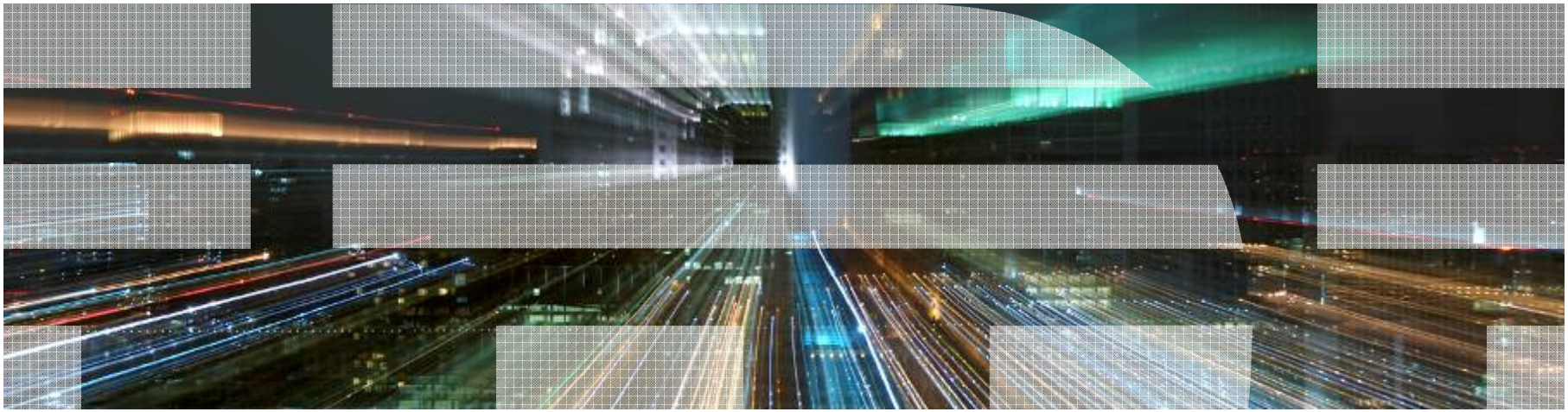
Kyle Charlet
STSM, IMS SOA and Modernization
charletk@us.ibm.com

IMS Application and Database Modernization



Agenda

IMS modernization overview

- IMS database access
- DataPower and IMS database

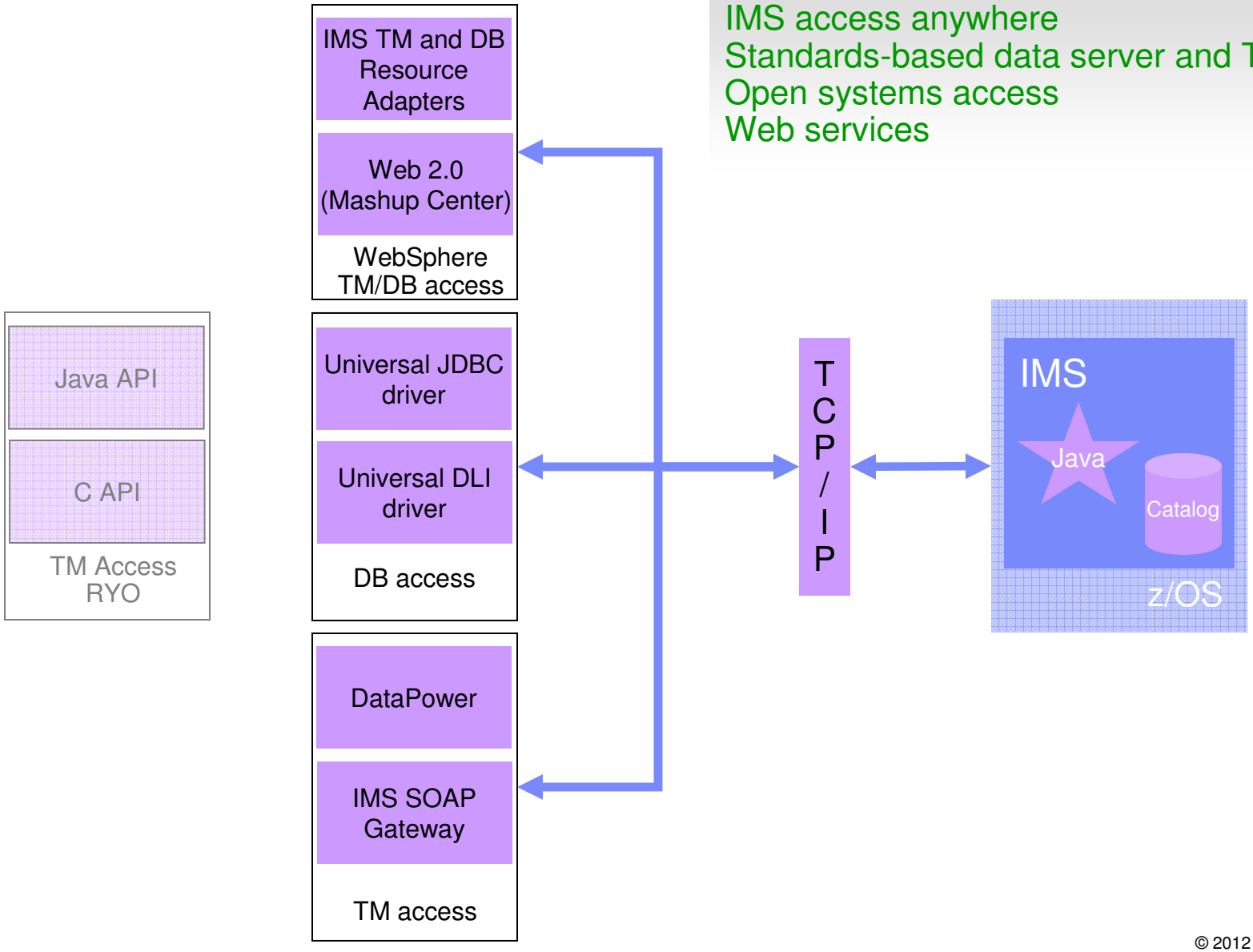
Intended IMS DB portfolio integration

IMS and Java on System z

- IBM Java on System z strategy

IMS Modernization Solutions

IMS access anywhere
Standards-based data server and TM
Open systems access
Web services



IMS Open Database

Solution statement

- Extend the reach of IMS data
  - Offer scalable, distributed, and high-speed local access to IMS database resources

Value

- Business growth
  - Allow more flexibility in accessing IMS data to meet growth challenges
- Market positioning
  - Allow IMS databases to be processed as a standards-based data server

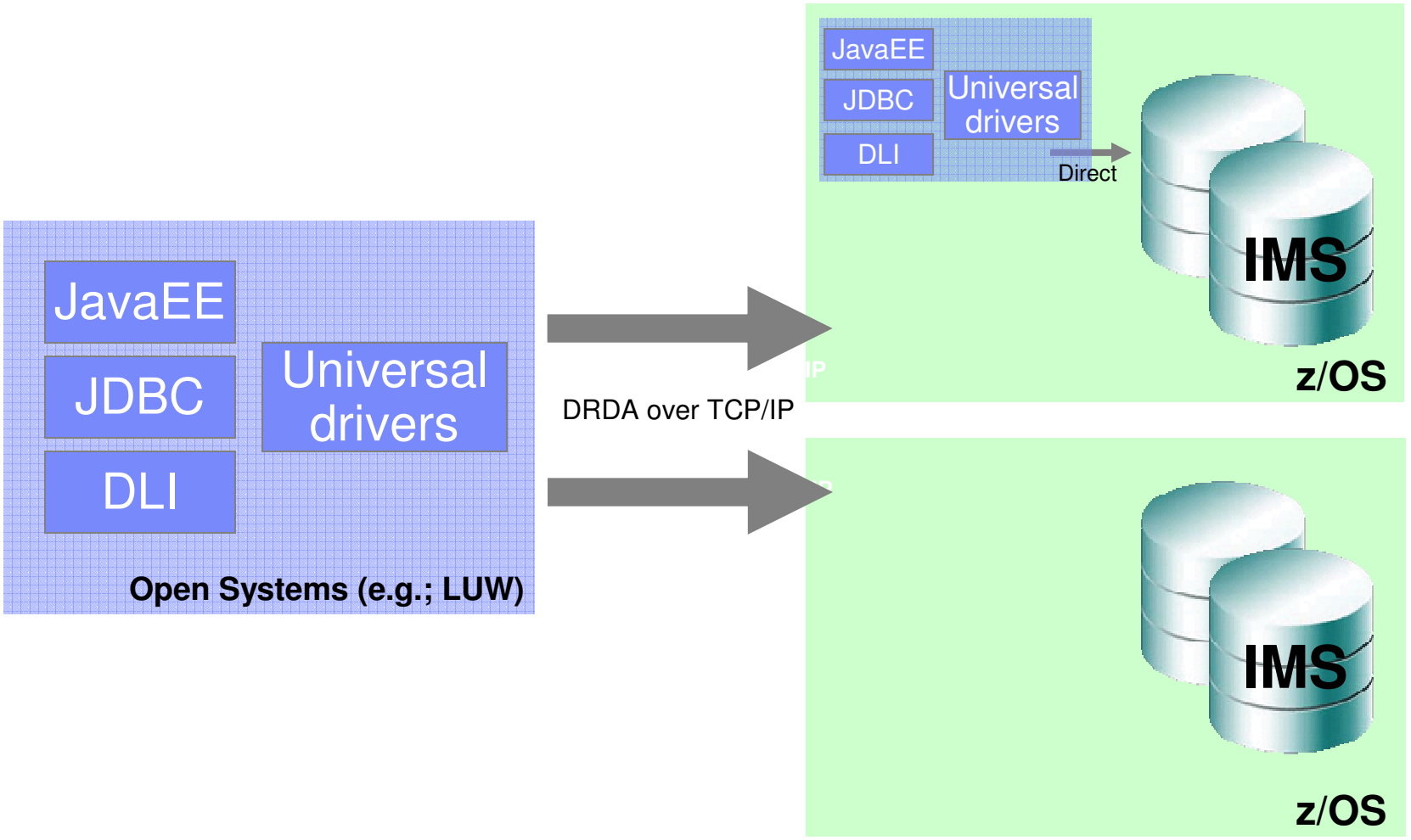
Key differentiators

- Standards-based approach (Java Connector Architecture, JDBC, SQL, DRDA)
- Solution packaged with IMS

Enables new application design frameworks and patterns

- JCA 1.5 (Java EE)
- JDBC

IMS Open Database



Solution highlights – JEE deployment

Universal DB resource adapter

- JCA 1.5
  - XA transaction support
    - Manage multiple datasource connections in a single UOW
  - Local transaction support
    - Manage multiple datasource connections each in their own UOW
  - Connection pooling
    - Pool released connections for future use
  - Connection sharing
  - Multiple programming models available
    - JDBC (Universal JDBC driver incorporated)
    - CCI with SQL interactions
    - CCI with DLI interactions

Solution highlights – JDBC

Universal JDBC driver

- Significant enhancements to classic JDBC offered in IMS 9 and IMS 10
  - Standardized SQL support
  - XA transaction support (type 4)
  - Local transaction support (type 4)
  - Concurrency control
    - Control release of distributed locks
  - Updatable result set support
  - Batching support
    - Fetch multiple rows in a single network call
  - JDBC metadata discovery support

Standard SQL and metadata discovery enables significant integration opportunities for IMS

Solution highlights – DLI

Universal DLI driver

- Java implementation of DL/I API
- Complete DL/I support for database access
- All IMS command codes supported
- Can mix usage of JDBC and DLI drivers in the same application
  - SQL cannot always express what DLI offers

Open Database and the Universal drivers

Deep synergy with the IMS catalog

- Direct access to IMS metadata in the catalog
- Virtual and cloud deployment capabilities
  - No longer file-system dependent for metadata
- Industry-leading data type support
  - Complex and flexible
- Mapping support

Deep synergy with Java z/OS and z196

- Significant performance improvements
- Continued partnership with Java z/OS lab

Continued SQL standardization and support

- Including similar connection parameters as DB2 for commonality across IBM drivers
- More to come

Continued integration across the IBM portfolio

Data types

Data types have multiple metadata elements

- Application data type
  - Universal drivers use application data type to present data to clients
- Physical data type
  - Universal drivers use physical data type to marshal/unmarshal data to and from the database
- Example
  - Application data type is DECIMAL(10,2) [decimal with precision 10 and scale 2]
  - Physical data type is a signed packed decimal (AD community doesn't need to know this)

New data type support

- Structs (nested n levels with no constraint on element data types)
  - Accessed via SQL and DLI
- Arrays (nested n levels with no constraint on element data types)
  - Accessed via SQL and DLI
- User-defined
  - Name of UDT can be defined to the catalog and intended to be used at runtime by Universal drivers to marshal/unmarshal data
  - Can be part of a Struct or Array element

Maps

Mapping support

- A Map is metadata that describes how a field (or set of fields) are mapped for a particular segment instance
- Metadata captures the various cases and for each case defines the set of fields to be used for that case
- Maps can be defined to the catalog
- Example
  - Insurance segment mapped multiple ways depending on value of a ‘Policy Type’ field

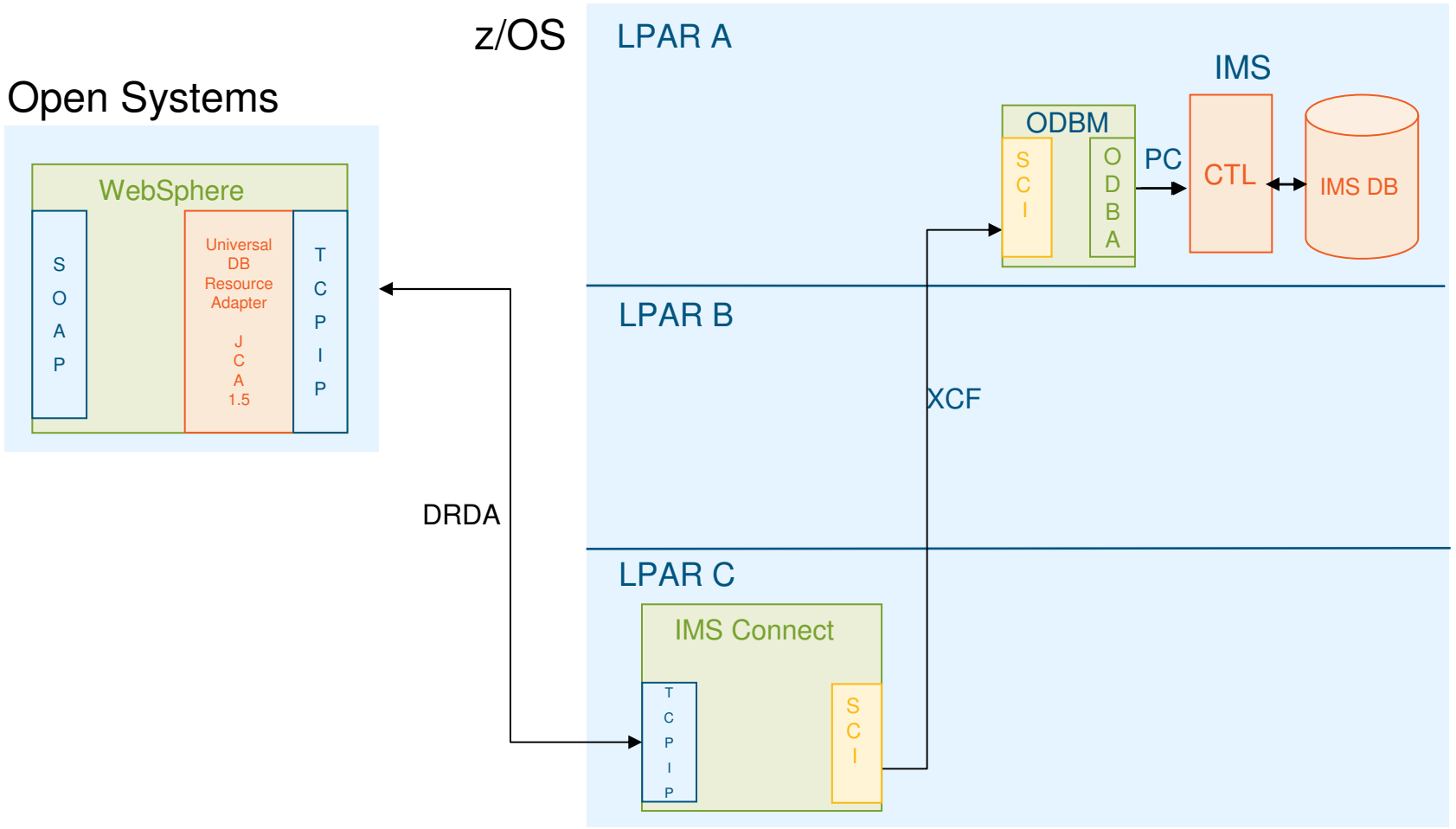
Policy Type	Property Type	Rooms	Value	Address	Make	Model	Year	Value	Color
M	-	-	-	-	Ford	Escort	1989	2K	Red
H	Single Family	5	500K	555 Disk Drive Way, 95141	-	-	-	-	-

Additional enhancements

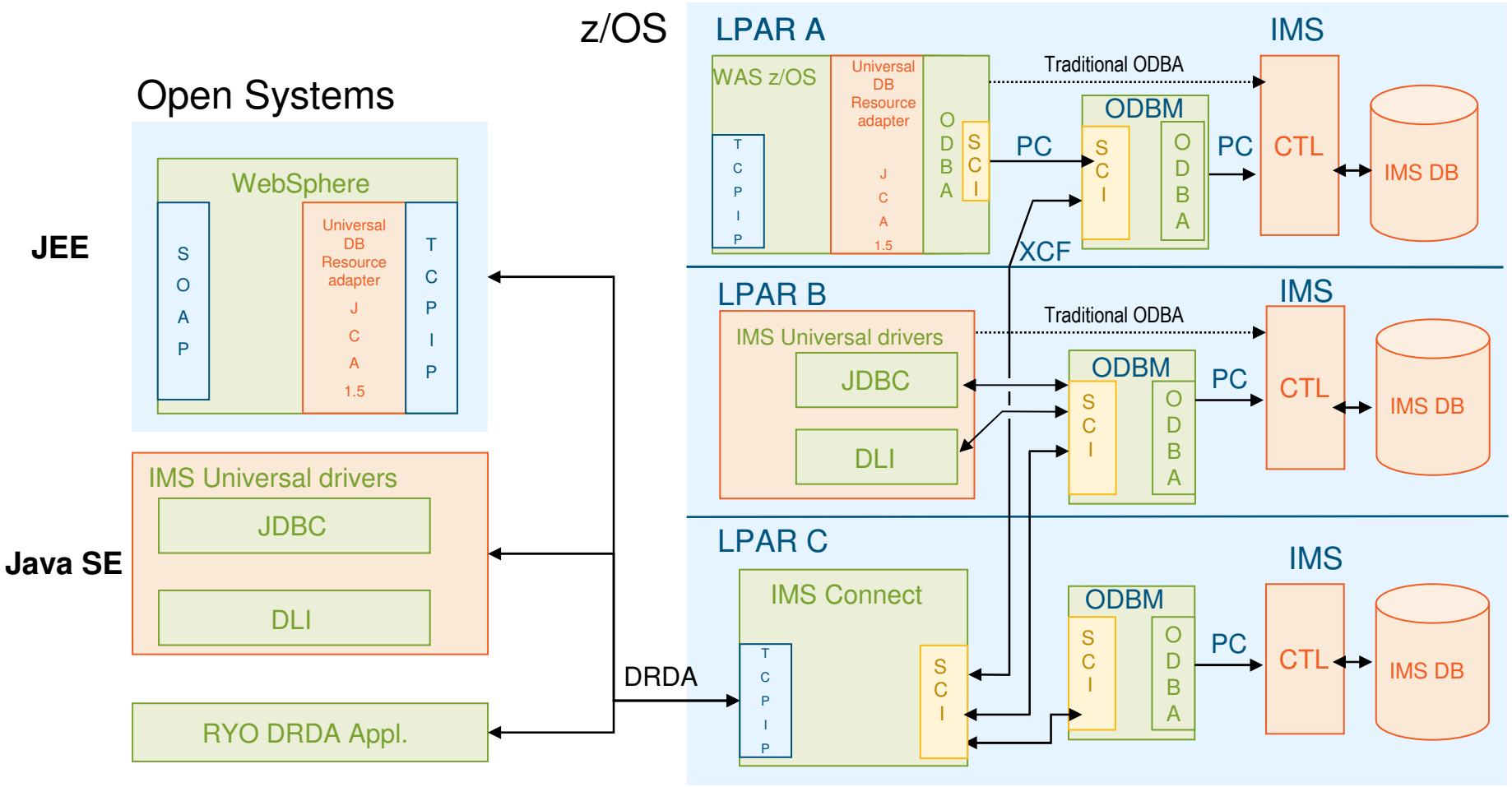
- SQL
  - FETCH FIRST <n> ROWS ONLY
  - INNER JOIN <table2> ON <table1.col1> = <table2.col2>
- Connection properties
  - currentSchema
  - maxRows
  - fetchSize
  - Tracing
    - traceFile, traceFileAppend, traceDirectory, traceLevel
- Variable length segment support
  - VL segments contain a two byte length (LL) field that will identify the size of the segment instance
  - Universal Drivers are now sensitive to the LL field of a VL segment and will manage the IO area of the segment instance on all CRUD calls

FIELD=PERSONAL_INFO (VLOB min length=82 max length=112)			
INNER FIELD=LENGTH (2 bytes)	INNER FIELD=NAME (30 bytes)	INNER FIELD=ADDRESS (50 bytes)	INNER FIELD=EMAIL (optional field 30 bytes)
112	RICHARD	555 Bailey Ave	tran@abc123.com
82	KEVIN	555 Bailey Ave	<does not exist physically on disk>

IMS Open Database environment



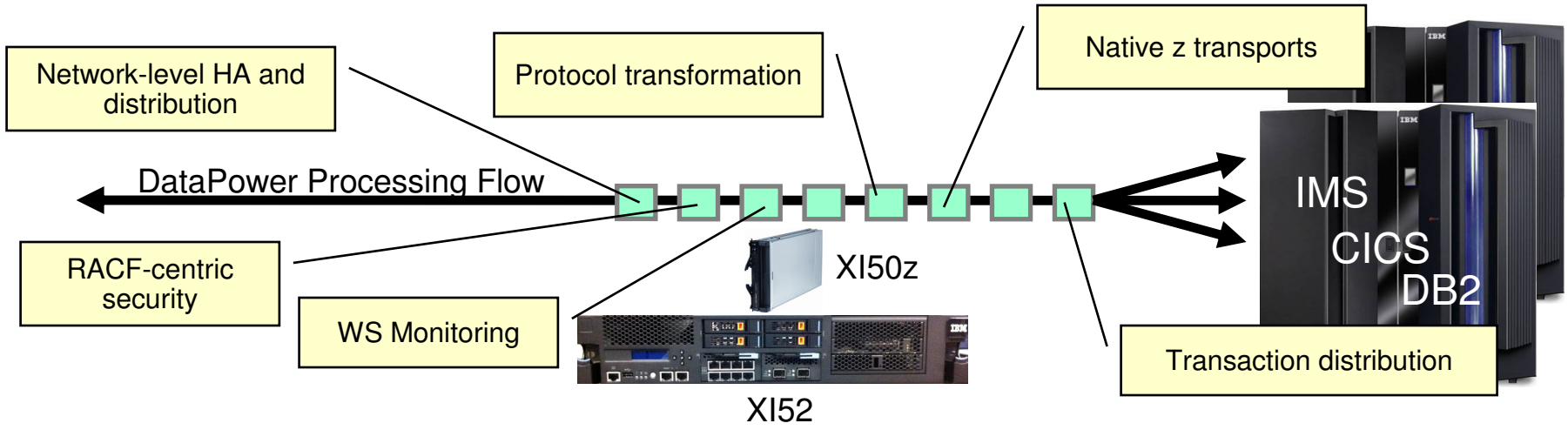
IMS Open Database environment



Premier System z web service enablement through DataPower SOA appliances

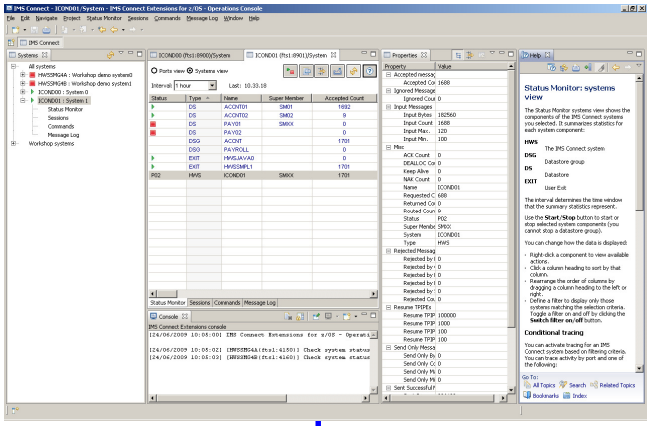
IBM cross-brand initiative

- Deep synergy between DataPower, System z, Rational and Common Transformation tooling to support DataPower as the premier System z gateway for IMS, CICS and DB2
- Intended support for IMS DB access
- Intended support for top-down service approach for inbound and outbound IMS transactional requests



IMS Modernization

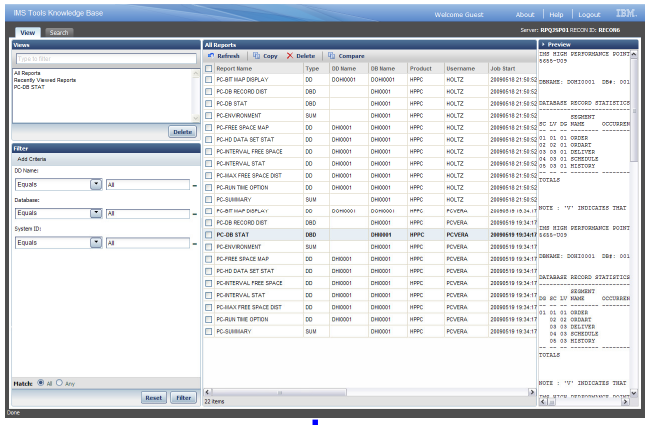
IMS Explorer for Development (Eclipse)



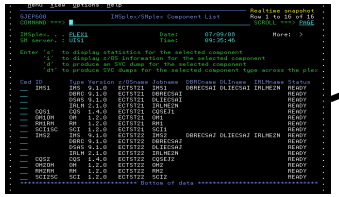
Developers

Administrators

IMS Explorer for Administration (Web Browser)

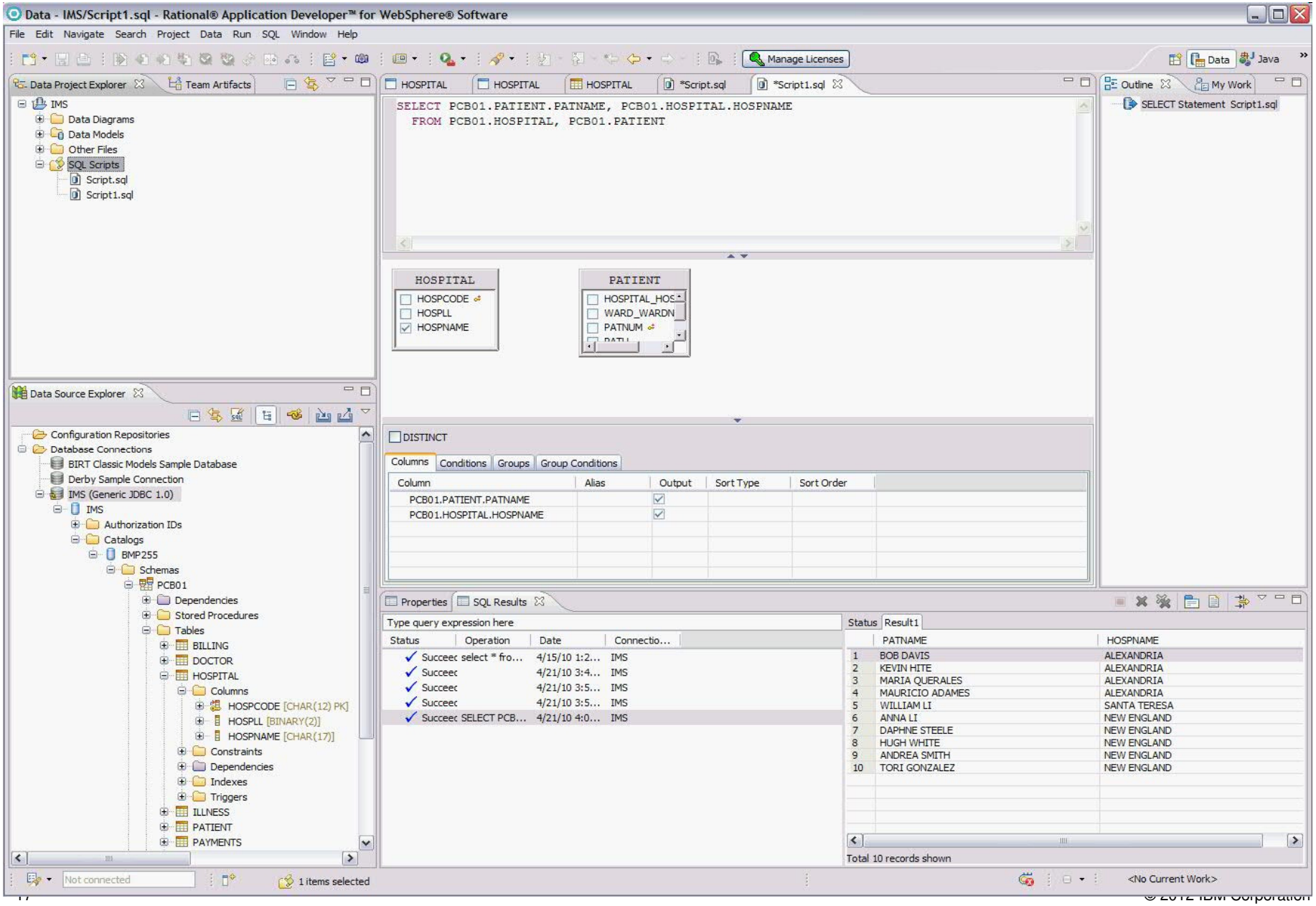


ISPF

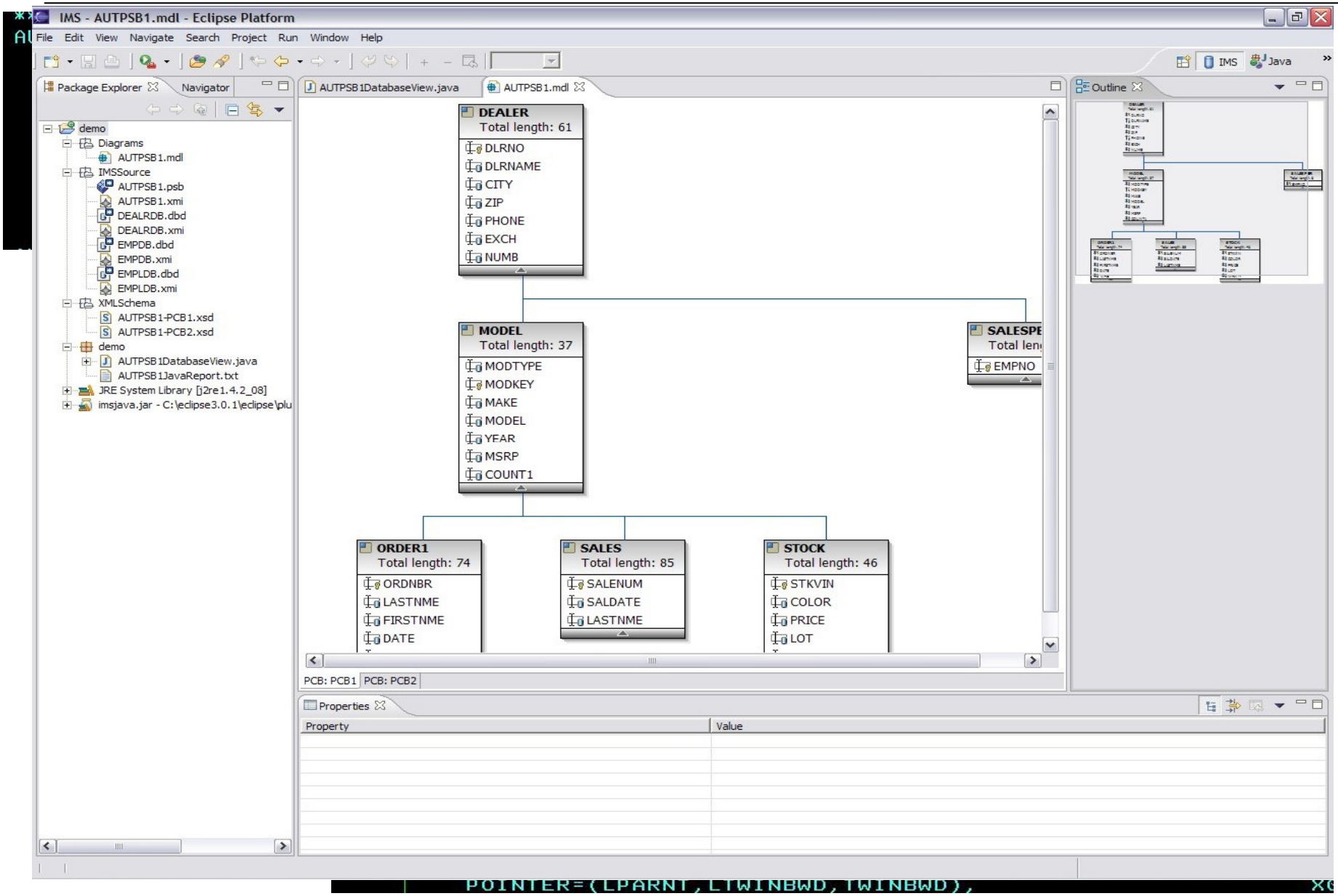


IMS

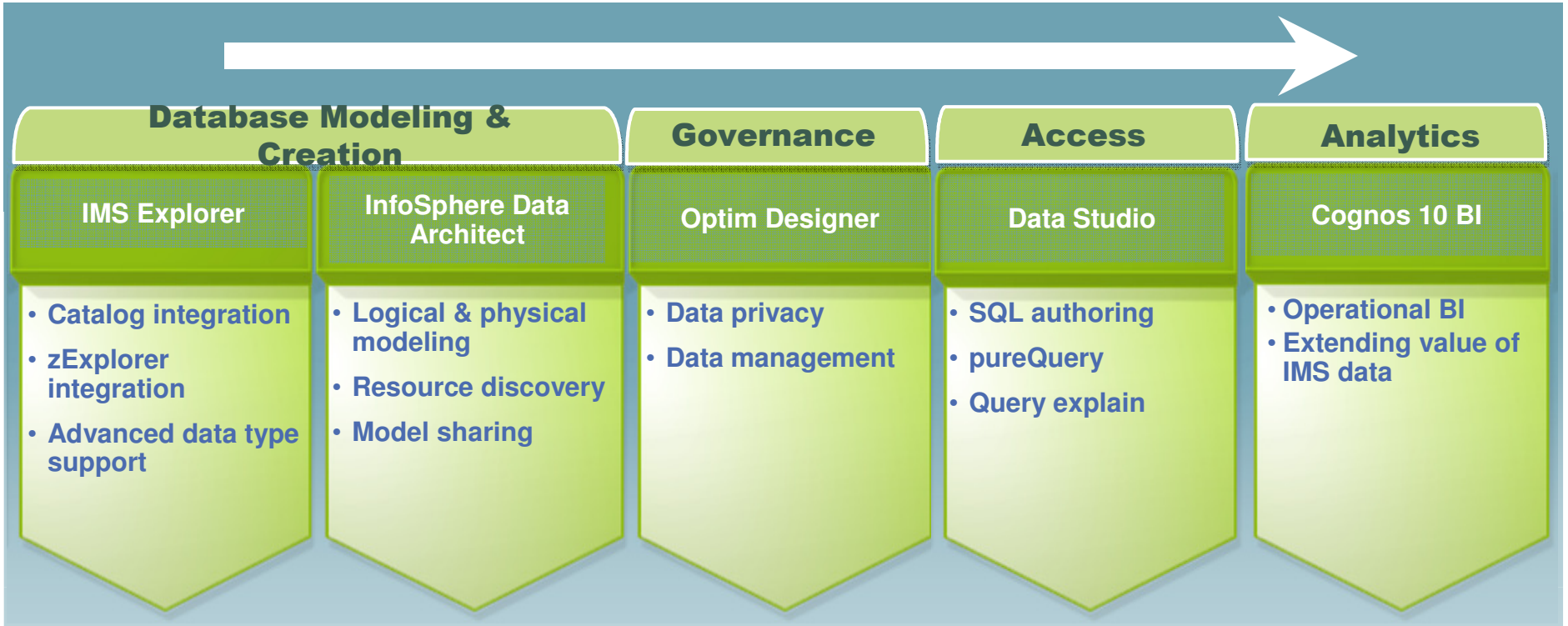
IMS Explorer for Development



IMS Explorer For Development



IBM Portfolio Integration 2012-2013



- Physical modeling & resource discovery
- Database resource creation

Java z/OS

z196 and Java6.0.1: Engineered Together

- Up to 2.1x improvement to Java throughput
- Reduced footprint
- Tighter integration with z/OS facilities
- Improved responsiveness in application behavior

J9 R2.6 Virtual Machine

- Significant enhancements to JIT optimization technology
- z196 exploitation of instructions and new pipeline
- New Balanced GC policy to reduce max pause times
- Default GC policy changed to gencon

z/OS Unique Enhancements

- JZOS 2.4.0
- z/OS Java unique security enhancements

Performance

- 2.1x improvement to multi-threaded workload
- 1.93x improvement to CPU-intensive workload

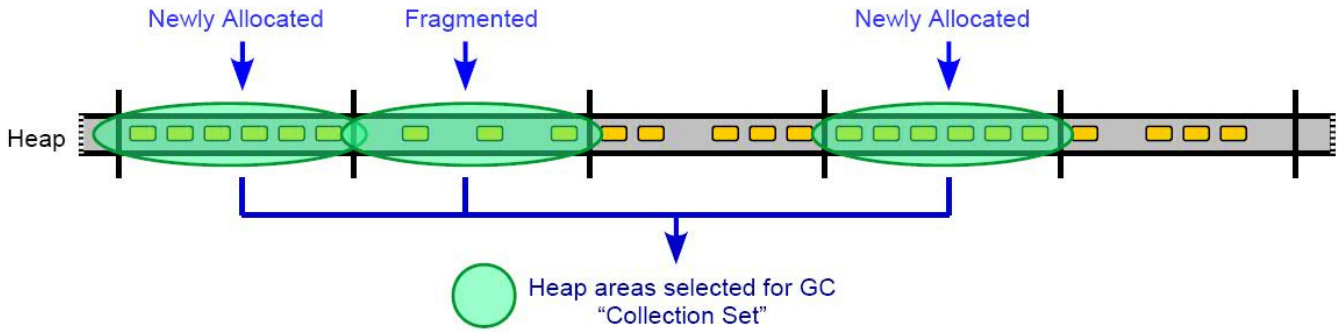
IBM J9 2.6 Technology Enhancements:
Garbage Collection: Balanced Policy

Improved responsiveness in application behavior
- Reduced maximum pause times to achieve more consistent behavior
- Incremental result-based heap collection targets best ROI areas of the heap
- Native memory aware approach reduces non-object heap consumption

Next generation technology expands platform exploitation possibilities
- Virtualization – Group heap data by frequency of access, direct OS paging decisions
- Dynamic reorganization of data structures to improve memory hierarchy utilization (performance)

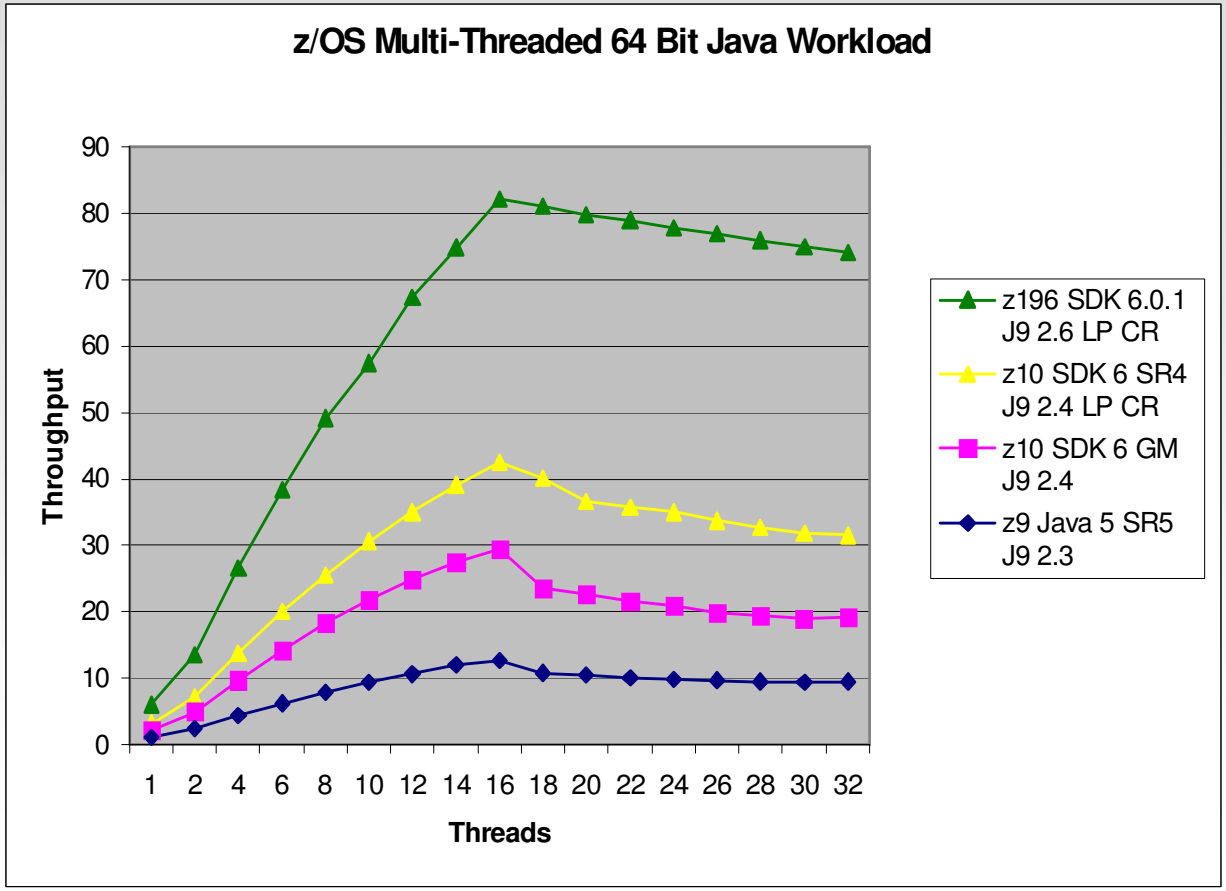
Recommended deployment scenarios
- Large (>4GB) heaps
- Frequent global garbage collections
- Excessive time spent in global compaction
- Relatively frequent allocation of large (>1MB) arrays

Input welcome: Help set directions by telling us your needs



z/OS Java SDK 6.0.1 performance

Aggregate HW and SDK improvement z10, z196, Java6 to Java6.0.1

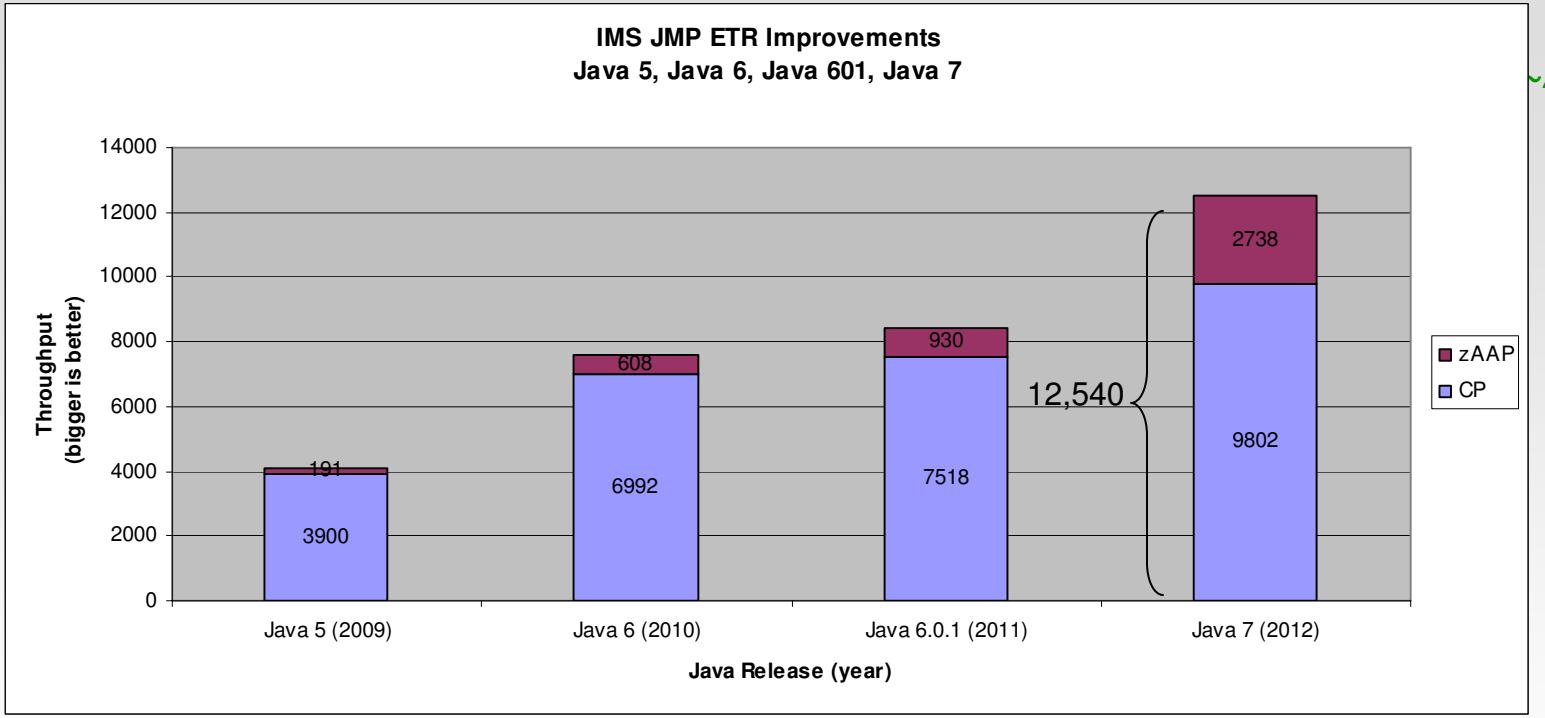


~7x aggregate improvement from z10, z196, Java6 and Java6.0.1

(Controlled measurement environment, results may vary)

IMS JMP region performance

Aggregate SDK and SW improvement



2.5x aggregate throughput improvement from Java5 to Java 7

2 GCP + 2 zAAP

(Controlled measurement environment, results may vary)

z196™ – z/OS V1.12

Java and IMS

Java is an integral component of the IMS modernization strategy

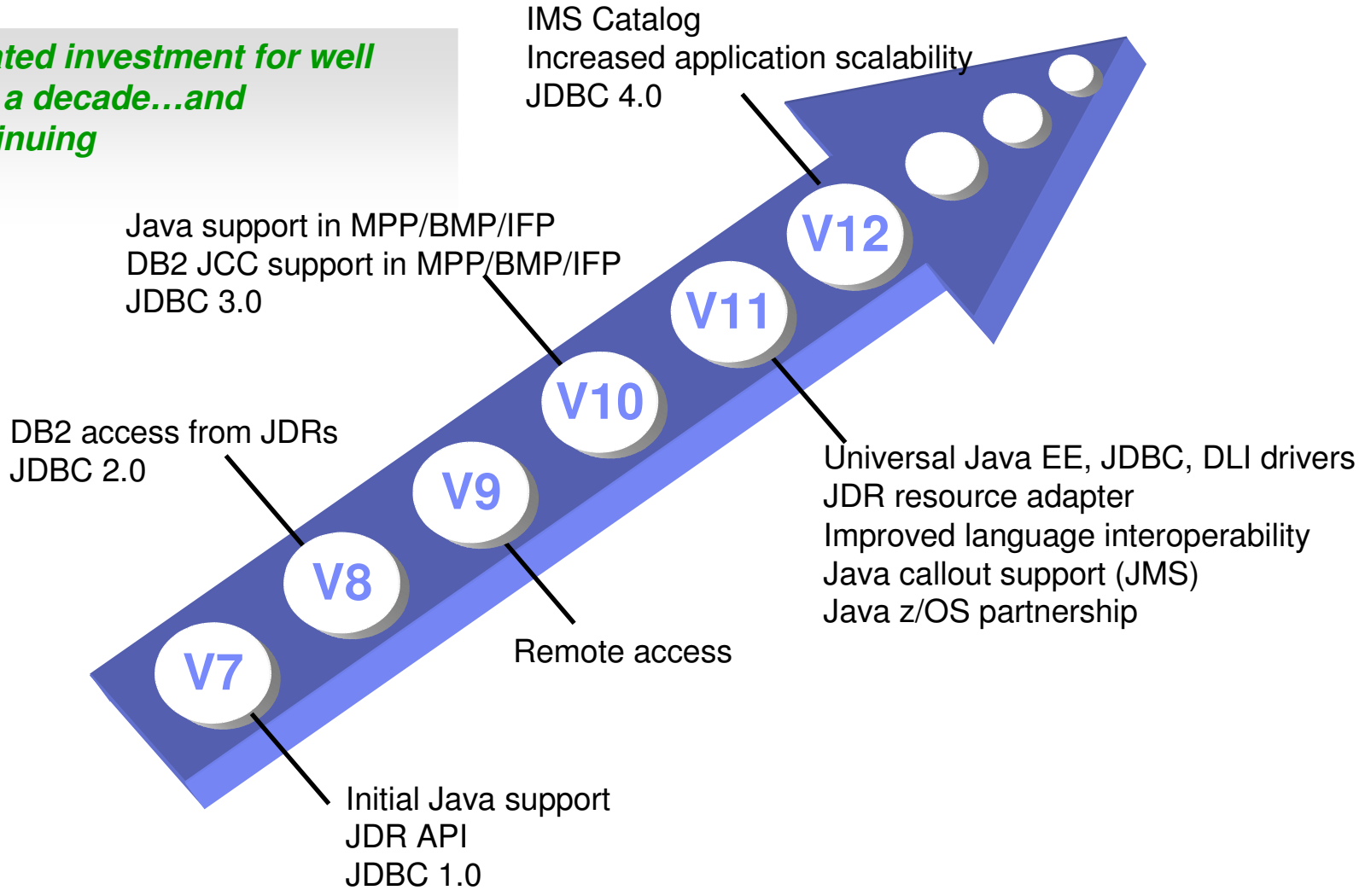
- Enable customers to quickly achieve IMS value while significantly reducing development costs and improving productivity
- IMS leverages the IBM JVM for System z and integrates it into the IMS runtime containers

IMS family has a long-term commitment to Java

- Investing over 50 FTEs (full-time equivalents) in Java technology moving forward
  - IMS dependent region types (JMP, JBP, MPP, BMP, IFP)
  - Java EE platform (WebSphere Application Server)
  - z/OS and open systems access to IMS assets

Java and IMS – IMS 7 to IMS 12 (highlights)

Dedicated investment for well over a decade…and continuing



Java and IMS moving forward

Java z/OS stakeholder

- Continued partnership to maximize synergy between IMS and Java z/OS

Performance

- Aggressive performance analysis and cooperative approach to continue h/w and s/w exploitation

Enterprise modernization

- Language interoperability
- Universal drivers/JDR resource adapter

Java and IMS moving forward

Additional runtimes

- Intended support of WebLogic
- Intended support of .NET
  - Data provider

Integration

- Aggressive approach to horizontal integration across IBM portfolio
  - Rational
  - Cognos
  - Data Studio
  - InfoSphere

Java and IMS moving forward

Continued modernization of the core system

- Get by offset support
- Database versioning
- Dynamic database
- Native SQL
- Programming models

Summary

IMS is committed to enterprise modernization

- Deep synergy across many organizations within IBM
- Portfolio integration is very important
- Constantly validating the enterprise roadmap with customers

The partnership of IMS and Java technology is capable of handling mission-critical workload

- IMS is an important stakeholder in the IBM Java on System z strategy
- Java running in IMS regions has been benchmarked at over 9400 transactions per second

Many customers are modernizing their IMS application development patterns and access paradigms around Java as the primary language of choice

- Over 40 proof of concepts in the last year alone