

F02 The New DB2 Java Client

Curt Cotner, Distinguished Engineer, DB2 for z/OS Development, IBM

This presentation provides a sneak peak at the new Java client that provides JDBC and SQLJ support for the latest DB2 UDB products. This new Java client is designed to significantly improve application portability, performance, and the functionality of both JDBC and SQLJ.


F02

The New DB2 Java Client

Curt Cotner

DB2 for z/OS Development

cotner@us.ibm.com

A horizontal banner with a green background and a purple border, surrounded by green circles of various sizes. The text "IBM Data Management Technical Conference" is written in purple on the banner.

IBM Data Management Technical Conference

Anaheim, CA

Sept 9 - 13, 2002

What is DB2 Common Client?

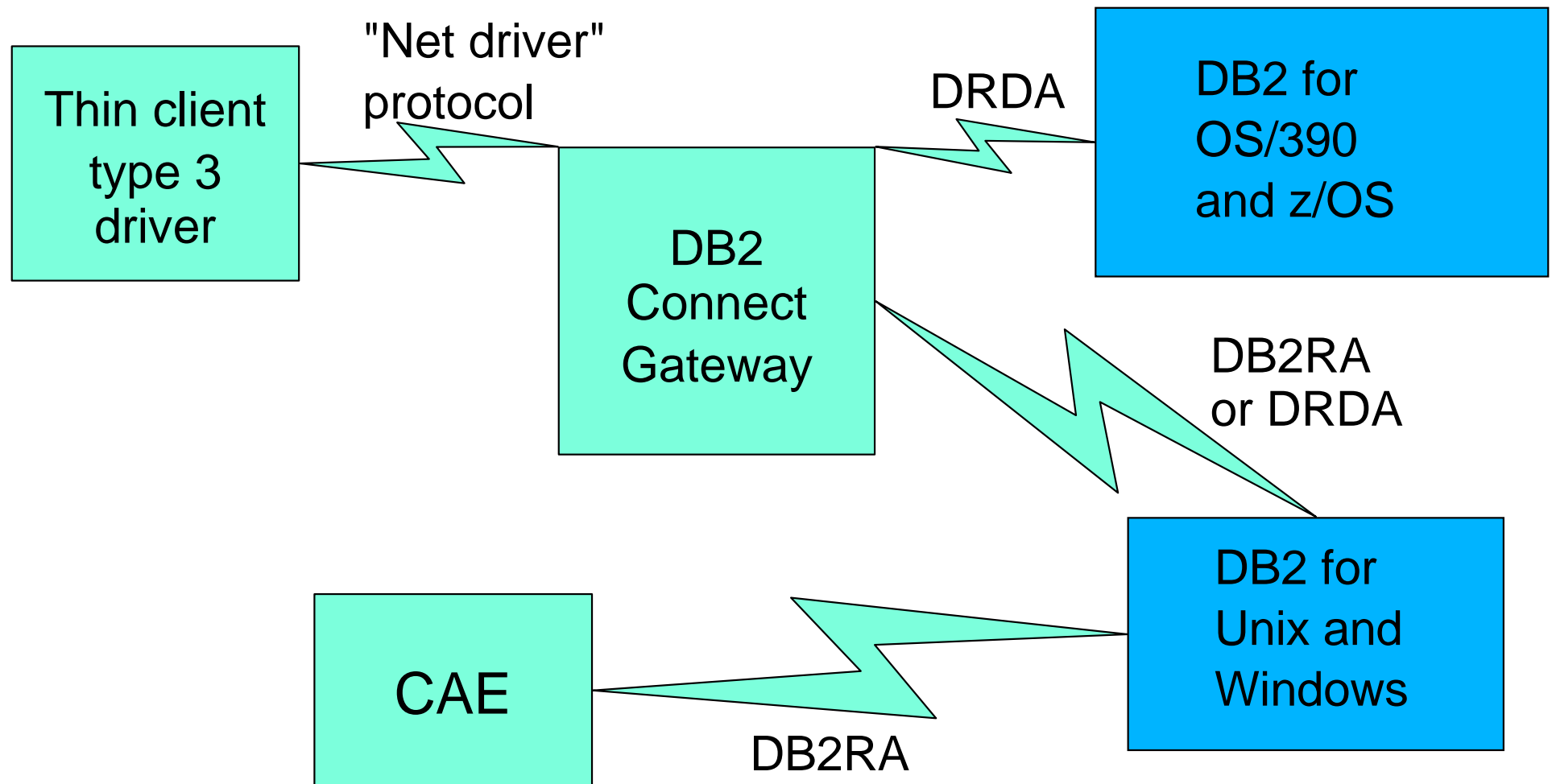
- Significant reengineering of DB2 Connect and CAE
- Uses DRDA protocols for all client communication
 - ▶ eliminates DB2RA and net driver protocols
- Improved DB2 Connect and CAE consistency/performance
 - ▶ much higher percentage of common code
 - ▶ fewer unique code paths for specific hardware configurations
- Several significant improvements to DRDA
 - ▶ support for long SQL names and statements
 - ▶ DRDA query block sizes can now be up to 2M bytes
 - ▶ rely on server-supplied stored procedures for SQL error messages, database metadata, etc.
 - ▶ many internal performance improvements



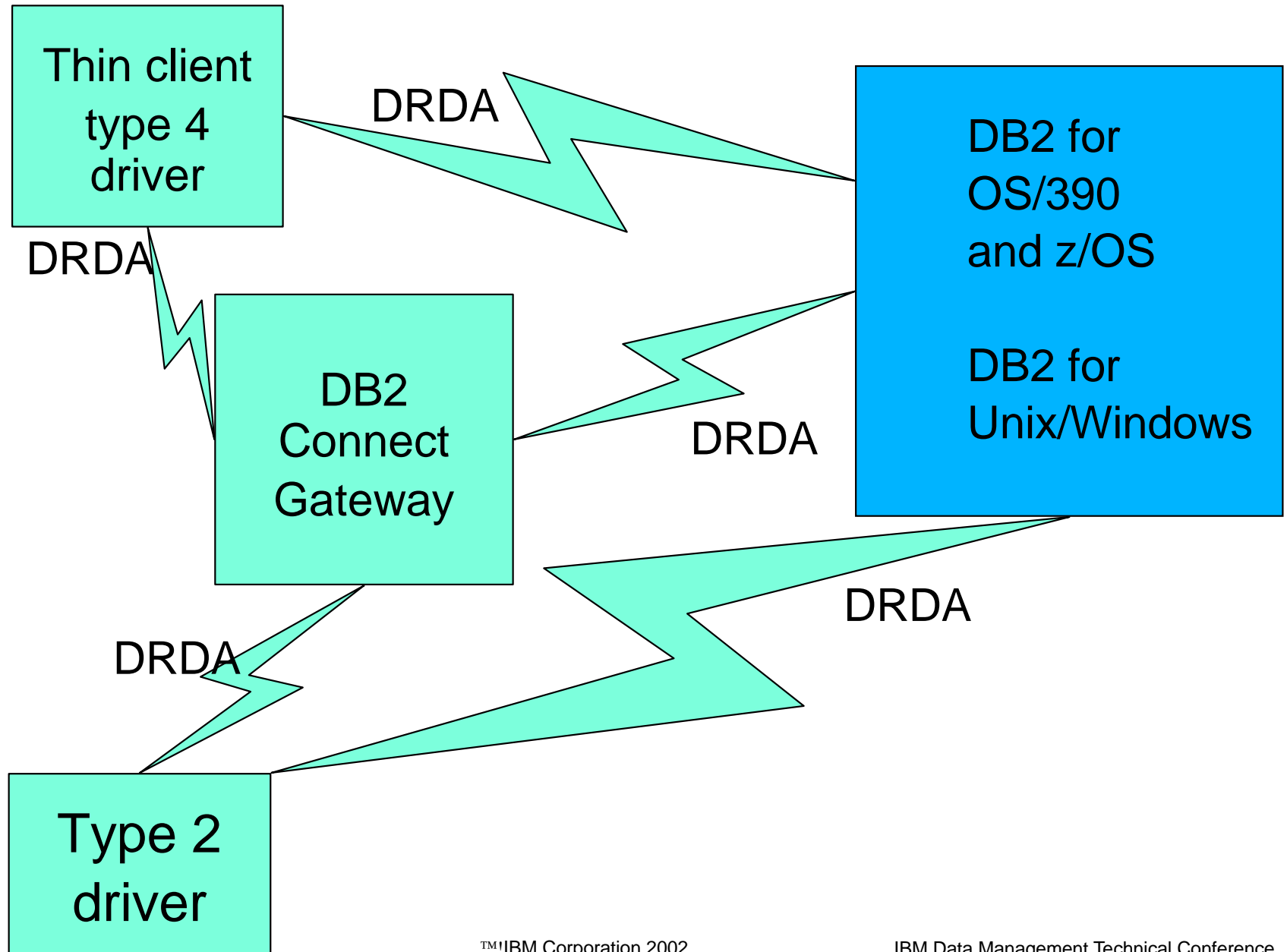
Objectives for new Java Client

- Single driver for Unix, Windows, and OS/390
 - ▶ eliminates major cause of Java porting problems
- Provide fully compliant JDBC 2.0 driver
- Improved Java driver integration with DB2
- Simplify install/deployment of Type 2 driver
- Provide 100% Java application development process for SQLJ
- Improve JDBC and SQLJ performance significantly
 - ▶ SQLJ is now faster than JDBC on Unix/Windows
- Type 4 driver for thin clients
- Trace improvements
 - ▶ dynamically turn trace on/off
 - ▶ multiple levels of trace detail

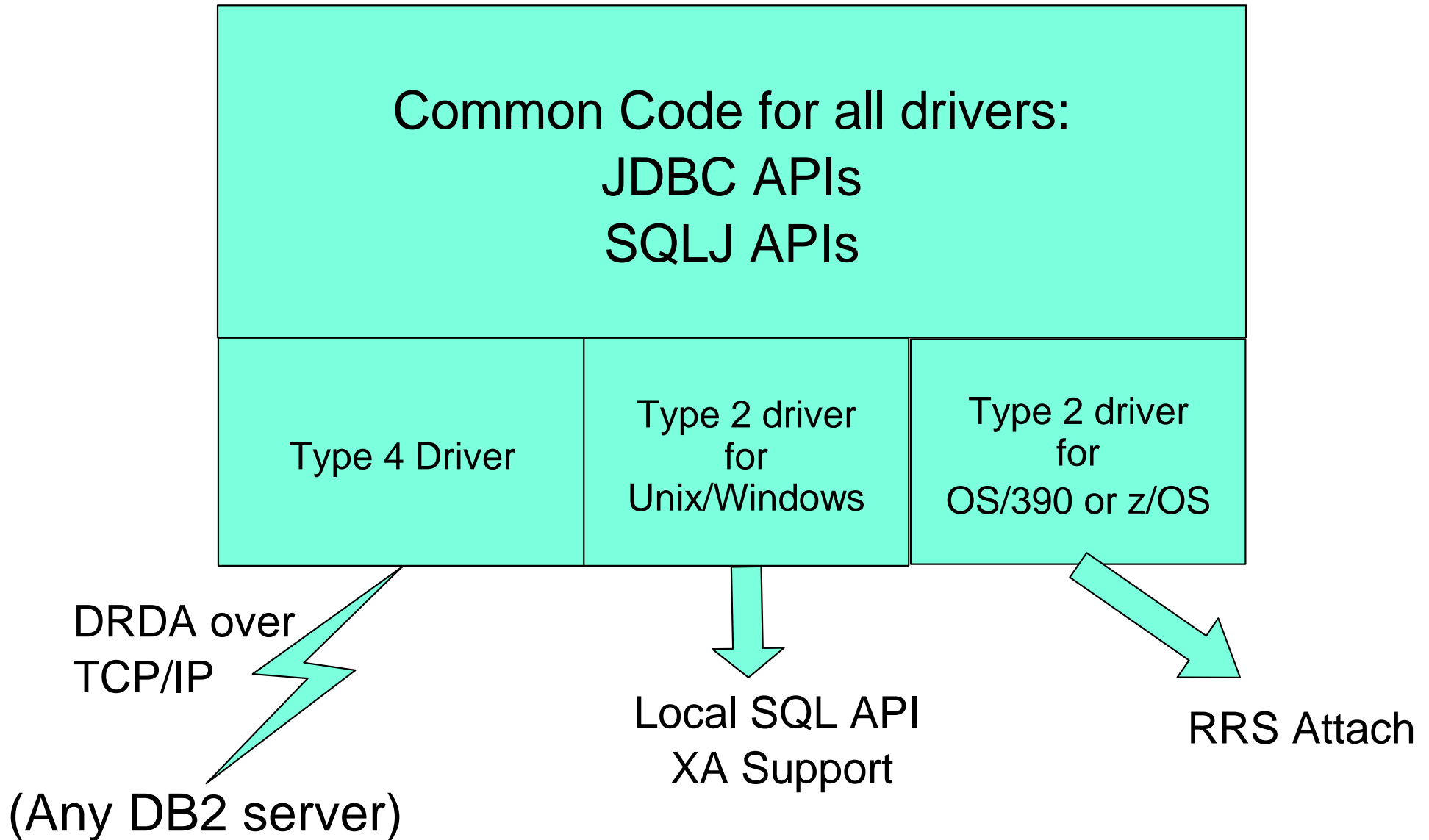
Current DB2 UDB Java Client



New DB2 UDB Java Client



DB2 Java Client Internal Architecture



JDBC driver types

- Java defines 4 types of JDBC drivers
 - ▶ Type 1 -- implements JDBC as a layer on top of ODBC
 - ▶ Type 2 -- uses native method calls (JNI) to call DLLs that issue the SQL (the DLLs are specially written for JDBC)
 - ▶ Type 3 -- network driver that is database vendor independent
 - ▶ Type 4 -- network driver that speaks the database's native network protocol
- Type 1 drivers generally have more CPU overhead than type 2
- Type 3 and type 4 drivers must route through a network layer, so they are not efficient for local JDBC connectivity

DB2's type 4 JDBC driver

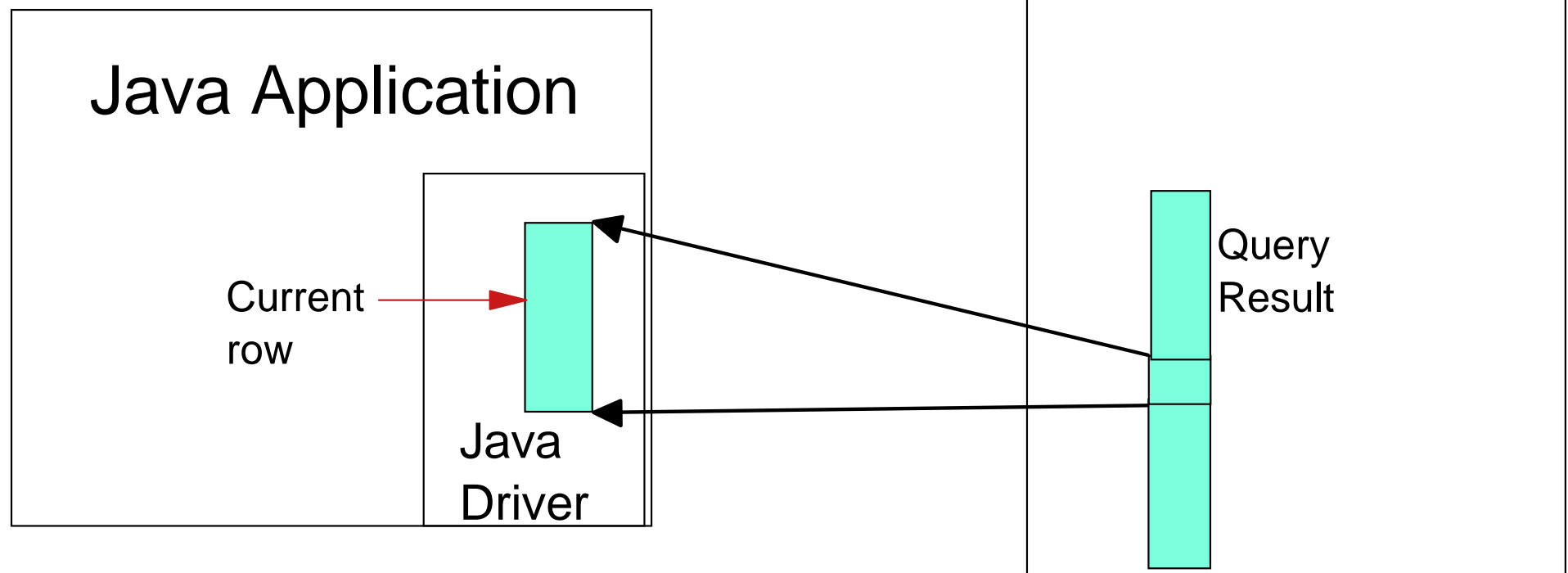
- Targeted at clients that fit this profile:
 - ▶ prefer to not install the JDBC driver on their machine
 - ▶ don't need high-end performance/scalability/availability
- The type 4 driver will NOT support:
 - ▶ JTA (2--phase commit)
 - ▶ parallel sysplex workload balancing
 - ▶ connection concentrator functionality
 - ▶ static SQL profiling
 - ▶ Query Patroller
 - ▶ some of the performance optimizations in the type 2 driver
 - ▶ full range of code page translations
- EJBs or high-end application servers should use the type 2 JDBC driver.

New Java API Enhancements

- Scrollable cursor support
- Batch updates
- Improved security for DB2 authentication
- Improved Java SQL error information
- Java API for Set Client Information (SQLESETI)
- Native DB2 server SQL error messages
- Connection pooling improvements

Scrollable Cursor Support

- Exploits DB2 engine scrolling
- Supports update operations
- Java driver performs "local" scrolling within a block of data
- minimizes network traffic



Improved Security for DB2 Authentication

- Supports three main authentication techniques:
 - ▶ USERID/PASSWORD in clear text
 - ▶ Encrypted USERID/PASSWORD
 - ▶ Kerberos
- Uses these Java services for encryption and Kerberos:
 - ▶ IBM Java Generic Security Service (JGSS)
 - ▶ IBM Java Authentication and Authorization Service (JAAS)
 - ▶ IBM Java Cryptography Extension (JCE)
- Authentication technique can be specified by:
 - ▶ setting a Java property in the application
 - ▶ recording the desired technique in the DB2 dataSource definition



Improved SQL Error Information

- DB2Diagnosable class for reporting contents of the SQLCA and SQL error message text
 - ▶ getSQLCode()
 - ▶ getSQLErrmc()
 - ▶ getSQLErrp()
 - ▶ getSQLErrd()
 - ▶ getSQLState()
 - ▶ getSQLWarn()
 - ▶ getSQLErrorMessage()
- Information is accessible for both JDBC and SQLJ whenever an SQL exception is thrown

Java API for Set Client Information

- New methods added to provide Java APIs to the existing Set Client Information API (i.e. SQLESETI)
 - ▶ `setClientUser(String)`
 - ▶ `setClientWorkstation(String)`
 - ▶ `setClientApplicationInformation(String)`
 - ▶ `setClientAccountingInformation(String)`
- For DB2 for Unix and Windows users:
 - ▶ provides additional monitoring information
- For DB2 for OS/390 and z/OS users:
 - ▶ provides additional monitoring information
 - ▶ all four strings are included in all IFC records
 - Allows you to search accounting records or other IFC records for data related to a particular Java user or application.
 - ▶ the last two strings can be used for WLM prioritization of the Java connections to DDF

Native DB2 Server SQL Error Messages

- "Error Message" stored procedures are provided by each DB2 server (including DB2 for OS/390 V6 and V7)
- Allows DB2 client to return "native" error message text for the target DB2 server
- Native error message is only returned when explicitly requested
 - ▶ `getSQLErrorMessage()`

Java Trace

[ibm][db2][jcc][Thread:main][Connection@2152e6] setAutoCommit (false) called

[ibm][db2][jcc][Thread:main][Connection@2152e6] prepareStatement (INSERT INTO testTable values(?,2,3,4,5,6000,'char','varchar',X'FF',X'FFFF', '2001-11-15','08.30.00','2001-11-15-08.30.00.000000'); called

[ibm][db2][jcc][Thread:main][Connection@2152e6] prepareStatement () returned PreparedStatement@58b649

[ibm][db2][jcc][Thread:main][PreparedStatement@58b649] setInt (1, 1) called

[ibm][db2][jcc][Thread:main][PreparedStatement@58b649] executeUpdate () called

[ibm][db2][jcc][Thread:main][PreparedStatement@58b649] executeUpdate () returned 2

[ibm][db2][jcc][Thread:main][PreparedStatement@58b649] close () called

[ibm][db2][jcc][Thread:main][Connection@2152e6] getWarnings () returned null



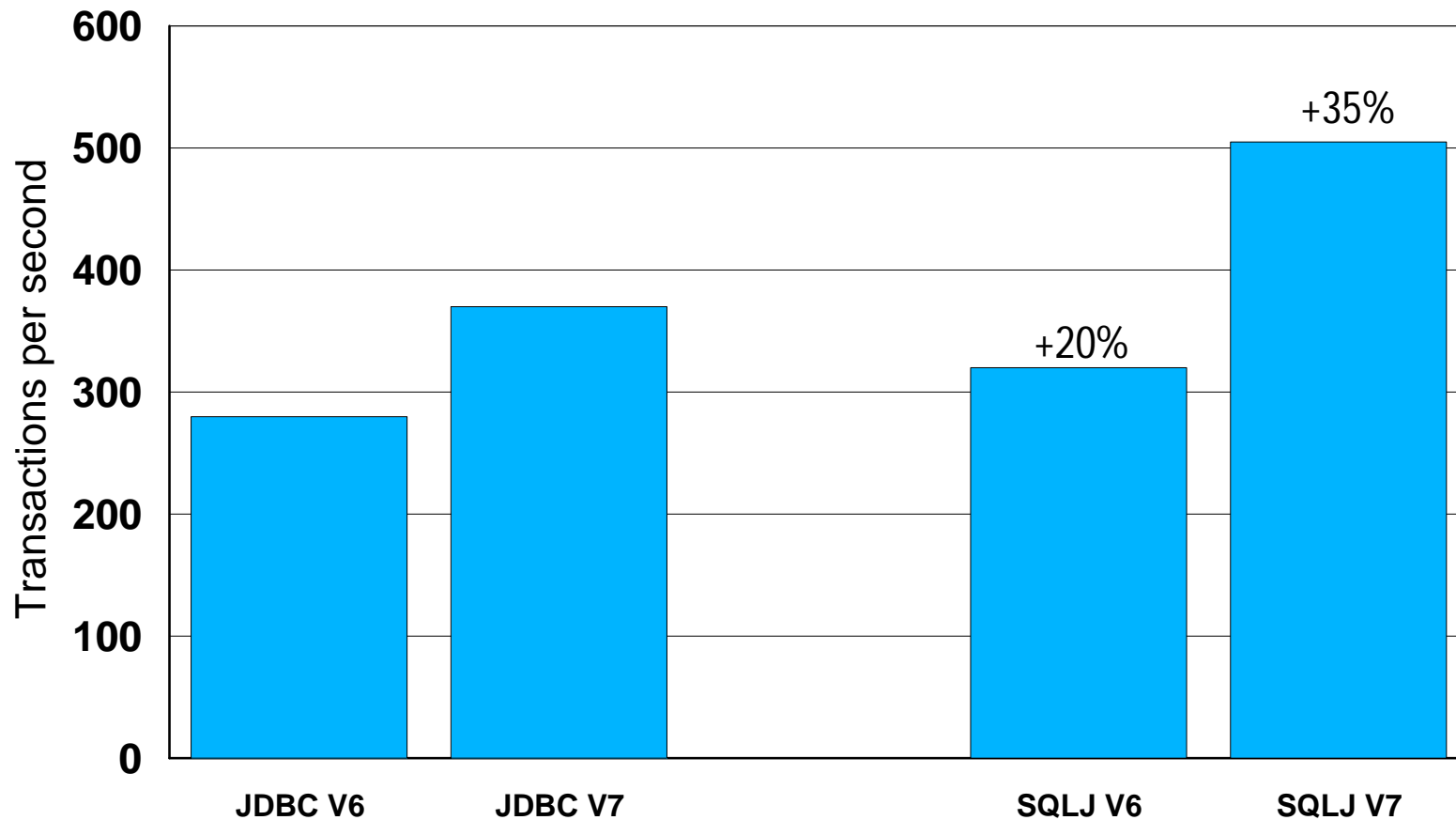
DataSource connection pooling

- JDBC driver can now perform its own connection pooling
- Provides tighter integration with DB2
 - ▶ "signon" support to reuse DB2 connection threads, including potential to change userid
 - ▶ special registers reset to initial values on each connection
 - ▶ temporary tables and prepared statements discarded on `con.Close()`
 - ▶ inflight units of work forced to rollback on `con.Close()`
 - ▶ better detection/correction of damaged connections on `con.getConnection()`

Why use SQLJ?

- Static SQL performance for Java applications
 - ▶ significant performance advantage over JDBC
- Static SQL authorization model
 - ▶ provides Java with a stronger authorization model
- Productivity
 - ▶ less code written by the application programmer
 - ▶ resulting code is easier to maintain

Java API Performance Comparisons



Normalized throughput for zSeries G7 with 3 engines with 100% cache hit for JDBC. SQLJ advantage increased from 20% to 35% when Java overhead was reduced.

SQLJ Application Development

- 100% Java application process
 - ▶ eliminates DBRM files and .bnd files
- New SQLJ serialized profile format
 - ▶ fully portable to all platforms -- user can deploy on any server platform without running db2profc on the target system.
 - ▶ contains information needed for all BIND operations, without having to recustomize on each BIND
 - ▶ old SQLJ serialized profiles are automatically converted to the new format
- Simplifies deployment of applications, but does require changes in existing procedures used by SQLJ users.

VisualAge Tooling (old news)

- Some support for SQLJ
 - ▶ manual invocation of the SQLJ translator
 - ▶ manipulation of SQLJ was cumbersome
 - ▶ no support for generating SQLJ statements
 - ▶ no SQLJ support for Container-Managed Persistence in WebSphere (CMP beans)

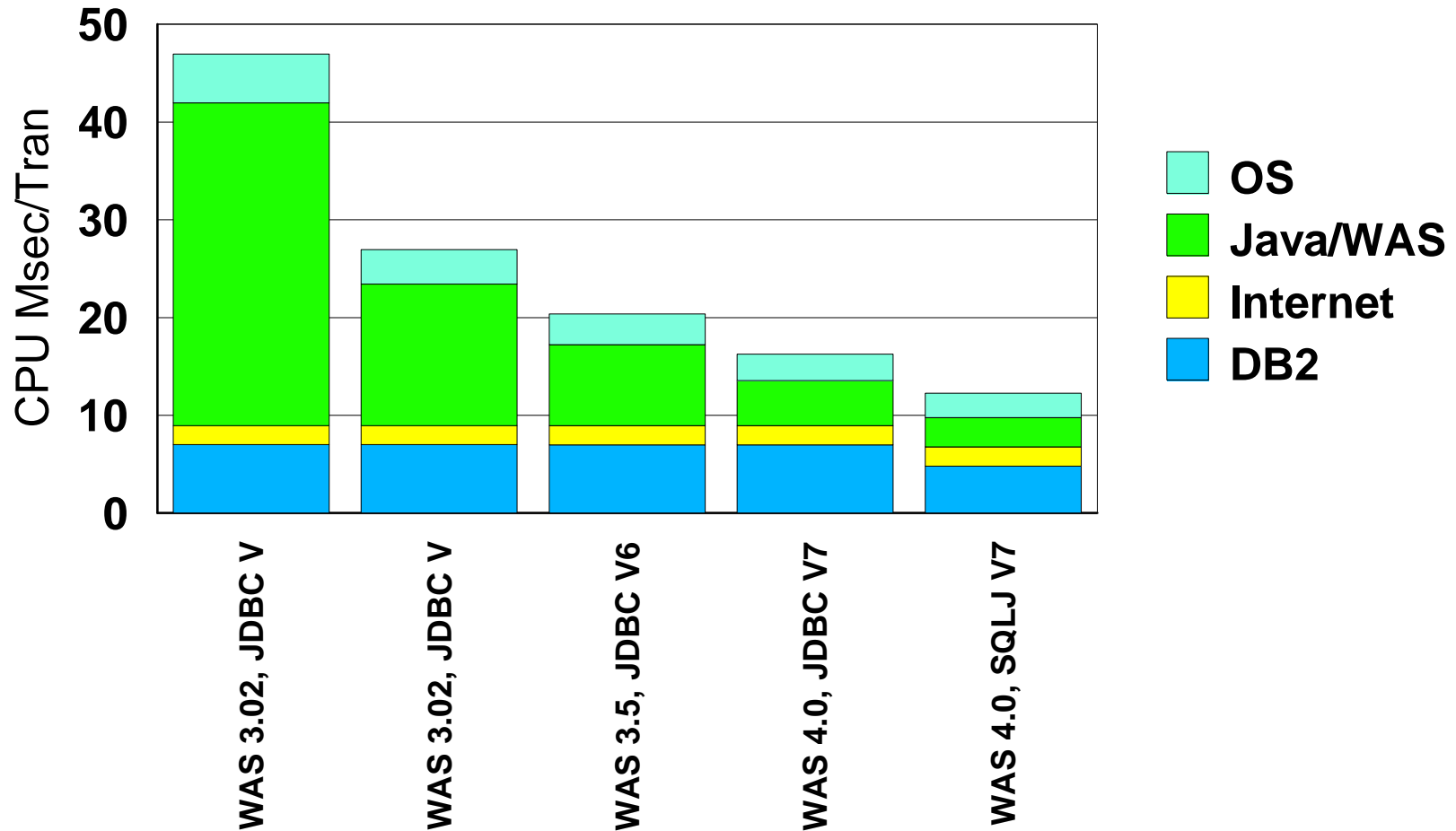
WSAD 5.0 Tooling (new!!!)

- Support for generating SQLJ for CMP beans
 - ▶ includes static singleton select for improved performance
- Automatic SQLJ translation
- SQLJ programs are fully supported by the WSAD workbench
 - ▶ .sqlj and .ser files are first class objects now
- Support for access intent has been added
 - ▶ better control over isolation level
 - ▶ automatically generates KEEP UPDATE LOCKS for JDBC access to DB2 for OS/390 when required



CPU Cost Comparisons

eRWW Workload



OS - BCP, RTL and OPENMVS

Internet - WebServer DLLs, WAS DLLs, TCPIP and VTAM

WAS 3.02, JDBC type 1 numbers are estimates