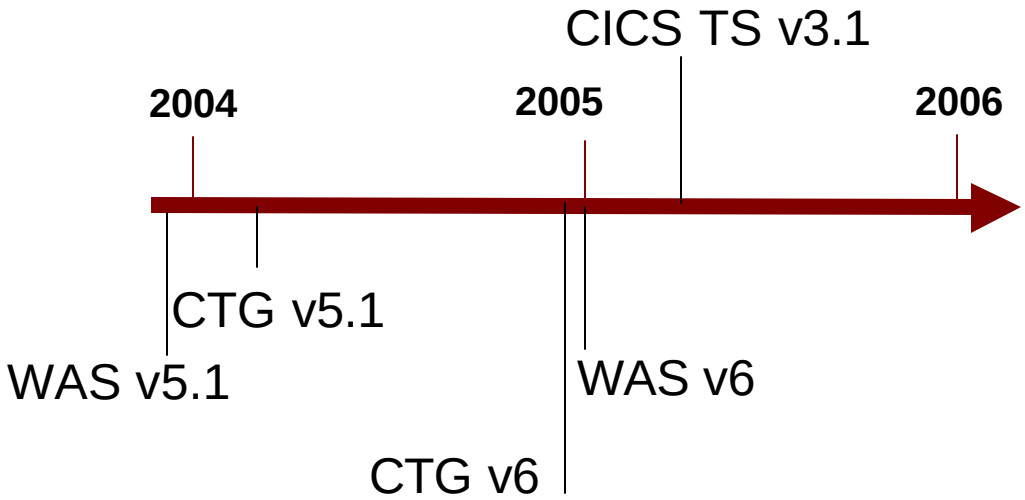
4. HTTP (HyperText Transfer Protocol)
5. WebSphere MQ (MQ APIs or JMS - Java Messaging Service)
6. TCP/IP sockets

Whitepaper ***Delivering e-business access to CICS: strategic options, G224-73240***
http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi

# CICS TG, WAS and CICS TS –release plan

| | |
|---|---|
| CICS TG v6 Announce: | 30th Nov 2004 |
| CICS TG v6 eGA: | 10th Dec 2004 |
| CICS TG v6 GA: | 14th Jan 2005 |

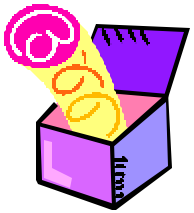
| | |
|---|---|
| CICS TS v3.1 Announce: | 30th Nov 2004 |
| CICS TG v3.1 GA: | 25th March 2005 |

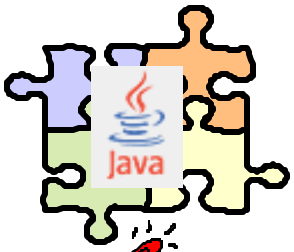



CICS TG v6 includes following products:

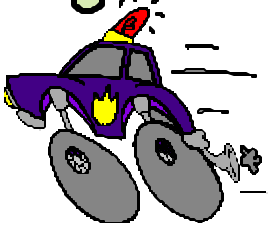
- CICS TG v6 for z/OS
- CICS TG v6 for multi-platforms
- CICS UC v6

# CTG v6 – What's in the box?






- JCA 1.5 and WebSphere v6 interoperability

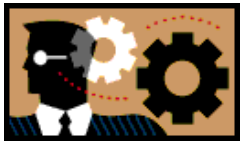


- Performance/Scalability

- Systems Management

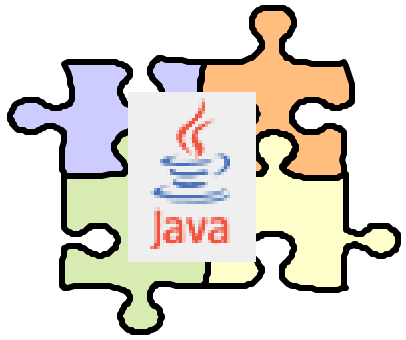


- Security



- Ease of Use

# JCA 1.5 - WebSphere App. Server




- WebSphere App. Server v6 = JCA 1.5 = J2EE v1.4 spec level
- CICS TG v6 ECI and EPI Resource Adapters will support JCA 1.5 only
  - i.e. CICS TG v6 RARs support WebSphere Application Server v6 only
  - WebSphere v5/5.1 customers can use CICS TG v5.1 (JCA 1.0) RARs to talk to a **remote** CICS TG v6 Gateway daemon
  - CICS TG v5.1 RARS (JCA v1.0 ) will be made available for download from the internet
- JCA 1.5 provides
  - J2EE 1.4 Application Server compliance
  - Connection Optimizations
  - For more details refer to talk CI01C
  - Note: There is no plan to provide JCA 1.5 inbound support for the CICS TG
- Tooling support:
  - Rational/WebSphere Studio tooling integration will continue to be provided

# JCA 1.5 Connection Optimisations

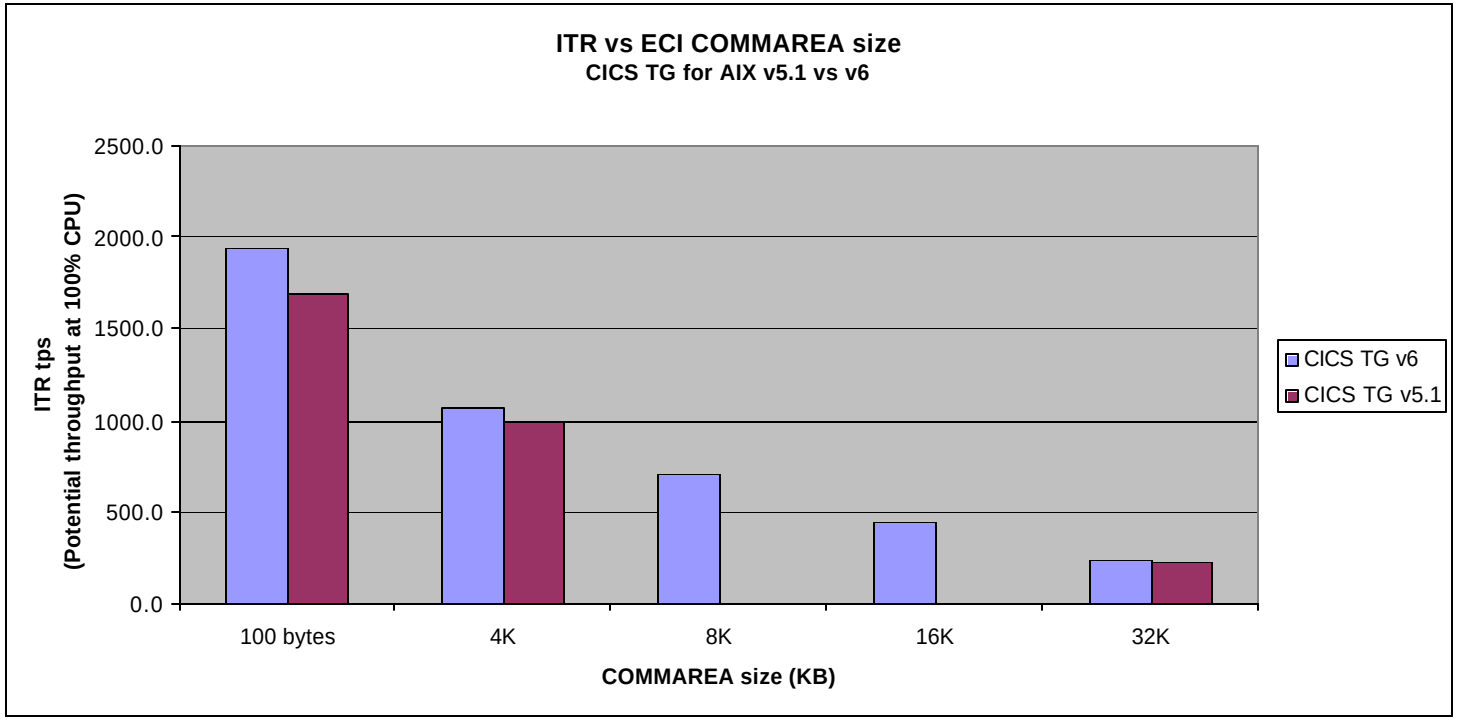
- Lazy Transaction Enlistment
  - Optimizes transaction enlistment calls so that RA only participates in a transaction if it is actually invoked
  - Provides potential better performance if JCA applications are used in an Enterprise Application
  - Not applicable to WAS z/OS local mode (which uses RRS)
- Lazy Connection Association
  - Improved connection pooling in WebSphere
  - Allows get-use-cache model to be efficiently used with WebSphere Connection Pool manager
  - For more details refer to talk CI01C – CICS TG – The WebSphere Connector for CICS

# Performance - improvements

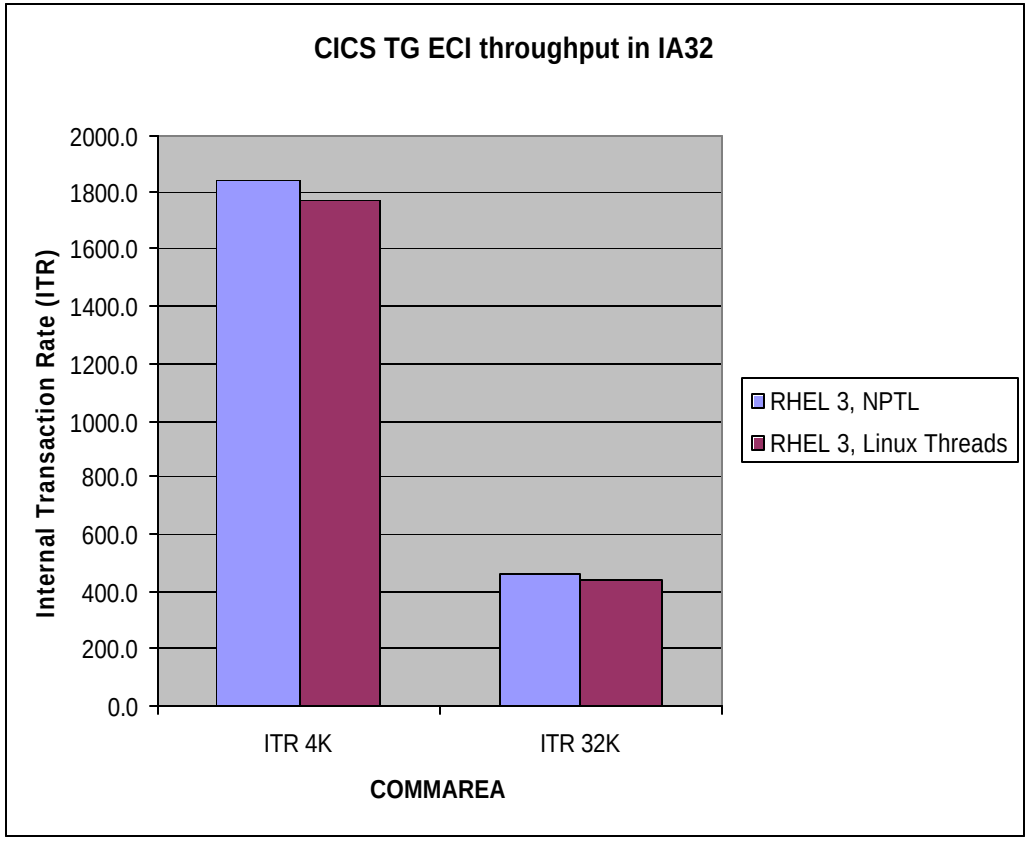
- CTG/CUC on multi-platforms
  - Internal runtime processing improvement
  - NPTL Linux threading model on SLES9 and RHEL3
- CTG z/OS
  - Improved scalability with EXCI pipes
  - Compiler optimizations
  - EXCI null truncation optimization (up to ~200K instructions )
  - zSeries crypto support for SSL handshakes via JSSE
  - zAAP support
- Java/JCA client
  - Optimized data handling when returning data to Java client that contains trailing nulls
- JCA 1.5
  - Architectural connection optimisations
    - Lazy transaction enlistment
    - Lazy connection association

# Performance – Windows/UNIX/Linux

- Improved runtime tracing enhancements within Client daemon
- Applicable to all platforms (Windows/UNIX/Linux)
- Gives ~15% saving in CPU usage



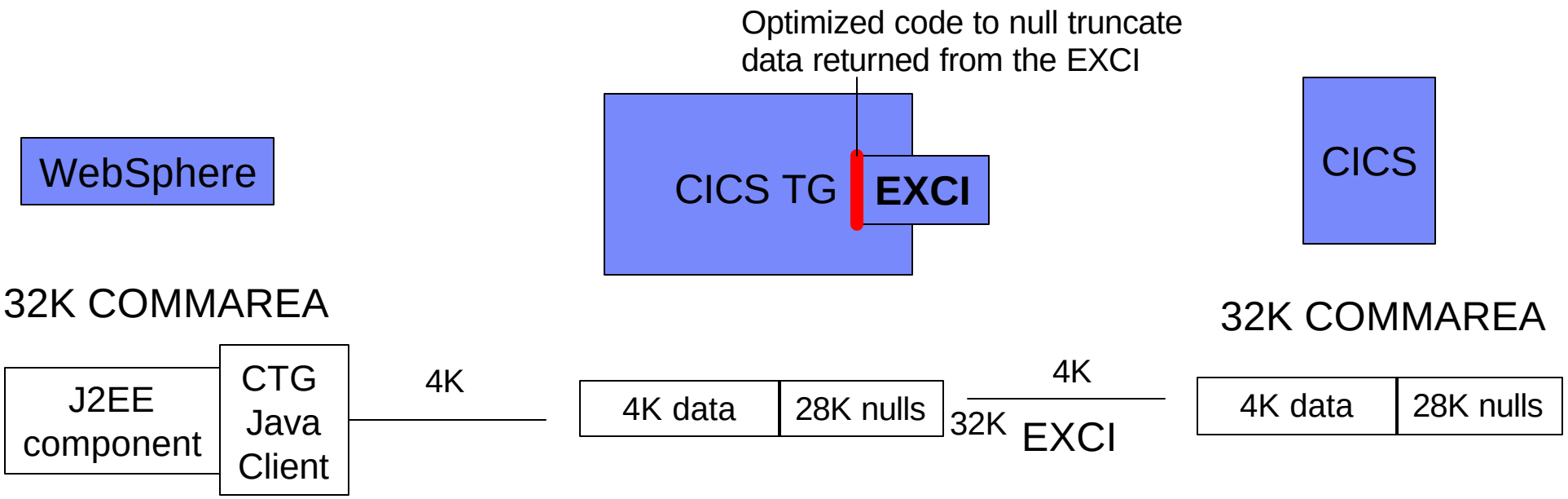
# Performance - Linux




- NPTL gives improved performance over Linux Threads out of the box (4-5%)
- NPTL is more scalable at high thread utilisations (many 100s)
- RHEL 3 and SLES9 both support NPTL threading model
- CICS TG supports:
  - SLES 8 and SLES9
  - RHEL 3
  - Linux/Intel and Linux on zSeries

Benchmark: ECI workload into CICS TG/AIX and CICS TS v2
Test hardware: Intel P4, 2.8Ghz, 1CPU, 2GB RAM

# Performance – z/OS

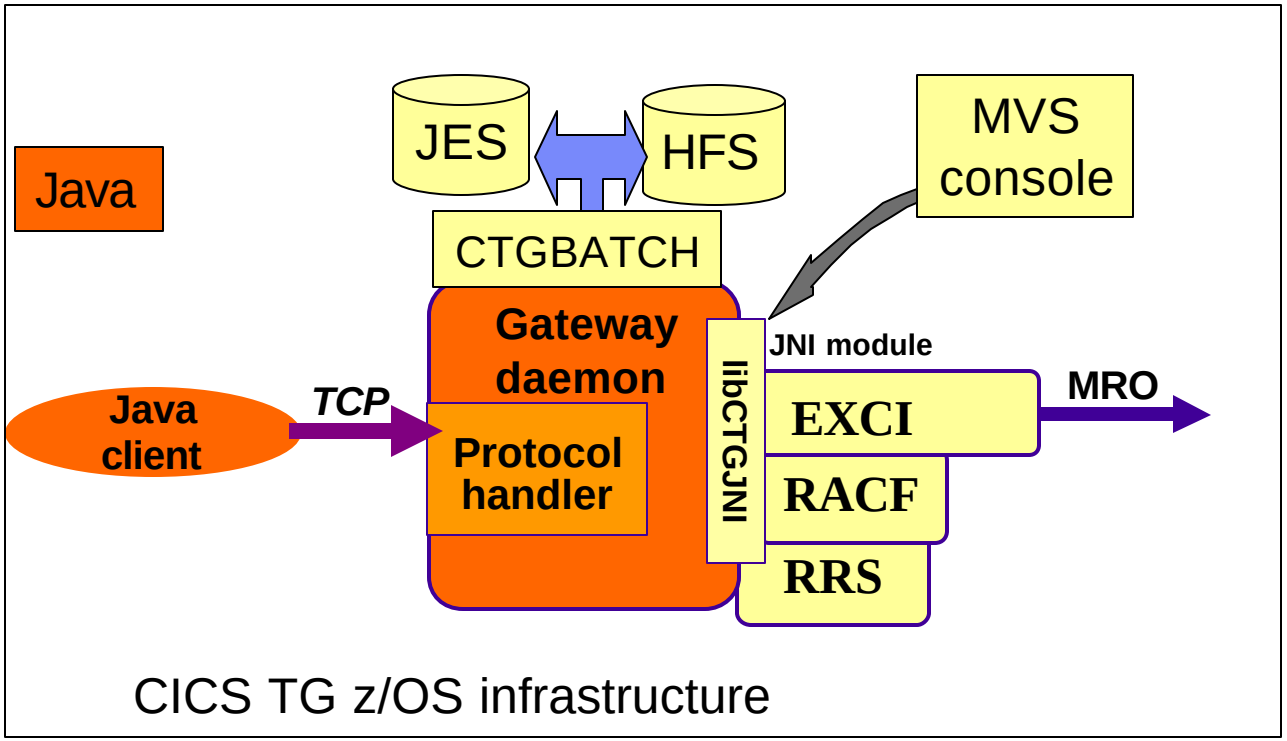



- Many applications are written to be data size agnostic – so just flow 32K data areas into CICS
- CICS, the EXCI transport and the CICS TG have automatic optimisations to remove trailing nulls from an ECI COMMAREA flowed into or out of CICS
- CICS TG v6 has optimised code within CICS TG Java/EXCI interface
- Maximum saving on a 32KB COMMAREA with minimal non-null data is ~220K instructions
  - Equates to max 40% saving in CICS TG code

# Performance – z/OS

- General improvements
  - Improved garbage collection in CICS TG Java code
  - More highly optimized JNI (C) code
    - Max saving of ~10K
- Support for zAAP offload in z/OS V1R6.
  - Potential for ~50% offload of CICS TG instructions to zAAP CP

# Performance – z/OS zAAP

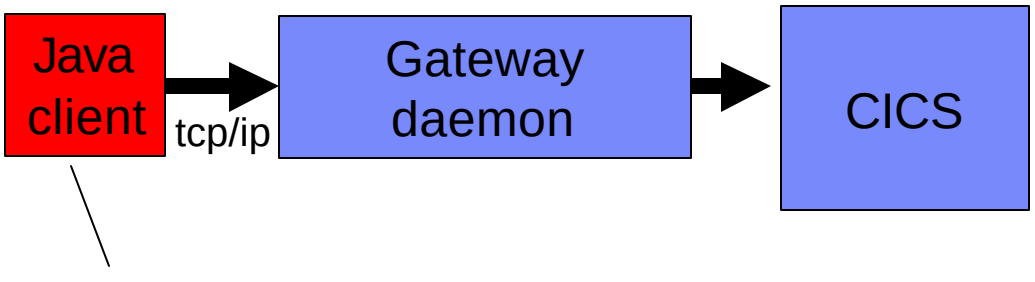


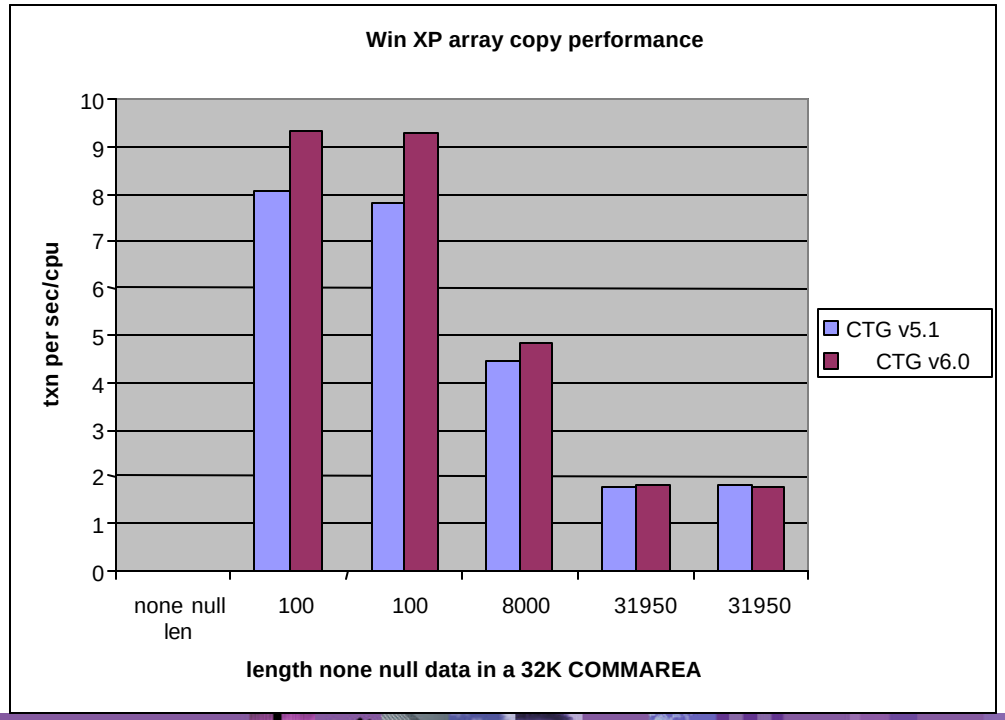

CICS TG z/OS infrastructure

- ~50% offload to zAAP CP (IFA) in early benchmarks
  - ECI workload using Gateway daemon
- For more detail refer to WSC White paper:
  *z/OS Performance: Capacity Planning Considerations for zAAP Processors*

- zAAP - Java off-load processors
  - IBM Java SDK 1.4.2
  - z/OS V1R6
  - Aim: reduction in CPU cost of Java processing, without affecting throughput
- CICS TG for z/OS
  - Mix of Java, and native code (C and assembler code)
- Java:
  - Inbound socket handling
  - Thread handling
  - Logging
- Native:
  - Pipe allocation, data marshalling
  - Logging and I/O
  - CICS EXCI libraries
  - RACF
  - RRS

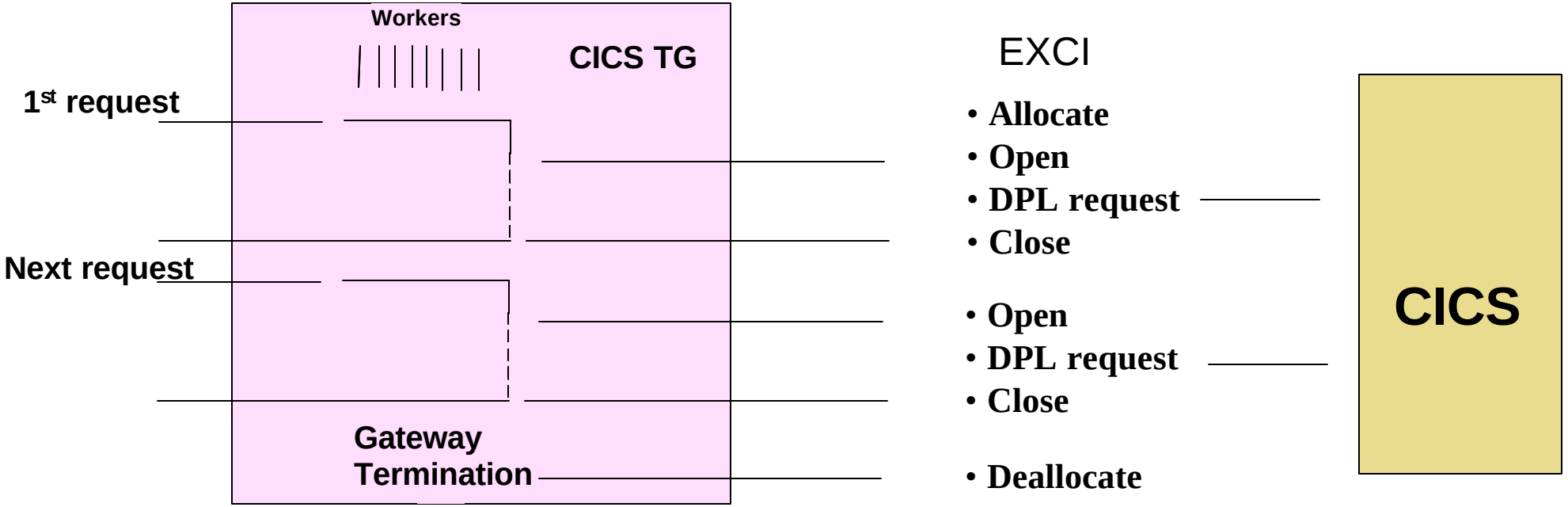
# Performance – Java client




- Client side CICS TG Java code optimisation (ctglient.jar)
- Affects all platforms and JCA and or non JCA
- Benefits WebSphere applications using CICS TG in local or remote mode
- Improved data handling when returning a datastream containing trailing nulls
- Provides benefit to CICS application returning data shorter than the original COMMAREA length
- Up to 12% CPU savings measured on Windows, AIX
- Up to 150K instructions saving measured on z/OS

# EXCI pipe scalability




- CICS limits max of 100 allocated EXCI pipes per calling address space
- Current CICS TG design caches allocated pipes
- Each pipe has an affinity to the calling thread

- This gives high performance but additional requirement raised for
  - Predictable usage pattern
  - Better diagnostics
  - More throughput per region

# EXCI pipe scalability - 2

- **CTG v6 feature - EXCI Pipe usage**
  - Improved diagnostics for pipe information messages
    - Messages on startup to inform of EXCI limit
    - Messages at thresholds to warn as limit approached, and breached.
  - Provide new environment variable (CTG_PIPE_REUSE=ONE) for *one pipe per thread*
    - Existing allocated pipes are de-allocated if the thread needs to allocate to a new APPLID
    - Can be used with Gateway daemon or WebSphere App. Server on z/OS.
    - Predictable usage: Ensures max number of pipes is <= number of threads
    - Potential performance cost, if frequent pipe switching goes on (partially addressed in CICS TS v2.3)
- **CICS TS v2.2 and 2.3 APAR PQ92943**
  - New user modifiable LOGONLIM in SYS.PARMLIB for each MVS image
  - Default will be 100, maximum will be 250 pipes per EXCI address space
- **CICS TS v2.3**
  - Improved performance of EXCI pipe de-allocate in CICS TS v2.3 (~20%)

# Systems management

- Quiesce support
  - Normal and immediate shutdown options
  - Normal shutdown allows running transactions to terminate
- Improved administration tool
  - Integrated with dynamic trace and shutdown functions
  - New ctgadmin command line interface
- Ability to run as a Windows service or UNIX daemon
  - Standard UNIX script provided (ctgd)
- Improved file handler for Gateway daemon log
  - Manages wrapping and max file size of Gateway daemon log
- APPC support on zSeries
  - Improved connectivity for CTG on Linux on zSeries

# System management – z/OS

- SDSF administration interface
  - New z/OS console administration interface
  - Shutdown and trace admin functions available from z/OS console
- Logging to JES
  - New CTGBATCH shell launcher supports logging to JES (SYSOUT=*) or any DD.
  - Easier to configure single address space
  - Improved diagnostics when running CTGBATCH
- Improved support for multiple Gateway regions
  - ctgenvvar configuration file does not need to be shared amongst all Gateway regions
  - ctgenvvar can be migrated to STDENV DD member

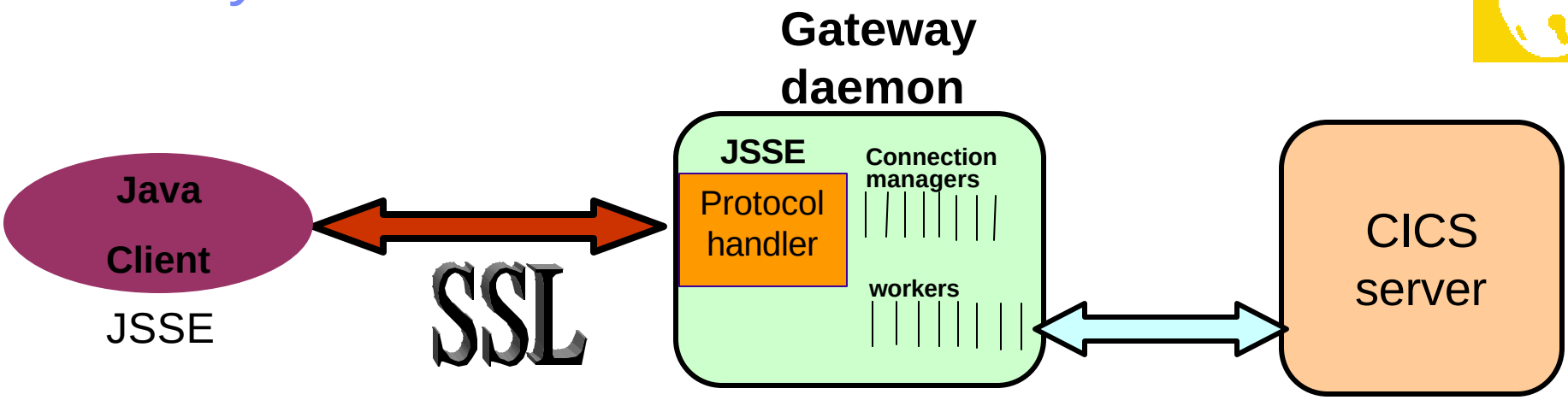
# Security - JSSE



- JSSE is now the strategic SSL implementation for the CICS TG
- Following new function has been added in v6
  - RACF keyring support on z/OS
  - zSeries hardware crypto support on z/OS
  - SSL Ciphersuite select option
  - Improved diagnostics

- Note: SSLight and SystemSSL no longer supported by CICS TG on any platform (previously announced in CICS TG v5.1 announce letter)

# Security - JSSE




- SSL support in CICS TG now implemented by Java Secure Sockets Extension (JSSE)
- Provides following advantages:
  - SSL support provided by JSSE in Java SDK, 1.4.2
  - Wide range of bulk data ciphers (40, 128, 256-bit)
  - SSL cipher suite in use can be controlled using new ciphersuiteselect parameter in CTG.INI
  - Crypto support for handshakes on z/OS
  - Support for range of keytools: iKeyman, keytool and RACF
  - SSL v3 is supported (no support for TLS v1)

# JSSE ciphers



- J2SE in IBM SDK 1.4.2 supports wide range of cipher suites i.e.:
  - **SSL_DHE_RSA**_WITH_***AES_256_CBC_SHA***
    - ***Text after*** WITH is the cipher for bulk data encryption
    - **Text before** WITH is the handshake cipher
- Bulk data ciphers:
  - AES, DES, Triple DES, RC2, RC4  40, 128 and 256-bit variants available
  - IBM SDKs are limited to 128-bit ciphers, policy files are available to upgrade to the 256-bit ciphers
  - Update $JAVE_HOME/lib/security/security.policy
    - https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=jcesdk
- Signature algorithms
  - SHA1 with RSA, MD5 with RSA, and MD2 with RSA signatures
  - SHA1 with DSA signature
  - Anonymous ciphers (***anon***) are not supported by default JSSE security policy
- Message digest algorithms  - provide data integrity
  - SHA1, SHA2, SHA3, SHA5, MD5, MD2
    - Cipher suite names ending with _SHA indicate indicate SHA-1 algorithm used

# JSSE – zSeries cryptographic support



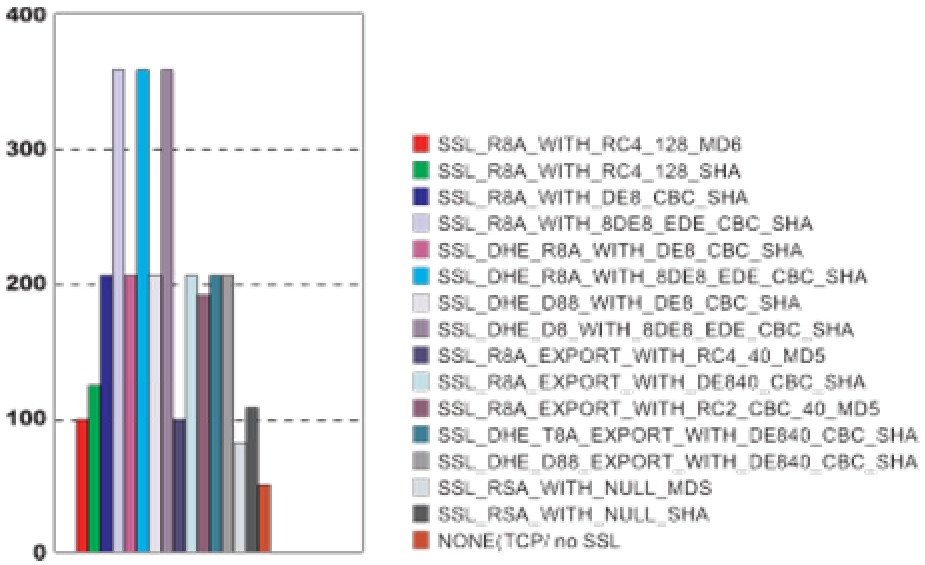
- IBMJSSE supports the IBMJCE4758 JCE provider on z/OS
  - Provides ability for CPU off-load for SSL handshakes

1. Requires addition of IBMJCE4758 update to Java security.policy file:

```
security.provider.1=com.ibm.crypto.hdwrCCA.provider.IBMJCE4758
security.provider.2=com.ibm.crypto.provider.IBMJCE
```

2. Key generation tools supported
   - hwkeytool - .jks keystors in USS/HFS
   - RACF – for RACF keyrings

# JSSE performance

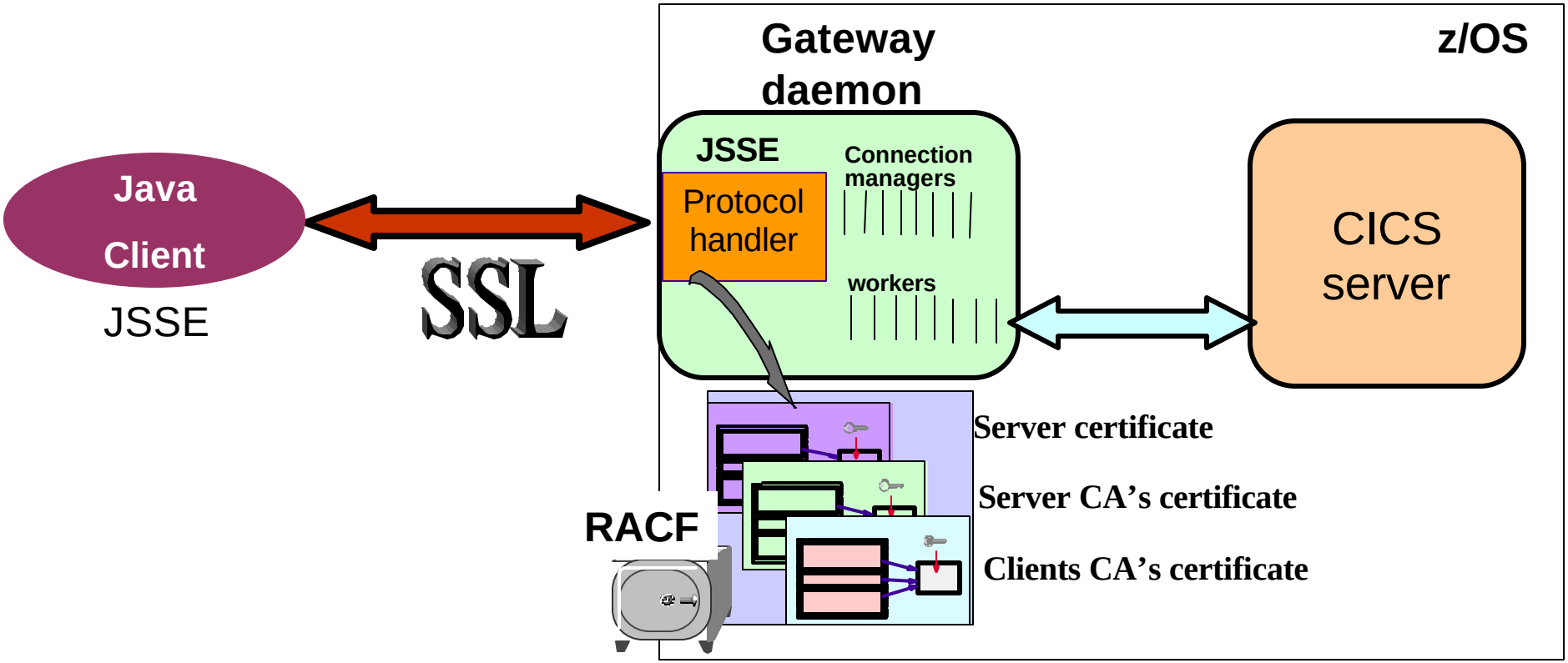



**Performance tips:**

1. Reduce number of SSL handshakes
   - Use JCA connection pool
   - Or re-use connections
2. z/OS: SSL handshakes can be off-loaded using crypto engines
3. Larger keys (512/1024/2048) have higher handshake cost
4. Chose appropriate bulk data ciphers:
   - SHA-1 is considered a stronger hash, MD5 provides better performance
   - RC4 has a low CPU cost in software vs DES
   - RC4 stronger ciphers (128 vs 40) do not require more CPU cycles

Figures are for JSSE bulk data encryptions
See WAS v5.1 infocenter article *Secure Sockets Layer performance tips*

# Security – RACF keyrings




- SSL keystores can now be stored in RACF database
- Access can be restricted to the Gateway region user ID
- Enable using option `esmkeyring` in CTG.INI
- Certificates can be created using RACF SDSF panels

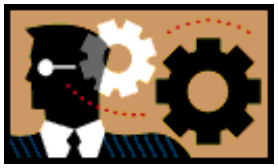
# JSSE - acronyms



- RSA
  - Public key algorithm developed by Rivest, Shamir and Adleman
  - Requires RSA or DSS key exchange
- DH
  - Diffie-Hellman public key algorithm
- DHE
  - Ephemeral Diffie-Hellman public key algorithm
  - Diffie-Hellman parameters are signed by a DSS or RSA certificate, which is signed by the certificate authority (CA)
- DSS
  - Digital Signature Standard, using the Digital Signature Algorithm for digital signatures
- DES
  - Data Encryption Standard, a symmetric encryption algorithm
  - Block cipher, high performance cost without crypto support
- RC4
  - A stream cipher designed for RSA
  - Variable key-size stream cipher with key length from 40 bits to 128 bits
- AES

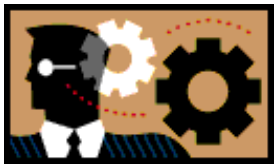
Full range of ciphers: http://www-106.ibm.com/developerworks/library/j-ibmsecurity.html?ca=dnt-541

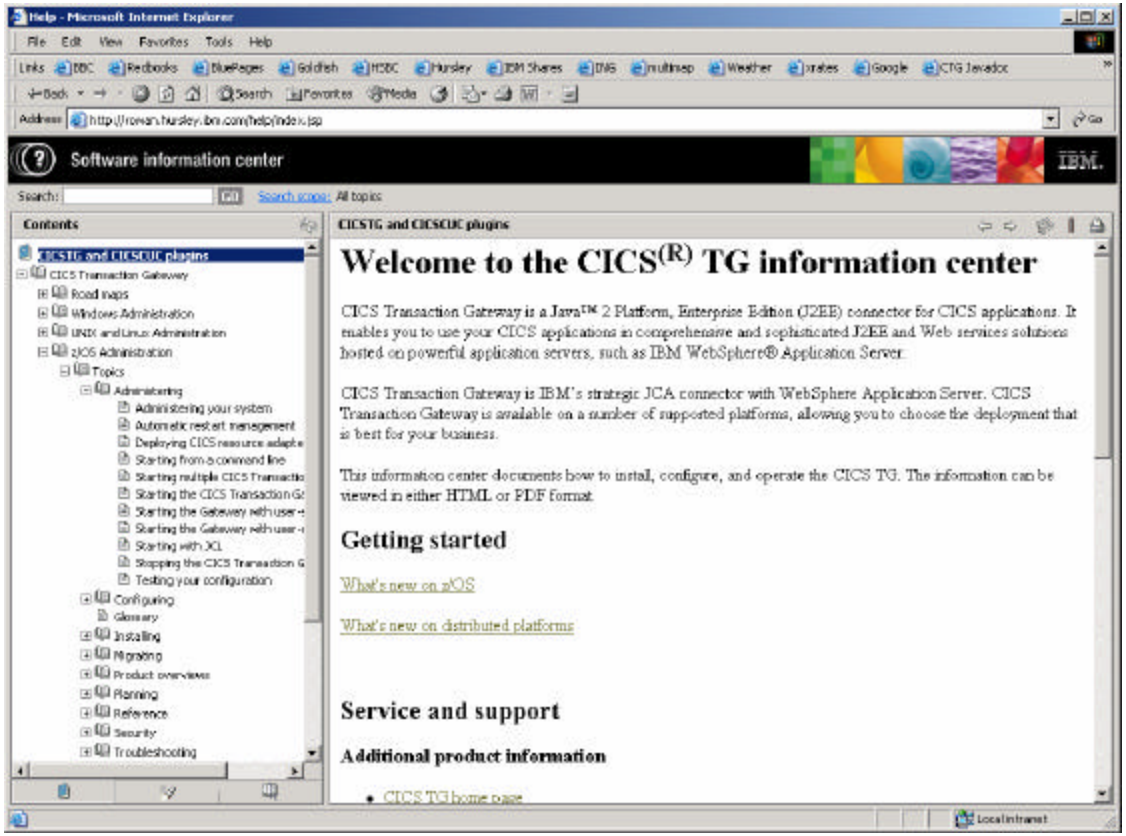
# Ease of use



- **Simplified installation and migration**
  - SMP/E standard install on z/OS
  - ISMP standard install on Unix/Windows platforms
- **Miscellaneous usability enhancements**
  - Lower case configuration files
  - Precompiled EAR samples for WAS
    - Also supplied in CICS support pack CA5C for WAS v5.1/CTG v5.1
  - CTG_JNI_TRACE_ON environment variable
    - Controls activation of JNI trace at startup
  - New Programming Guide

# Ease of use - Infocenter





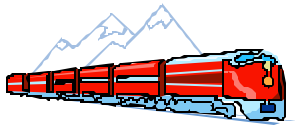

- Highly searchable
- Zip file included on media
- Easy integration with other IBM infocenters
- Both book and task views
- Plugs into any existing eclipse viewer
- PDF documentation still provided

View online: http://publib.boulder.ibm.com/infocenter/cicstg60/index.jsp

# CICS TG future development themes:

1. Interoperation and standards
   - Standards are key to interoperability
2. Customer value
   - Architectural limits
   - Systems Control
   - Log and Trace
   - APIs

# CICS TG – Trends and directions



Interoperation/Standards:

- 64-bit
- IP v6
- eWLM
- Other J2EE Application Servers

Architectural limits

- Global transaction support/2PC
- 32KB COMMAREA relief/Link with containers
- Improved failover
- TCP/IP network hardening

Systems Control

- Systems monitoring of Gateway daemon
- Improved logging and trace

# Other CICS TG talks

- **CI01C - CICS Transaction Gateway - The WebSphere Connector for CICS**
  - WebSphere and CICS integration with the JCA, P. Wakelin
- **CI02C - CICS Transaction Gateway and CICS Universal Clients: Introduction and Overview**
  - Introduction to the CICS TG and CICS UC, P. Masters
- **CI03C - Connecting Remote Gateway and Clients to CICS TS**
  - Configuring clients to connect to CICS, L. Compton
- **CI04C - Implementing the CICS Transaction Gateway on z/OS**
  - CICS TG for z/OS for sys progs, L. Compton
- **CI05C - CICS Transaction Gateway: Product Update and Strategy**
  - CICS TG v6 new functions and strategy, P. Wakelin
- **CI06C - CICS Transaction Gateway: Introduction to the JCA**
  - JCA overview, P. Masters
- **CI07C -Transaction Gateway for z/OS: Problem Determination**
  - Debugging and advanced configuration for CICS TG z/OS, P. Wakelin
- **CI08C - CICS and WebSphere Interoperability: An Overview**
  - CICS and WAS interoperation choices, L. Compton