

Featured integration solutions for IMS Version 13

- **IBM IMS Enterprise Suite**
 - offers open integration technologies for access to IMS transactions and data
- **IBM Cognos Business Intelligence (BI) 10.2**
 - offers a data connection type for direct connections to IMS databases.
- **IBM InfoSphere Data Explorer 9.0**
 - directly access to IMS operational data to explore critical business data.
- **WebSphere DataPower® Integration Appliance V6.0 (5725-K52)**
 - IMS Synchronous Callout Support
 - Access to IMS transactions that are running in IMS TM.
 - Access to databases in IMS DB

IMS Enterprise Suite

Integration solutions and tooling that support open integration technologies, enable new application development and mobile services, and extend access to IMS transactions and data.

Concurrent Application Threads Enhancement

- **Partition Specification Table (PST) used for**
 - Active dependent regions (MSG/BMP/IFP/JMP/JBP)
 - CICS/DBCTL threads
 - Open Database Access threads
- **Customers continue to require more PSTs!**
 - 31 dependent regions - 1980 (IMS 1.1.6)
 - 999 dependent regions - 1995 (IMS 5.1)
 - 4095 dependent regions - 2013 (IMS 13)
- **Related parameters**
 - MAXPST=
 - PST=
 - MAXTHRDS= for ODBM
 - MAXTHRDS=, MINTHRDS= for DBCTL
- **Details in IMS13 System Sections**

This slide is a reminder as IMS integration solutions grow the need for more application threads (PSTs) will increase.. Details are covered in IMS 13 systems section

ODBM MINTHRD Default 1 for RRS=N

▪ **Problem**

- EXCESSIVE TCB ATTACH AND DETACH PROCESSING IN AN ODBM RRS=N ENVIRONMENT.

▪ **Solution**

- MINTHRD value in an RRS=N ODBM environment is set to approximately 62% of the value of MAXTHRD
- IMS 12 PM63976 UK82608
- IMS 11 PM63977 UK82609

▪ **Benefit**

- performance issues associated with attaching and detaching threads are reduced.

Improved performance by changing the MINTHRD default from 1 to 62% of the MAXTHRD value to minimize the attach/detach processing.

IMS TM Resource Adapter Version 13

- Based on version 1.5 of the JEE Connector Architecture
- Documented in IMS Version 13 Application Programming (SC19-3646-00)
- Provides all key features available in Version 12:
 - Support for RACF password phrases from 9 -100 characters
 - APAR PM91898 (TM RA V13.2.0):
Note: IMS Connect V13 PM91312 and IMS V13 PM85849 are required
 - Support for changing RACF passwords from Java applications
 - Support for RACROUTE VERIFY return codes
 - Support for IBM WebSphere Application Server Version 8 and its resource workload routing function
 - Support for both synchronous and asynchronous callout requests from IMS applications
 - Support for integrating IMS MFS services in business processes through IBM Integration Designer and IBM Process Server
 - Support for a single message-driven bean (MDB) to pull callout messages from more than one IMS data store

03-IMS13 Integration: 5

If the password phrase is used, an internal PING message will be first sent to IMS Connect (ICON) to make sure that the level of IMS Connect does support the password phrase. Otherwise, the regular 8 bytes password value will be used for verification.

Note: IMS Connect V13 PM91312 and IMS V13 PM85849 are required for this function to work properly. Both APARs can be applied independently to TMRA and IMS Connect, however, both APARs must be applied to use the password phrase feature

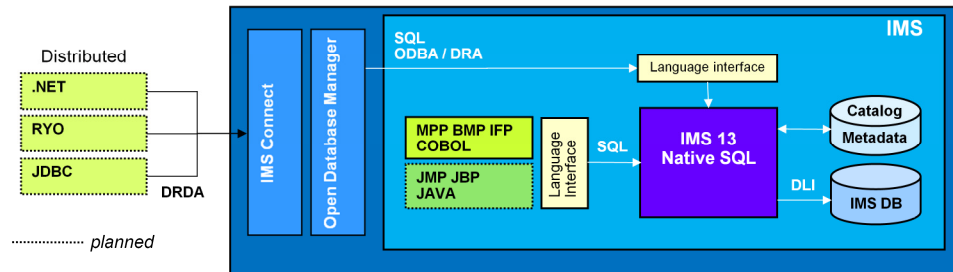
Support for multiple data stores per IMS activation specification for callout messages enables a single message-driven bean (MDB) to pull callout messages from more than one IMS data store. A shared-queues environment removes need to duplicate the MDB application to connect to each IMS member in the shared queue.

Integration Enhancements

- **IMS SQL Support**
- **IMS Enterprise Suite 3.1**
 - IBM IMS Data Provider for Microsoft .NET
 - SOAP Gateway Updates
 - Connect API for Java V2.2 Updates
 - Explorer for Development Updates
 - IMS Mobile Feature Pack
 - Java Message Service (JMS) API
 - IMS 13 synchronous program switch support
- **IBM IMS Explorer for Administration**
 - Extension to the IBM Tools Base v1.4 Administration Console component

IMS SQL Support

- Native SQL for COBOL and distributed applications (.NET/JDBC)
 - COBOL Details in [IMS13 Application Programming Enhancements Section](#)
- Provides standard SQL keywords to easily access IMS data
 - ✓ SELECT, INSERT, UPDATE, DELETE
 - ✓ Uses Dynamic SQL programming model
 - ✓ Converts SQL statements to DLI calls
 - ✓ Supports a subset of SQL keywords that are currently supported by IMS Universal JDBC driver
- Uses database metadata in IMS Catalog
 - ✓ No need to generate metadata for use in applications



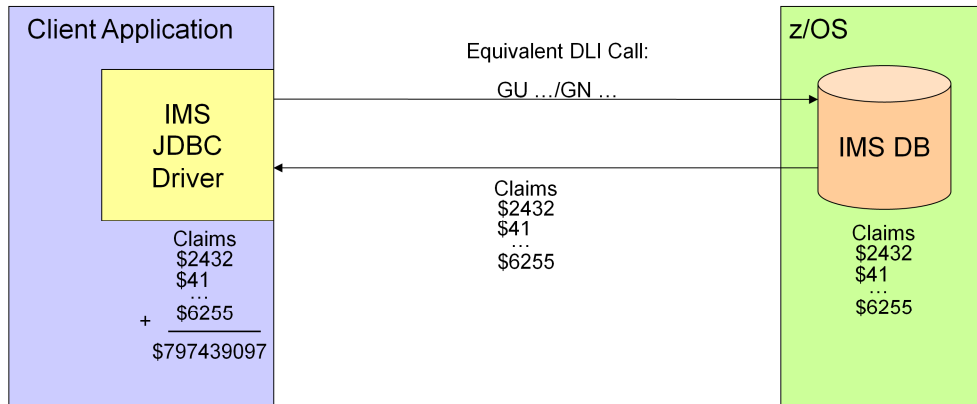
03-IMS13 Integration: 7

IMS DRDA DDM command support for native SQL enhancement

- The DDM command support for native SQL requires the Open Database Manager (ODBM) component of the IMS Common Service Layer (CSL).
 - ODBM translates the DDM commands into SQL and then routes the SQL calls to the appropriate IMS system.
 - The receiving IMS system's native SQL translates the SQL into DL/I.
- **IMS Data Provider for Microsoft .NET will use this support**
- **IMS Universal Drivers to be updated via service process**
 - Enables SQL processing to be handled directly by IMS instead of on the client side,
 - Results in increased performance for the IMS Open Database solution.

SQL Parsing in IMS Java Code

- **Requirement** - Improve performance for SQL data aggregation
- Example
 - How much money has my insurance company paid out in claims for the year 2011?
 - `SELECT SUM(CLAIMAMOUNT) FROM CLAIMS WHERE YEAR=2011`



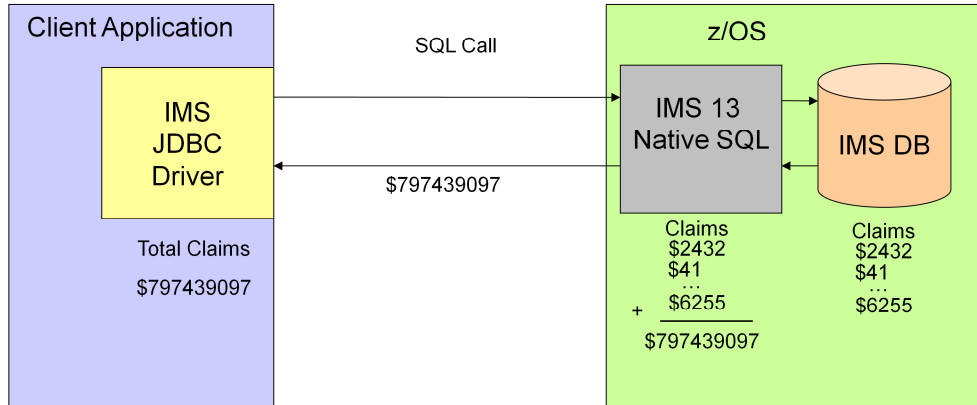
03- IMS13 Integration: 9

For the sake of simplicity to demonstrate the use case, some intermediate components like ODBM and IMS Connect are not shown in this chart. For a full view look at the underlying architecture chart.

In the scenario above a very simple summation call could result in having a large amount of data being transferred over the network. In this case all of the claims data from the year 2011 is being streamed to the client side

SQL Parsing in IMS

- **Solution IMS Java intends to use IMS 13 Native SQL**
 - IMS Service Process
- Example with IMS SQL call handler
 - How much money has my insurance company paid out in claims for the year 2011?
 - `SELECT SUM(CLAIMAMOUNT) FROM CLAIMS WHERE YEAR=2011`



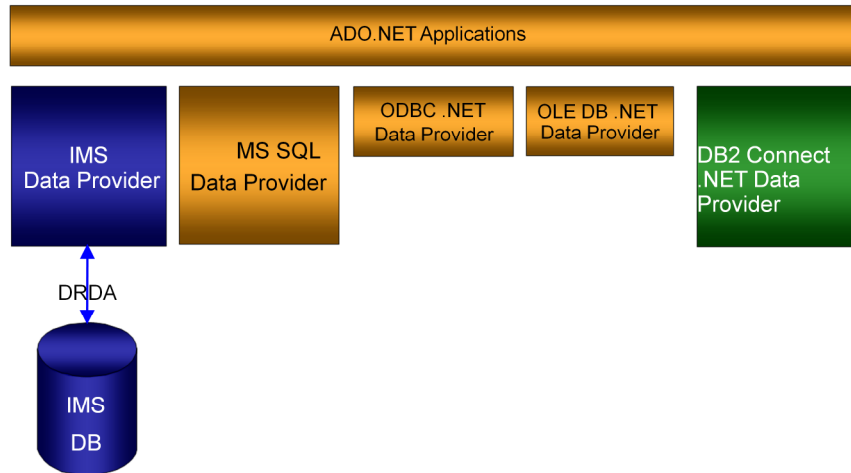
In this picture that Native SQL engine now handles the data aggregation on z/OS and only the final summation is streamed over to the JDBC driver.

IBM IMS Data Provider for Microsoft .NET

- **Available via the IMS Enterprise Suite V3.1 service process**
 - Based on the .NET Version 4.0 specifications
 - Application developers can use their preferred development environment, such as Microsoft Visual Studio, to call the IMS provided APIs.
 - Support is for IMS 13 or later
- **Benefit**
 - Provides the solution for Microsoft .NET-based applications to access and manipulate IMS data

IMS Data Provider for Microsoft .NET simplifies development of Microsoft .NET applications that are written in C#, Visual Basic, and other ADO.NET-compliant languages to access IMS data

IBM IMS Data Provider for Microsoft .NET Architecture



03-IMS13 Integration: 12

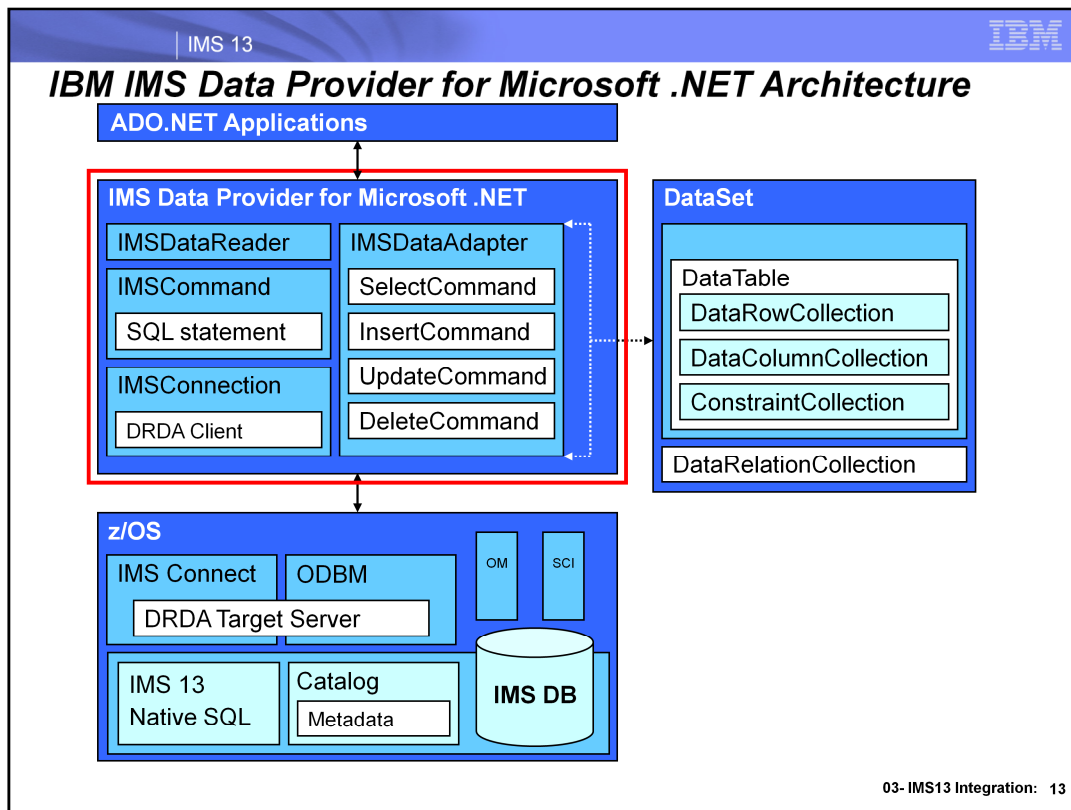
The .NET Framework from Microsoft is the building blocks to build applications by using the class library in the framework, supporting several programming languages that allows language interoperability. Programs written for the .NET Framework run in the Common Language Runtime (CLR) runtime environment, an application virtual machine that provides crucial services such as security, memory management, and exception handling. The class library and the CLR together constitute the .NET Framework.

ActiveX Data Object for .NET, or ADO.NET, is a set of software components for accessing data and data services. ADO.NET is part of the base class library that is included with the Microsoft .NET Framework.

.NET data providers are software components that enable an ADO.NET consumer to interact with a data source. The .NET Framework includes the System.Data.Common namespace, which provides a set of base classes that can be shared by any .NET data provider. This namespace facilitates a generic ADO.NET database application development approach with a consistent programming interface.

MS SQL is accessed directly via ADO.NET API. For 3rd party DMBS and other Data Access standards Data Providers are needed.

.NET comes with Data Providers for ODBC and OLE out of the box. IMS is added to the list of supported DBs by implementing .NET Data Provider.



Red box – contains all required ADO.NET interfaces (DataReader, DataAdapter, Command, Connection). Data retrieved from IMS will be stored in a standard DataSet class (in-memory datastore) or IMSDataReader which is similar to ResultSet in Java.

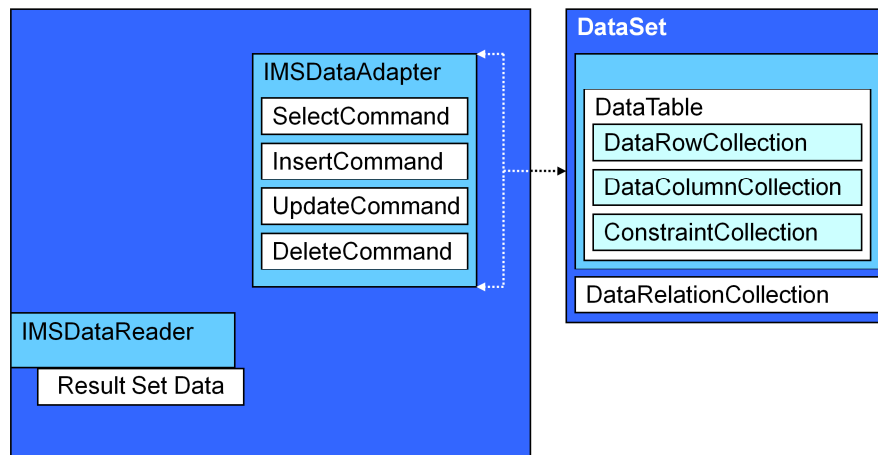
The .NET DP will be a DLL that .NET apps will be using directly. The DLL connects to z/OS via TCP/IP and DRDA protocol (just like IMS Type 4 Universal Drivers).

IMS Connect and ODBM address spaces are required and together they form DRDA Target Server. In IMS, Native SQL takes care of processing SQL queries and sending results back. Catalog feature is used for metadata, so no offline metadata is necessary (in fact, it will not be supported). Both online and offline (disconnected) modes of operation will be supported.

IMSConnection class includes the properties and methods that is required to establish a connection with the IMS DRDA Server

IMSCommand class represents an SQL statement to execute against a data source by specifying what type of SQL interaction you want to perform with a IMS database. This object together with the IMSConnection object provides the IMS .NET Data Provider customers with a connected data approach.

IBM IMS Data Provider for Microsoft .NET Architecture



DataSet a memory-resident relational representation of IMS data managed by IMSDataAdapter

DataReader object is used for fast-forward reading streams of IMS data

03-IMS13 Integration: 14

Data retrieved from IMS will be stored in a standard DataSet class for disconnect mode processing and DataReader for connect mode processing.

IMSDataReader serves similar purpose as a DataReader in the ADO.NET technology. This object is used for fast-forward reading streams of IMS data. This object cannot be used for writing data. Due to the stream behavior, once some data is read, you must save it for your purpose since you will not be able to go back and read it again.

IMSDataAdapter similar to the DataAdapter object in ADO.NET manages connection and interaction with IMS and gives the users of the IMS .NET Data Provider a disconnected behavior. This object opens a connection only when required and closes it as soon as it has performed the intended task.

Dataset is an in-memory data store and the user keeps manipulating the dataset till they are ready to push the change back to IMS using the IMSDataAdapter. Note that the IMSConnection object needs to be instantiated before calling the IMSDataAdapter object, but not opened. The IMSDataAdapter will open and close the connection during Fill and Update method calls transparently to the users.

Connecting to IMS databases

▪ IMSConnection Class

- Server=myserver, - IMS Connect Host name/Network address
- 5555; - IMS Connect Default DRDA Port Number
- Database=DFSSAM09; - PSB name in IMS Catalog
- Datastore=IMS13A; - IMS name defined to ODBM
- UID=imsadm; - SAF UserID
- PWD=ab1d; - SAF Password
- Connect Timeout=30; - time to wait for establishing a connection

Connect and Disconnect Mode Concepts

- **Connected architecture**
 - Every select, insert, update, delete is an access to the IMS database

- **Disconnected architecture**
 - Fetch the data from IMS database into local DataSet
 - Manipulate data w/o accessing the IMS database
 - When complete commit all your changes to the IMS database.

In connected architecture for each data add, retrieve, update, and delete operation requires access to the database. such as for every select, insert, delete ,update your application will access the database.

In disconnected architecture once you fetch the data you can perform operations to the data without accessing the database. and when you have completed all the data operations then you commit all your changes to database.

Connect mode

1. IMSConnection Open connection

Obtain socket connection from connection pool

2. IMSCommand build SQL string

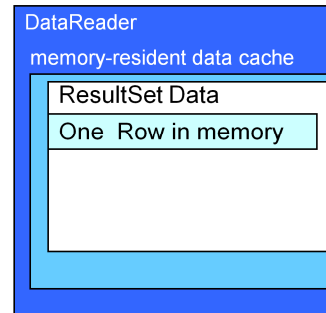
3. IMSDataReader retrieve data from IMS store into DataReader

- First SQL call allocates a program specification block (PSB)
- Database locks held until Close connection

4. IMSConnection Close connection

Deallocates the PSB

Return socket connection to connection pool



03-IMS13 Integration: 17

Use DataReader when:

Dealing with large volumes of data—too much to maintain in a single cache.

Reduce the memory footprint of your application.

Want to avoid the object creation overhead associated with the DataSet

Want to perform data binding with a control that supports a data source that implements IEnumerable

Wish to streamline and optimize your data access

Reading rows containing binary large object (BLOB) columns

Note each Open, SQL call and Close Connection causes an allocate/deallocate of the PSB

C# Application Connect mode Read only

```
Example.cs x
using IBM.Data.IMS;

static void IMSReader()
{
    // Use connection string to configure connection properties
    IMSConnection connection = new IMSConnection("Data source = MyIMS,5555;
        Database = Insurance");
    // Establish connection to IMS database
    connection.Open();

    // Specify SQL query in the IMSCommand object
    IMSCommand command = new IMSCommand("SELECT * FROM PCB.CUSTOMERS",
        connection);

    // Execute query via the DataReader object
    IMSDataReader reader = command.ExecuteReader();

    // Iterate through results and output on the screen
    while (reader.Read())
        Console.WriteLine(reader.GetString(0));

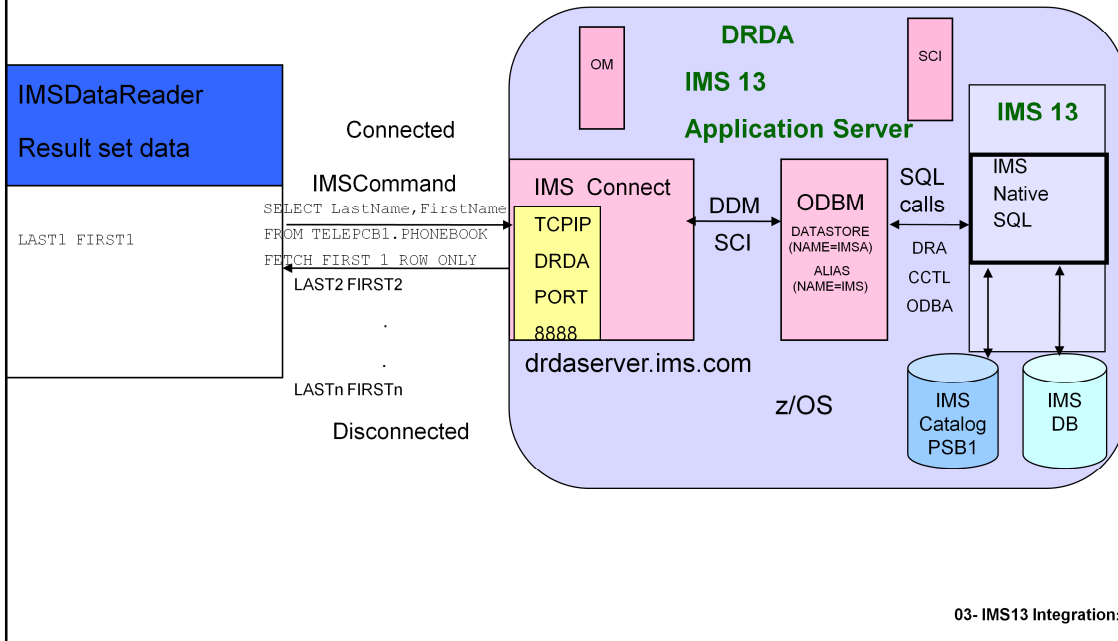
    // Close the reader
    reader.Close();

    // Close the connection
    connection.Close();
}
```

03-IMS13 Integration: 18

Point of example is to show how IMSDataReader is used for read only processing while in a connection

IMSDataReader



An IMSDataReader implements a DataReader in the ADO.NET technology. This object is used for fast-forward reading streams of data.

This object cannot be used for writing data. Due to the stream behavior, once some data is read, you must save it for your purpose since you will not be able to go back and read it again.

```
SELECT LastName,FirstName FROM TELEPCB1.PHONEBOOK
FETCH FIRST 1 ROW ONLY
```

C# Application Connect Mode change data

```
Example.cs X
using IBM.Data.IMS;

static void IMSWriter()
{
    // Use connection string to configure connection properties
    IMSConnection connection = new IMSConnection("Data source = MyIMS,5555;
        Database=Insurance");
    // Establish connection to IMS database
    connection.Open();

    // Specify SQL command in the IMSCommand object
    IMSCommand command = new IMSCommand("INSERT INTO PCB.CUSTOMER (NAME,
        POLICY) VALUES ('EVGENI', 1210050000)", connection);

    // Execute command
    int i = command.ExecuteNonQuery();

    // Close the connection
    connection.Close();
}
```

• INSERT, UPDATE and DELETE commands are used identically

Point of example is to show how IMSCommand is used for SQL commands that can change the IMS data while in a connection

Disconnect mode

- **DataSet** - Disconnected memory-resident data cache
 - Connections are not maintained for long periods
 - Database locking does not occur

1. IMSConnection set connection parms
2. IMSCommand build SQL string
3. IMSDataAdapter

Open connection

Retrieve data from IMS store into DataSet

Allocate PSB

Close connection

Deallocate PSB

4. Change data in DataSet

1. IMSDataAdapter FILL/Update Methods

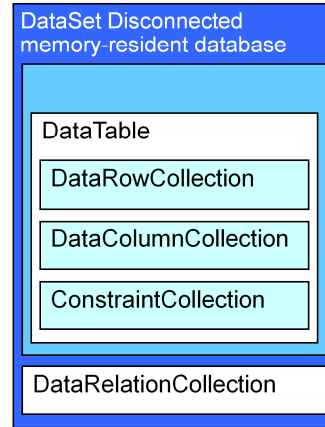
Open connection

Allocate PSB

Write changes from DataSet to IMS

2. Close connection

Deallocate PSB



03-IMS13 Integration: 21

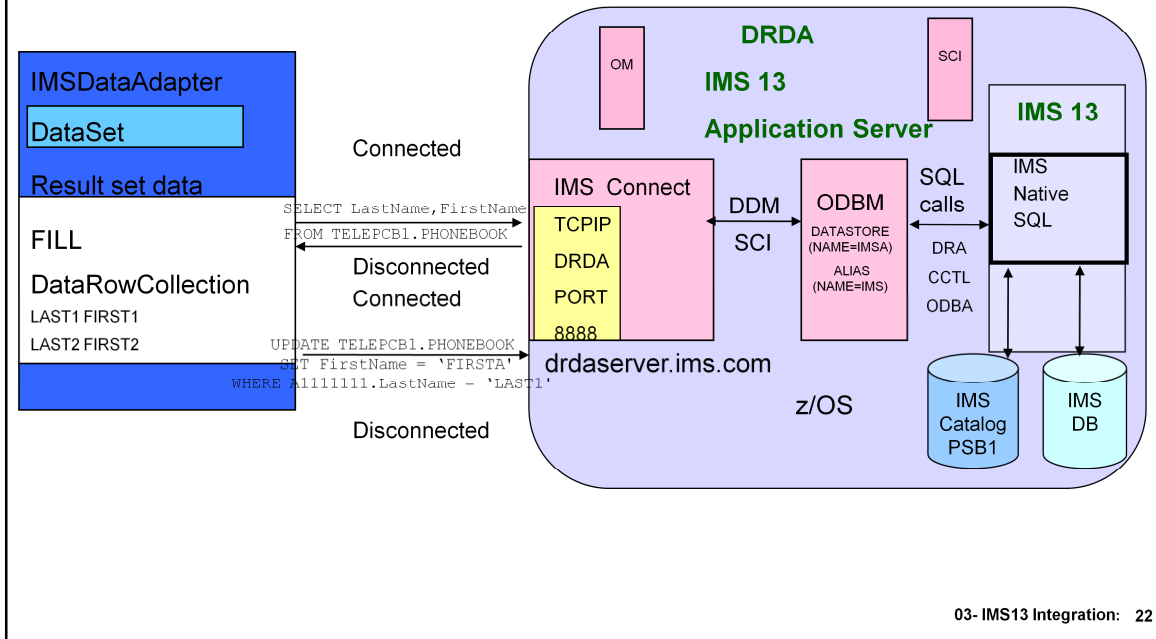
Use DataSet when:

You require a disconnected memory-resident cache of data, so that you can pass it to another component or tier within your application.

You are working with data retrieved from multiple data sources, such as multiple databases, tables, or files.

You want to perform data binding against a control that requires a data source that supports List.

IMSDataAdapter

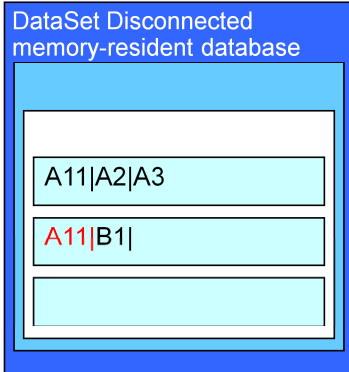


Similar to the DataAdaptor object in ADO.NET, the IMSDataAdapter manages connections and interactions with IMS and gives the users of the IMS .NET Data Provider a disconnected behavior. This object opens a connection only when required and closes it as soon as it has performed the intended task. Here the SQL statement is held in its properties and when it is executed the result set is filled into the corresponding DataSet specified by the application.

The Dataset is an in-memory data store and the user keeps manipulating the dataset till they are ready to push the change back to IMS using the IMSDataAdaptor. Note that the IMSConnection object needs to be instantiated before calling the IMSdataAdaptor object, but not opened. The IMSDataAdapter will open and close the connection during Fill and Update method calls transparently to the users.

Disconnected Mode Step 1 Populate Dataset

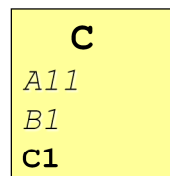
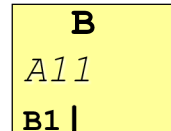
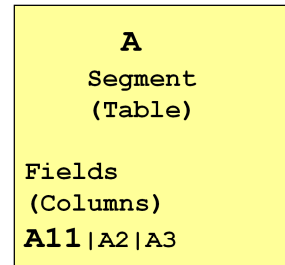
IMSDataAdapter Fill DataSet



```
SQL SELECT *
FROM PCB.A, PCB.B
WHERE A=A11 and B=B1
```

Row 1

Row 2

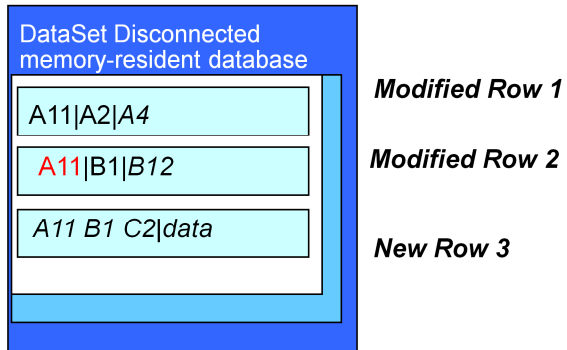


This slide provides a mapping of IMS hierarchical database concepts and relational database concepts. It also shows how IMS Foreign key is used to provide referential integrity. This is used to enforce ADO.NET fill/update processing.

Fill your **DataSet** with current data by using the primary key values of the rows returned by the **SelectCommand**

**Disconnected Mode
Step 2 change data in
Data Set**

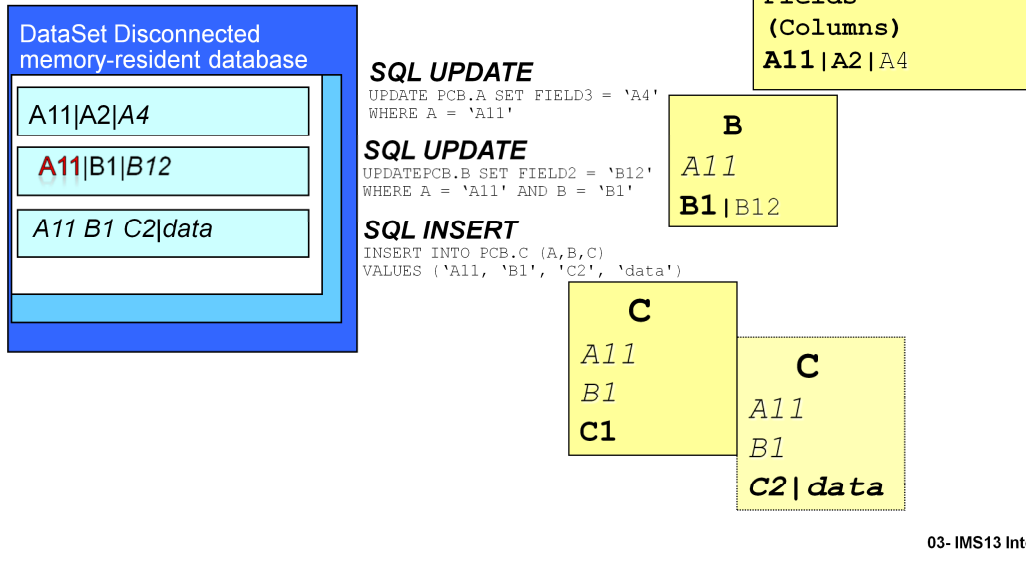
Change DataSet



Make changes to DataSet while still disconnected from IMS

Disconnected Mode Step 3 Update IMS database

IMSDataAdapter UPDATE database



Any row in the returned result set whose primary key corresponds to an existing row in the DataSet will be used to update that row, and that row's state will always become DataRowState.Modified, *even if the returned row is identical to the current row*

Any row in the returned result set whose primary key doesn't correspond to any existing row will be used to create a new row, and that row's state will become DataRowState.Added

Any row in the DataSet that doesn't correspond to a row in the returned result set will stay at DataRowState.Unchanged

For this example, the Update method executes a two UPDATE statements, followed by an INSERT statement due to the ordering of the rows in the DataSet.

```
UPDATE PCB.A SET FIELD3 = 'A4' WHERE A = 'A11'
```

```
UPDATE PCB.B SET FIELD2 = 'B12' WHERE A = 'A11' AND B = 'B1'
```

```
INSERT INTO PCB.C (A,B,C) VALUES ('A11', 'B1', 'C2', 'data')
```

Example of Using Disconnected Mode

```
Example.cs X
IMSConnection conn = new IMSConnection("Data source=myims,5555;Database=Insurance");
string queryString = "SELECT * FROM PCB.CUSTOMERS";

//Create new DataAdapter object
IMSDataAdapter adapter = new IMSDataAdapter(queryString, conn);

//Create CommandBuilder object for automatic command generation
IMSCommandBuilder builder = new IMSCommandBuilder(adapter);
DataSet customers = new DataSet();

// This call will execute IMSDataReader to fill DataSet with query results
adapter.Fill(patients, "Patients");

foreach (DataRow pRow in customers.Tables["Customers"].Rows)
    Console.WriteLine(pRow["NAME"]);

//Change value in Dataset
DataRow myRow = customers.Tables["Customers"].Rows[1];
myRow["CUSTNUM"] = 25;

//Let CommandBuilder generate Update query
builder.GetUpdateCommand();

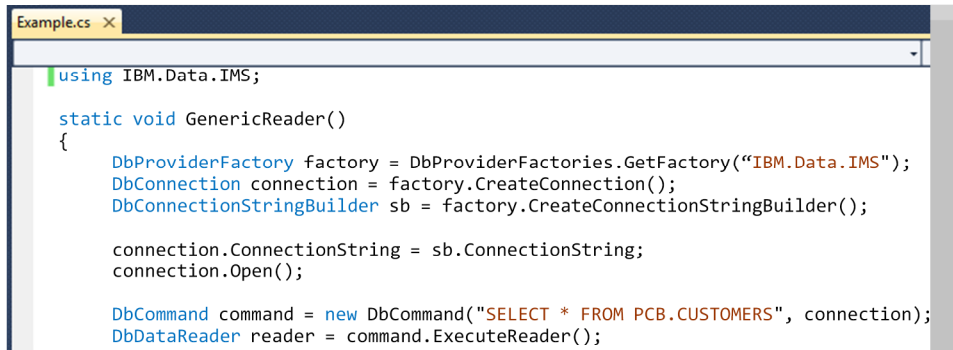
//Flush the changes back to IMS
adapter.Update(customers, "Customers");

//Clear existing Dataset and refill it with updated data
customers.Clear();
adapter.Fill(customers, "Customers");
```

03-IMS13 Integration: 26

Generic Coding

- .NET Framework outlines the "**generic coding**", or "**factory-based**" interface that is supported by IMS Data Provider
 - Facilitates generic ADO.NET application development, constant programming interface across different databases
 - When using this technique, proprietary class names, such as [IMSConnection](#), are replaced with common names, such as [DbConnection](#)



```
Example.cs X
using IBM.Data.IMS;

static void GenericReader()
{
    DbProviderFactory factory = DbProviderFactories.GetFactory("IBM.Data.IMS");
    DbConnection connection = factory.CreateConnection();
    DbConnectionStringBuilder sb = factory.CreateConnectionStringBuilder();

    connection.ConnectionString = sb.ConnectionString;
    connection.Open();

    DbCommand command = new DbCommand("SELECT * FROM PCB.CUSTOMERS", connection);
    DbDataReader reader = command.ExecuteReader();
}
```

03-IMS13 Integration: 27

To write generic code that either is not tied to a particular database or supports several different databases, the .NET Framework provides a factory-based interface that is supported by the IMS™ Data Provider for Microsoft .NET.

The .NET Framework features a namespace that is called `System.Data.Common`, which includes a set of base classes that can be shared by any .NET data provider. This namespace facilitates a generic ADO.NET database application development approach, offers a constant programming interface across different databases, and enables the factory design model for client database applications. These features increase design flexibility and reduce module maintenance cost.

When you use this technique, proprietary class names such as `IMSConnection` are replaced with common names, such as `DbConnection`.

Transactions

- IMS Data provider supports Local **Transactions**
 - **IMSTransaction** object is responsible for rolling back and committing database transactions

```
Example.cs x
IMSCommand command = connection.CreateCommand();
IMSTransaction transaction;

// Start a local transaction.
transaction = connection.BeginTransaction("SampleTransaction");
command.Transaction = transaction;

try
{
    command.CommandText = "INSERT INTO REGION (ID, NAME) VALUES (100, 'Portland')";
    command.ExecuteNonQuery();

    command.CommandText = "INSERT INTO REGION (ID, NAME) VALUES (200, 'Vegas')";
    command.ExecuteNonQuery();

    transaction.Commit(); // Attempt to commit the transaction.
}
catch (Exception ex)
{
    transaction.Rollback(); // Attempt to roll back the transaction.
}
```

03-IMS13 Integration: 28

By default, every SQL command is autocommitted.

If multiple SQL statements need to be executed as a single transaction, the Transaction property of the IMSCommand object must be initialized to an IMSTransaction object. An IMSTransaction object is responsible for rolling back and committing database transactions. When the application creates an IMSTransaction object by calling the BeginTransaction() method on the IMSConnection object. All subsequent operations associated with the transaction (for example, committing or aborting the transaction), are performed on the IMSTransaction object.

Note when using IMSTransaction CLOSE will rollback any pending transactions:

```
// Close the connection
connection.Close();
```


Troubleshooting the IMS Data Provider for Microsoft .NET

- **Logging and tracing**
 - A configuration file is used to enable logging and tracing
 - 0 - Off
 - 1 - Error
 - 2 - Warning
 - 3 - Info
 - 4 - Verbose
- **Error Handling**
 - IMSEException class collects instances of the IMSError class instance
 - an error occurs on a database operation in your application
- **Error messages**
 - IMS Data Provider for Microsoft .NET error messages start with IXN.

Error messages and logging are available to facilitate troubleshooting.

An instance of the IMSError class is created whenever an error occurs on a database operation in your application. Each instance of IMSError created by the IMSDataAdapter is managed by the IMSErrorCollection class, which in turn is created by the IMSEException class.

Error handling

IMS Data Provider for Microsoft .NET provides an IMSEException class that collects instances of the IMSError class. Catching exceptions in your code can prevent the application from failing and provide a relevant error message to your user.

Logging and tracing

You can enable logging and tracing by providing a configuration file and specifying the trace level.

Error messages for IMS Data Provider for Microsoft .NET

Error messages for the IMS Data Provider for Microsoft .NET starts with IXN. Some error messages that are related to connections are followed by errors from the DRDA server. Errors that are related to SQL queries often include an error code from the SQL support in IMS.

Security

▪ Security Considerations

- IMS Data Provider for Microsoft .NET supports Secure Sockets Layer (SSL) protocol when the IMS host system uses the z/OS Communications Server IP Application Transparent Transport Layer Security (AT-TLS).
- IMS managed Userid and PSB security check
 - Resource access security (RAS) or APSB
 - A security check is performed by RACF to determine if the user is authorized to use the PSB

Restrictions

- The IMS Data Provider for Microsoft .NET has the following restrictions:
 - Stored procedures currently are not supported.
 - SQL queries must be issued directly from .NET applications.
 - Global transactions are not supported.
 - The command builder class that is used for automatic generation of commands in disconnected mode is not implemented
 - Not all SQL statements are supported.
 - see the **IMS Version 13 Application Programming APIs** information.
 - Not all IMS data types are supported in the current release.
 - For the list of supported IMS data types,
 - **IMS Enterprise Suite > IMS Enterprise Suite Version 3.1 > IMS Data Provider for Microsoft .NET > Developing .NET applications with the IMS Data Provider for Microsoft .NET**

In the 1st release, .NET data provider only supports local transactions (single participant).

System Requirements

▪ **Software requirements**

- IMS Version 13, with the following required functions and APARs:
 - IMS Connect
 - CSL - Open Database Manager (ODBM) , OM and SCI
 - IMS catalog must be properly configured.
 - IMS Version 13 APARs PM96324 and PI05437 must be applied.
- Microsoft .NET Framework Version 4.0 or later
- Microsoft Visual Studio 2010 or later

▪ **Supported Platforms**

- Windows XP systems with Service Pack 2 or later (32- or 64-bit)
- 1.6GHz or faster processor
- 1 GB (32 Bit) or 2 GB (64 Bit) RAM (Add 512 MB if running in a virtual machine)
- 3 GB of available hard disk space

IMS Data Provider for Microsoft .NET installation

- Go to the **IMS Enterprise Suite download site** and log in.
- Select IMS Enterprise Suite Version 3.1 and click Continue.
- Select IMS Data Provider for Microsoft .NET and click Continue
 - IMS Data Provider for Microsoft .NET download page
- Select the IMS Data Provider for Microsoft .NET repository file
- Click **Download now** to download the selected files.
- Store the compressed repository file in a accessible location
- You must have IBM Installation Manager Version 1.5.3 or later installed

IMS ES Explorer for Development

IMS Enterprise Suite V3.1 Explorer for Development

The screenshot displays the IMS Explorer interface with several components:

- SQL Editor:** Contains the query: `SELECT HOSPNAME, HOSPCODE, HOSPLL FROM PCB01.HOSPITAL`
- Schema Diagram:** A hierarchical tree view showing segments like HOSPITAL, ORDER, SALES, STOCK, SALESINF, and STOCKSALE, with their respective fields and total lengths.
- SQL Results:** A table showing the output of the query:

Status	Operation	Da	HOSPLL	HOSPCODE	HOSPNAME	
✓	Succeed	select * from pcb...	8/2	1	R.1210010000A	ALEXANDRIA
✓	Succeed	select * from pcb...	8/2	2	R.1210020000A	SANTA TERESA
✓	Succeed	select * from pcb...	8/2	3	R.1210030000A	SANTA CLARA
✓	Succeed	select * from pcb...	8/2	4	R.1210040000A	NEW ENGLAND
- Properties Panel:** Shows details for the selected segment, including fields like HOSPCODE, HOSPLL, and HOSPNAME.
- Annotations:**
 - An arrow points to the SQL editor with the text: "Generate SQL to access data".
 - An arrow points to the schema diagram with the text: "Edit PSB sensitive segments and attributes".
 - An arrow points to the schema diagram with the text: "See database relationships change DBD field attributes".

Ability to access the IMS Catalog

03- IMS13 Integration: 35

The IMS™ Enterprise Suite Explorer for Development (IMS Explorer) is an Eclipse-based graphical tool that simplifies IMS application development tasks such as updating IMS database and program definitions, and using standard SQL to manipulate IMS data. Its graphically-driven editors allow the user to display the segment hierarchy for any IMS database, including logical relationships and secondary indexes. It also provides user assistance in the form of rich GUI controls and contextual help to reduce IMS development effort.

The Explorer's graphical editors can be useful for the importing, visualization, and editing of IMS database and program definitions. You can also use the IMS Explorer to easily access and manipulate data stored in IMS by using standard SQL.

IMS Enterprise Suite Explorer for Development

- Enhancements for V3.1 include:
 - Ability to import large numbers of DBDs and PSBs.
 - Automatic imports of referenced DBDs when DBDs and PSBs from the IMS catalog or the host are imported.
 - Ability to import COBOL and PL/I data structures from the host.
 - Support for transaction unit testing.
 - Uses IMS Connect API for Java
 - Can be used in addition to IBM IMS Batch Terminal Simulator
 - Support for IMS catalog navigation.
 - View IMS resources in an IMS catalog-enabled system
 - Import IMS resources into IMS Explorer projects from the view.
 - Show all instances of a given resource or find referenced DBDs or PSBs
 - A Problems View for troubleshooting information
 - Shows resource problems and missing files

Creating and running a unit test case for application debugging

- Create a transaction test project File > New > IMS Explorer Transaction Test Project.
- **Define the input and output messages in the transaction.**
- Create a test case New > IMS Transaction Test Case.
- **Develop a script of the dialog between an existing or future front-end application and one or more transactions running in IMS.**
- Define a IMS server New > Server click IBM > IMS Transaction Server.
- From the test project, right-click and click Run As > Run Configurations.
- **Run the test case.**

03-IMS13 Integration: 37

You can create unit test cases and provide input message data in human readable format for debugging. After you create a unit test case, you can create variations of it with different input message data, to easily exercise different code paths in the IMS transaction.

IMS transaction unit testing

Run an IMS transaction test case

Execution time is 0.61400 seconds for the input and output message pair myhandout.

Runtime configuration: Test Telephone DTP Test case name: hursndup.tbc

Input messages

Input and Output Message Pair	Field value	Field Length
myhandout		
Segment 1		
INPUT_MSG		
IN_LJ	99	2
IN_ZZ	9	2
IN_TRCTM	RTMNO	10
IN_CHD	DISPLAY	8
IN_NAME1	LAST1	10
IN_NAME2		10
IN_BTH		10
IN_ZIP		7

0

Output messages

View output message with: DTPNO - OUTPUT

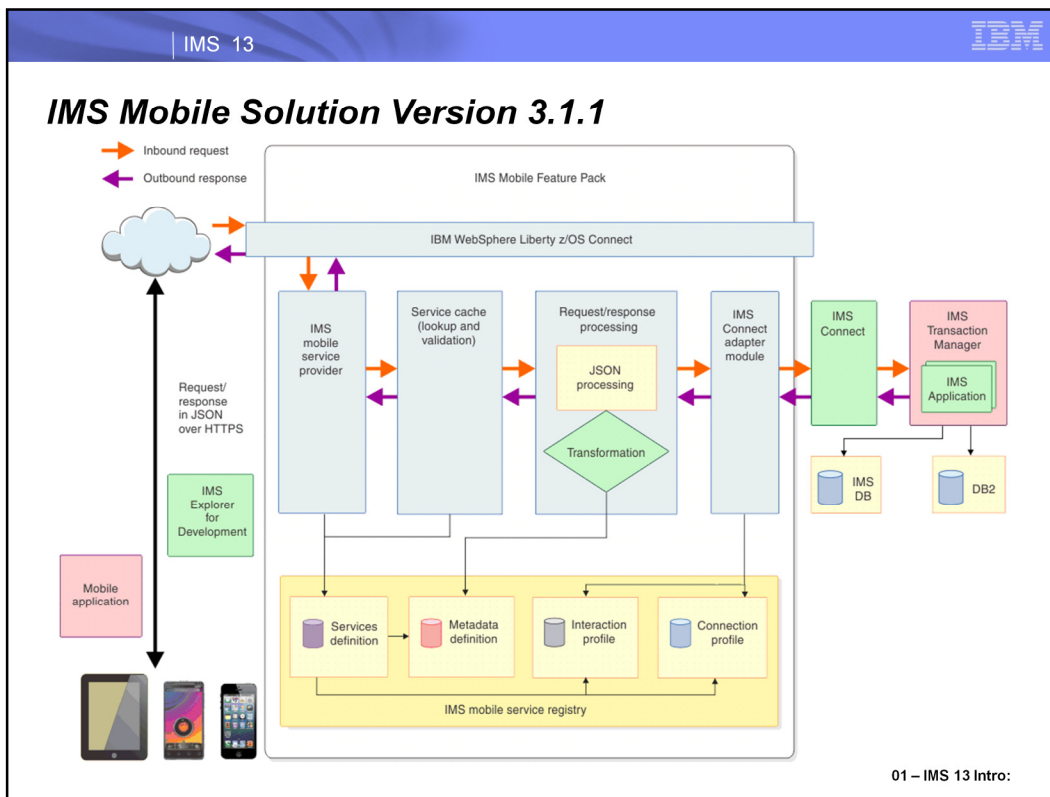
Input and Output Message Pair	Field value	Field Length
myhandout		
Segment 1		
OUTPUT_MSG		
OUT_LJ	99	2
OUT_ZZ	9	2
OUT_MSG	ENTRY WAS DISPLAYED	40
OUT_CHD	DISPLAY	8
OUT_NAME1	LAST1	10
OUT_NAME2	FIRST1	10
OUT_BTH	9-112-1111	10
OUT_ZIP	001801	7
OUT_RESPNO	0003	4

?

Close 38

IMS ES 2.2 Explorer - Importing data structures

- **Importing data structures**
 - metadata field layouts imported from data structure files (COBOL copybooks or PL/I includes)
 - Allow COBOL and PL/I importers to directly import data structures from zOS
 - Just like DBD or PSB source import from z/OS
- **DBD file**
 - select Import COBOL or PL/I Data Structures.
 - select the data structure file that contains the COBOL copybooks or PL/I includes that you want to import.
 - Optional:
 - using source from the IMS catalog, specify the map and case name to import the data structure
- **Benefit**
 - RDz is not required
 - no need to shell share with RDz



The IMS mobile solution that is enabled by IMS Mobile Feature Pack provides a RESTful endpoint to access IMS transactions, servicing RESTful requests and responses in JavaScript Object Notation (JSON) data format.

The solution comprises the following major parts:

IBM® WebSphere® Liberty z/OS® Connect (z/OS Connect), which serves as the IMS gateway server for IMS services. The z/OS Connect feature in WebSphere Application Server for z/OS Liberty Profile is a fast, secure, scalable, and reliable connector that can reach any z/OS asset, including IMS, and delivers common services and management for consistent operations.

The IMS Mobile feature, which is a service provider for z/OS Connect that needs to be installed as a feature. The IMS Mobile feature includes the following functions: A data transformation module that converts request messages from the JSON format to the native representation of the input message and then converts the response messages to JSON. An IMS Connect adapter module that interacts with IMS Connect for IMS Transaction Manager (TM) access. A service management and administration interface that is used by IMS Explorer for Development to provide the user interface for service creation, testing, and management. The connection and interaction information and the metadata about the service are stored in the service registry. An in-memory service cache provides shorter response time for services.

The diagram shows the request and response processing and the components that are involved at run time. When a request comes in, z/OS Connect handles the user access control and then forwards the request to the IMS mobile service provider. The IMS mobile service provider is invoked, and the service cache is looked up or updated, depending on the type of the request. The request message is converted from JSON to the native representation of the input message (for example, bytes array or arrays for segments in COBOL and PL/I applications) by the data transformation module. The request then goes through the IMS Connect adapter module to interact with IMS Connect to access IMS transactions.

Cross-product integration

- The IMS Explorer supports cross-product integration (shell-sharing) with the following products:
 - IBM® Rational® Developer for System z®
 - IBM Data Studio
 - IBM Problem Determination Tools Plug-ins for Eclipse
 - IBM Explorer for z/OS®
 - IBM CICS Explorer® Software Development Kit (SDK)
 - IBM Rational Team Concert™

DLIModel

- IMS Enterprise suite for z/OS , V2.1 is the last release to provide the DLIModel Utility plug-in.
- Explorer provides ability to import DLIModel projects
 - Explorer does not support XML DB or DB Web Services
- Customers using IMS Database Web Services should transition to using the IBM Data Studio which leverages the IMS Universal Drivers

Customers using DLIModel Utility support for Database Web Services should transition to using the IBM Data Studio Database Web Services support, which leverages the IMS Universal Drivers.

IMS ES SOAP Gateway

IMS Enterprise Suite SOAP Gateway

- Enhancements for V3.1 include:
 - Support for 64-bit z/OS for extended memory usage.
 - Support for the send-only with acknowledgement protocol for synchronous callout
 - Ability to run the SOAP Gateway management utility commands in batch mode in one JVM instance

64-bit support for z/OS

SOAP Gateway now runs on the z/OS platform in 64-bit mode, allowing organizations to take advantage of their 64-bit operating environment for extended memory usage.

Send-only with ACK support for synchronous callout

Send-only with acknowledgement protocol support for synchronous callout allows SOAP Gateway to receive a final confirmation that the response message was delivered to the original IMS application that issued the callout request. This confirmation provides SOAP Gateway users additional information about whether a callout response message was sent to IMS and whether IMS received the message.

SOAP Gateway management utility batch mode support

Administrators can now use the batch mode of the management utility to facilitate web service deployment and server management for better performance and manageability. Instead of issuing one command at a time, each with its own JVM instance, you can pass a file with a list of commands to the SOAP Gateway management utility **iogmgmt -batch** command for execution as a batch in one JVM instance.

Enhanced security cipher suite support

SOAP Gateway is enhanced to use the FIPS 140-2 approved cryptographic provider(s); IBMJCEFIPS (certificate 376) and/or IBMJSSEFIPS (certificate 409) for cryptography. The certificates are listed on the NIST web site at <http://csrc.nist.gov/cryptval/140-1/1401val2004.htm>. SOAP Gateway also adds the support for Transport Layer Security (TLS) V1.2 and for cipher suites with key length of 2048 and key strength of 112 bit, as required by NIST SP800-131a.

Transaction tracking

- **SOAP Gateway transaction tracking IDs and logging**
 - SOAP Gateway can attach 40 byte *horizontal* tracking ID to inbound web service request
 - ID is sent with the inbound request through IMS Connect to the target IMS application and returned with the response message to SOAP Gateway
 - IMS Connect tracking ID captured by the IMS Connect Event Recorder exit routine (HWSTECL0).
 - This information can be consumed by the IBM IMS Connect Extensions for z/OS and equivalent tools.
 - For IMS 12 requires APAR PM69983 applied to IMS Connect
 - IMS log records for transactions include the tracking ID
 - IBM IMS Performance Analyzer for z/OS and IBM IMS Problem Investigator for z/OS, or equivalent tools, to inspect IMS log records.
 - Benefits
 - Correlates transactions between SOAP Gateway, IMS Connect, and IMS
 - Provides information for diagnostic purposes

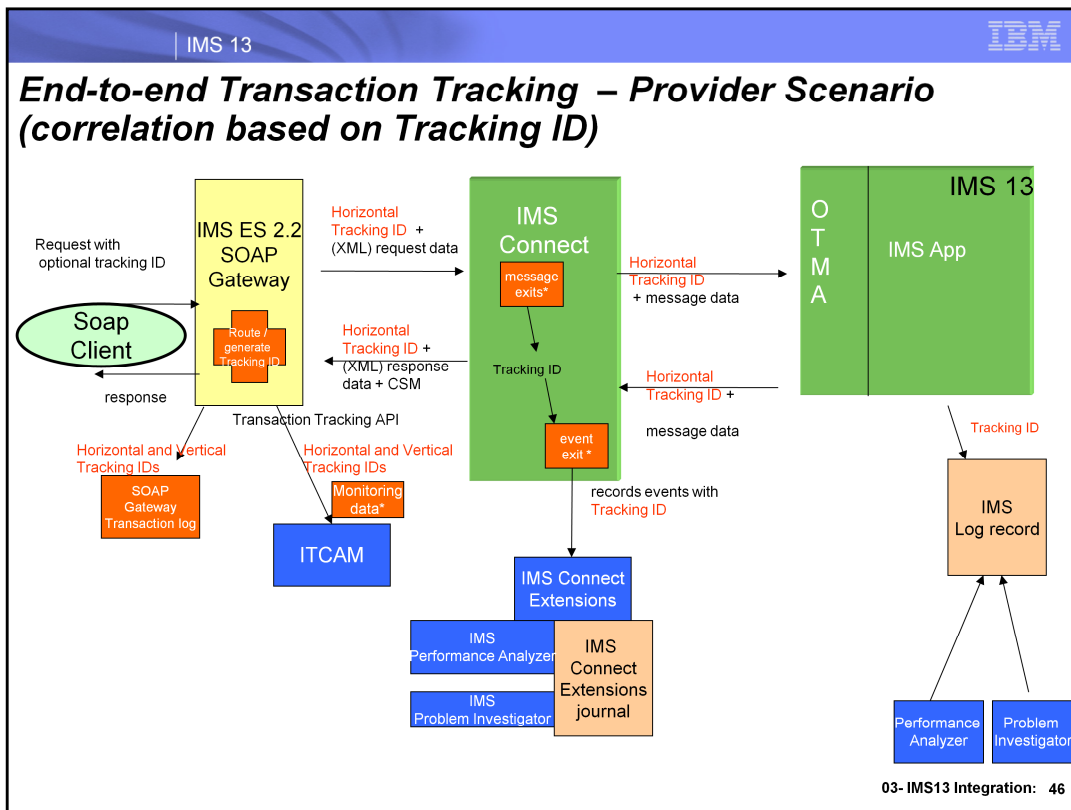
03-IMS13 Integration: 45

SOAP Gateway can now attach a 40-byte message ID to incoming request messages for web services. This ID is sent with the inbound request through IMS Connect to the target IMS application, and is returned with the response message to SOAP Gateway.

Three different types of message ID are supported:

- SOAP Gateway can get the value of the messageID element in the incoming SOAP message header, and use that value as the message ID.
- SOAP Gateway can get the value of a user-specified element in the incoming SOAP message header, and use that value as the message ID.
- SOAP Gateway can generate a unique ID for every incoming SOAP message.

Requirement: IMS 12 with service for APAR PM69983 applied to the target IMS Connect host is required to use horizontal IDs.

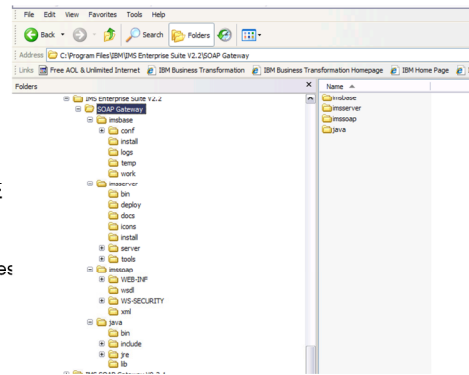


IBM Tivoli Composite Application Manager for Transactions (ITCAM) Transaction Tracking API (TTAPI).

IBM Tivoli Composite Application Manager for Transactions (ITCAM) data collector

Advanced installation and maintenance

- **Server is divided into three components**
 - Imsserver – can be mounted as READ only
 - Contains the servers executable code
 - Imibase – can be mounted as READ and WRITE
 - Contains the servers configurations and log
 - Imsoap - can be mounted as READ and WRITE
 - Contains the user-deployed web service-related files
 - WSDL's , correlators, connection bundles
- **Benefits**
 - easier to apply maintenance
 - allocate additional disk space when more web services are added
- **IBM® Installation Manager for z/OS® V1.5.3**
 - Supports centralized repository delivered through the SMP/E process
- **Benefits**
 - simplifies maintenance process
- **Multiple SOAP Gateway server instances**
 - share a single instance of the Java™ Virtual Machine (JVM).
- **Benefits**
 - reduces the amount of storage required for each additional server instance



03- IMS13 Integration: 47

An installation of IMS Enterprise Suite Version 2.2 SOAP Gateway consists of three parts that can be installed in different directories (or mount points on z/OS). This three-part architecture separates the binary files that run the SOAP Gateway server and the management utility from server configuration files and user files such as

web services-related artifact files. This separation makes it easier to apply maintenance and allocate additional disk space when more web services are added.

For z/OS the SOAP Gateway installation requires IBM Installation Manager V1.5.3. Installation manager simplifies maintenance by allowing the installing and upgrading of the server by pulling from a centralized repository that is delivered through the SMP/E process.

Advanced installation and maintenance

- **Enhanced migration support of server properties**

- the `-migrate` command is enhanced to support the migration of server properties
 - support the migration of existing server properties
 - propagate the server configuration to multiple server instances

```
iogmgmt -migrate /absolute path to IMS_Enterprise_Suite_SOAP_Gateway/
```

- **Benefits**

- eliminates the need to reconfigure the server

03- IMS13 Integration: 48

The `iogmgmt -migrate` now supports the migration of server properties. To migrate from version 2.1, specify the absolute path to the installation of IMS Enterprise Suite Version 2.1.

Clone creates a copy of the web services and server properties from a master Version 2.2 server.

The correlator schema has changed in IMS Enterprise Suite Version 2.2 SOAP Gateway. When you upgrade to IMS Enterprise Suite Version 2.2, the process of migrating existing web services `iogmgmt -migrate` handles the correlator migration.

If the installation directory for the **imsserver** component is read-only, you must change it to read/write mode before you run the migration tool.

WS-Security

- **WS-Security SAML unsigned tokens for synchronous callout applications**
 - Originating Userid (**PSTUSID**) for the IMS synchronous callout application is passed to the external web service for further authentication and authorization
- **Benefit**
 - Provides message-level security for synchronous callout
- **WS-Security enhancement for provider web services**
 - support for Security Assertion Markup Language (SAML) 2.0 sender-vouches signed tokens
- **Benefit**
 - Provides additional message integrity for service provider processing
 - Extends SOAP Gateway support of WS-Security standards

For the synchronous callout scenarios, in addition to transport-level security through basic authentication, server authentication, or mutual authentication, SOAP Gateway now supports message-level security with SAML 1.1 and SAML 2.0 sender-vouches unsigned tokens.

SAML is an XML-based standard developed by Security Services Technical Committee (SSTC) of Organization for the Advancement of Structured Information Standards (OASIS). This standard facilitates:

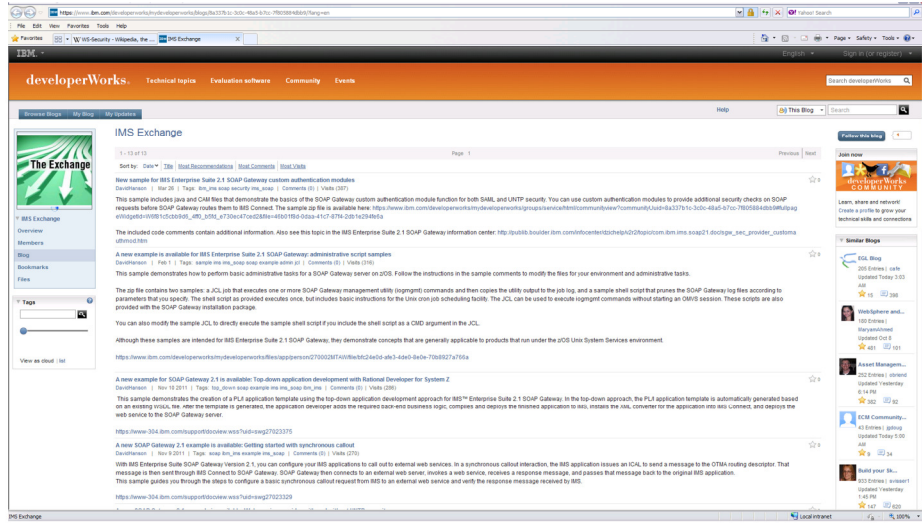
- The exchange of user identity and security attributes information between communicating parties at the SOAP message level.
- The exchange of authentication and authorization assertions across web service transactions.

WS-Security SAML confirmation method is supported for synchronous callout applications by extracting the user ID (the user that initiates the synchronous callout application) from the correlation token and passing it to the external web service.

SOAP Gateway also supports custom authentication modules for accessing the security header for validation before the SOAP request messages are sent out to the external web service server.

IMS ES 2.2 SOAP Gateway New Samples

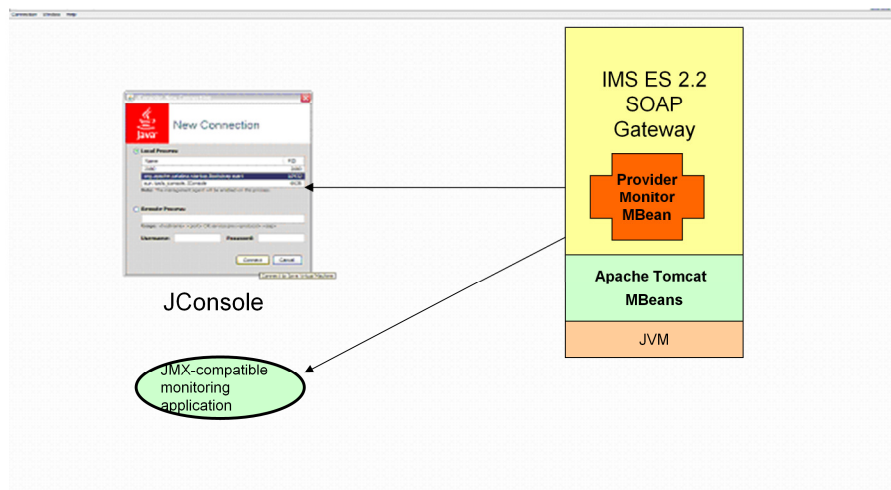
- IMS Exchange web site updated
- Link on IMS Enterprise Suite SOAP Gateway web page



03-IMS13 Integration: 51

JMX MBean interface for web service provider monitoring

- C:\...IMS Enterprise Suite V2.2\SOAP Gateway\java\bin
 - JConsole.exe



03-IMS13 Integration: 52

JMX MBean interface for web service provider monitoring

In addition to the standard JMX instrumentation for the SOAP Gateway JVM, a customized MBean interface for SOAP Gateway `SOAPGatewayProviderMonitorMBean` provides statistics about SOAP Gateway web services activity, connection bundles, and connections to IMS Connect.

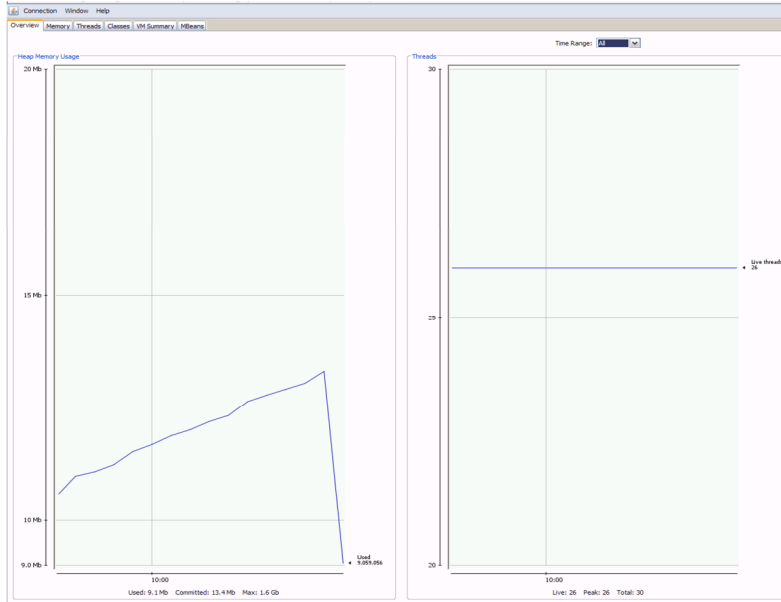
JMX-compatible monitoring application

Apache Tomcat 7.0 MBeans - The Apache Tomcat servlet container, Catalina, is instrumented with JMX MBeans.

org.apache.catalina.startup.Bootstrap - basic JVM information including heap memory usage, thread count, loaded classes, and CPU utilization.

MBeans tab.- expand the folder `com.ibm.ims.soap.server`, then the node `SOAPGatewayProviderMonitorMBean`, and then click **Operations**.

JConsole sample output



Rational Developer for System z

- Rational Developer for System z IMS Edition
 - No support for IMS ES 2.2 Explorer for Development
 - Move to RDz 90 day trial
- Rational Developer for System z 8.5 and later
 - RDz 90 day trial download site
 - <http://www.ibm.com/developerworks/downloads/r/rdz/>

IMS ES Connect API for Java

IMS ES Connect API for Java

- IMS and IMS Connect type-2 commands
- Performance Enhancements
- Support for SendOnly synchronous callout response messages with acknowledgement
 - Function requires that both the following APAR/PTFs are applied
 - IMS Connect 12: PM39569/UK74666
 - IMS OTMA 12: PM39562/UK74653
- Benefits
 - Custom written IMS Connect TCP/IP Java client applications
 - Can send and receive commands to IMS and IMS Connect
 - Can request and receive an indication of response delivery to IMS for synchronous callout processing

03-IMS13 Integration: 56

Function requires that both the following APAR/PTFs are applied:

- IMS Connect 12: PM39569/UK74666
- IMS OTMA 12: PM39562/UK74653

New AIB field - AIBUTKN

Provides optional specification of a 1-8 byte map name included in the OTMA state data prefix to be sent to the callout destination.

IMS 12: PM73135

Sample application program for type 2 commands

```
samples.com.MyCompany.MyProjects.Type2Command.Type2CommandSample.java
myTmInteraction.setImsDatastoreName("MYDATASTORENAME"); myTmInteraction.setInteractionTypeDescription
(ApiProperties.INTERACTION_TYPE_DESC_TYPE2_COMMAND);
    myTmInteraction.setAckNakProvider(ApiProperties.API_INTERNAL_ACK);
    myTmInteraction.setImsConnectCodepage("cp037"); // EBCDIC
    myImsConnectCodepage = myTmInteraction.getImsConnectCodepage();
    // get InputMessage instance from myTMInteraction
    inMsg = myTmInteraction.getInputMessage();
    inData2DByteArray = new byte[1]{};
    inData2DByteArray[0] =
        new String("CMD(QUERY TRAN NAME (IVTNO) SHOW (ALL))").
            getBytes(myImsConnectCodepage);
    // Populate InputMessage object with input byte array
    inMsg.setInputMessageData(inData2DByteArray);
    // Execute interaction
    myTmInteraction.execute();
    // Get output from myTMInteraction
    outMsg = myTmInteraction.getOutputMessage();
    if((outMsg.getImsConnectReturnCode() == ApiProperties.IMS_CONNECT_RETURN_CODE_SUCCESS) &&
        (outMsg.getImsConnectReasonCode() == ApiProperties.IMS_CONNECT_REASON_CODE_SUCCESS))
    {
        // Get the type-2 command response object
        Type2CmdResponse myT2CmdRsp = outMsg.getType2CommandResponse();
    }
}
```

03-IMS13 Integration: 57

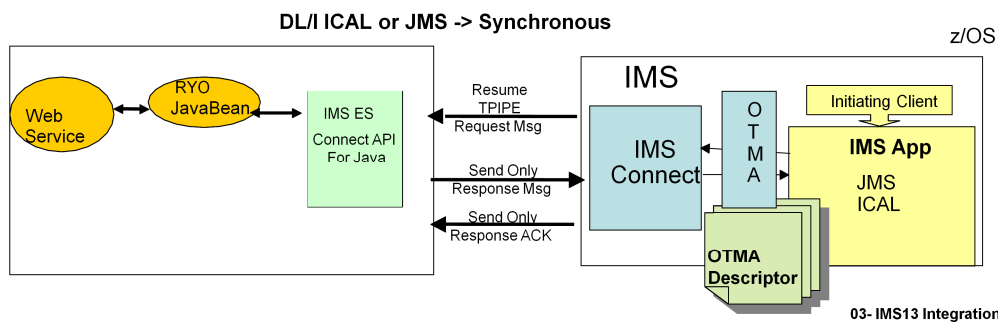
Sample code

IMS ES Connect API for Java

- **SendOnly synchronous callout response messages with acknowledgement**
 - client application gets acknowledgement when the response message is received by IMS

INTERACTION_TYPE_DESC_SENDONLYACK_CALLOUT_RESPONSE

IMS Service Consumer



The send-only protocol for synchronous callout responses can include an acknowledgement to the callout response so the client does not need to switch to receive state after sending the response to IMS Connect

The IMS Enterprise Suite 2.2 Connect API for Java™ includes support for send-only synchronous callout response messages with acknowledgement.

The new interaction type description is set with the following method call for a TmInteraction object: `myTmInteraction.setInteractionTypeDescription(INTERACTION_TYPE_DESC_SENDONLYACK_CALLOUT_RESPONSE)`

This feature allows a client application to get an explicit acknowledgement when the response message is received by IMS, compared to a normal send-only response message that does not get any receipt confirmation.

The client receives the acknowledgement after calling the `execute()` method of the TmInteraction object. After the acknowledgement is received, the return code and reason code from the request status message (RSM) can be retrieved with the existing `getImsConnectReturnCode()` and `getImsConnectReasonCode()` methods. If the acknowledgement indicates a successful message in complete status message (CSM) format, both values are 0.

This interaction type is comparable to the existing `INTERACTION_TYPE_DESC_SENDONLYACK` interaction type, but it is applicable to synchronous callout response messages instead of requests that are initiated from the Connect API for Java client application.

Performance data

- For inbound, 16,000 transactions per second using V2R2 which is a 3% improvement over V2R1
- For callout, 11,000 transactions per second using V2R2 which is a 56% improvement over V2R1

A base line was established for Inbound and Callout with IMS Enterprise Suite V2R2's Connect API for Java. We were able to reach 16,000 transactions per second for inbound using V2R2 IMS Connect API for Java code. The average CPU % used for LPAR1 with 5 CP's running IMS and IMS connect was 58% and for LPAR2 with 6 CP's running the IMS Connect API for Java client code was 13.52%. 100 clients Using the same environment for outbound, we saw 11,000 ICAL requests/sec with CPU% used for LPAR1 at 78.57% and LPAR 2 at 12.12%. 10 TPIPE's and 10 response threads were used for Callout. 150 TPNS clients drove the COBOL IMS echo application sending messages on the TPIPEs.

Based on our performance measurements, ES 2.2 IMS Connect API for Java impacts the total cost of ownership by reducing the CPU usage.

The improvement of going from V2R1 to V2R2 was 3% for inbound and 56% for outbound.

IBM IMS Explorer for Administration

IBM IMS Explorer for Administration

- **Web-based console to operate and administer IMS**
 - Graphically administer IMS Databases and Transactions
 - Connects to the IMS Operations Manager through IMS Connect
 - Discover IMS resources
 - Show the health of the resources
 - query, start, and stop IMS resources
 - View IMSplex
 - visualize relationships between various IMS resources
 - transactions, programs and databases in one view
 - Replacement for IMS Control Center
- **Software requirements**
 - IMS Explorer for Administration is available through
 - APAR PM94292 extension of the Administration Console component
 - IBM Tools Base for z/OS , V1.4 (5655-V93)
 - IMS Version 12
 - IMS Connect
 - Common Service Layer OM and SCI
 - Supported web browser
 - Firefox, Internet Explorer, Safari

The Explorer for Administration provides accessibility from any supported web browser to manage IMS resources

Explorer for Administration – Select the Resource

IBM Tools Base Administration Console for z/OS View Configure

Resources

Search

Custom Groups

Custom Groups

IMSPlex View

Search Results

Click Add

Resources...! +

above to get started

IBM Tools Base Administration Console for z/OS View Configure

Resources

Search

IMSPlex View

CSDMEC20

PLEX1

HWS1

IMS1

Transactions

Programs

Databases

EC01663

PLEX1

IMS1

Transactions

Programs

Databases

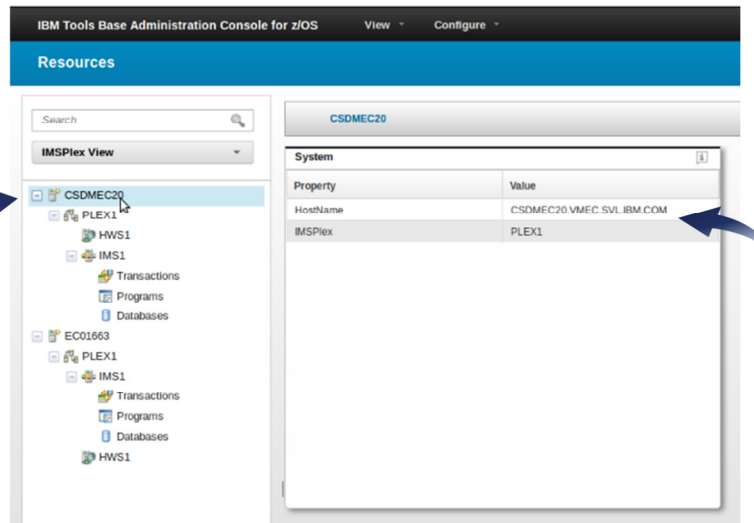
HWS1

62

03- IMS13 Integration: 62

Select from custom groups the IMSPlex view and notice on the right you will have two system nodes; here we have configured two LPARs to the two therefore see two system nodes.

Explorer for Administration – System View(s)



The screenshot displays the IBM Tools Base Administration Console for z/OS. The main window is titled "Resources" and shows a tree view of system nodes. The "CSDMEC20" node is selected, and its details are shown in the right pane. The details pane shows a table with the following data:

Property	Value
HostName	CSDMEC20.VMEC.SVL.IBM.COM
IMSplex	PLEX1

Clicking on the system nodes shows you the hostname and the IMSplex for the systems. If you have more than one IMSplex they will appear under the system node.

Explorer for Administration – IMSplex Resources

Member Name	Status	IMSplex	Member	Member Type	Member Sub Type	Job	Version	OS Image	Completion Code
OM1OM	✓	CSLPLEX1	OM1OM	OM		OM1	1.5.0	CSDMEC20	0
RM1RM	✓	CSLPLEX1	OM1OM	RM	SNGLRM	RM1	1.5.0	CSDMEC20	0
SCI1SC	✓	CSLPLEX1	OM1OM	SCI		SCI1	1.5.0	CSDMEC20	0
HWS1	✓	CSLPLEX1	OM1OM	IMSCON		HWS1	12.1.0	CSDMEC20	0
IMS1	✓	CSLPLEX1	OM1OM	IMS	DBDC	IMS1	12.1.0	CSDMEC20	0

In the navigation tree on the left you can see the IMSplex “PLEX1”, when you click on it the resources part of the IMSplex appear on the right. Notice the status indicated as green and the various members of the IMSplex (SCI, OM, ICON, IMS, RM) and their attributes such as resource versions.

Explorer for Administration – Manage Transactions

Select	Transaction Code	Status	Commit Mode	Conversational	Fast Path	Class	Limit Count	Message Queue Count	Global Queue Count	Member	PSB	PGM Status
<input type="checkbox"/>	ADDINV	❌	MULT	N	N	4	2	0	0	RMS1	DFSSAM04	✅
<input type="checkbox"/>	ADDPART	⚠️	MULT	N	N	4	2	0	0	RMS1	DFSSAM04	✅
<input type="checkbox"/>	AOBMP	✅	SNGL	N	N	23	65535	0	0	RMS1	TS21ACB0	✅
<input type="checkbox"/>	AOP	✅	SNGL	N	N	4	4	0	0	RMS1	TS1UAP0	✅
<input type="checkbox"/>	BHA1	✅	SNGL	N	N	1	65535	0	0	RMS1	PMAPJK13	✅
<input type="checkbox"/>	BHA2	✅	SNGL	N	N	1	65535	0	0	RMS1	PMAPJK23	✅
<input type="checkbox"/>	EMHTX2	❌	SNGL	N	E	1	0	0	0	RMS1	EMHPSB2	✅
<input type="checkbox"/>	EMHTX3	✅	SNGL	N	E	1	0	0	0	RMS1	EMHPSB2	✅
<input type="checkbox"/>	DEBSTRAN	✅	MULT	N	N	1	65535	0	0	RMS1	DEBS	✅

In the navigation tree if you expand it till you see your IMS, you can drill down further into the IMS's transactions, programs and databases. Here we are highlight in the tree the transactions.

Notice in the transactions main page you also see the related programs on the right. Hovering over the red, yellow or green icon will show you the status's of the transaction.

The default columns are predefined attributes but can be altered by clicking on the attributes button above the transaction rows and submitting your change. You can select any IMS Attributes that return from the QUERY Tran TYPE2 command.

Also notice you can multi select a group of transactions and start or stop them.

If you double click on any transaction row, you will be taken to the transaction details and relationship page where you will see more details about this transaction and its relationships.

Explorer for Administration – Start/Stop Transactions

CSDMEC20 > PLEX1 > IMS1 > Transactions

Select Attributes

Select	Transaction Code	Status	Commit Mode	Conversational	Fast Path	Class	Limit Count	Message Queue Count	Global Queue Count	Member	PSB	PGM Status
<input checked="" type="checkbox"/>	ADDINV		MULT	N	N	4	2	0	0	IMS1	DFSSAM04	
<input checked="" type="checkbox"/>	ADDPART						2	0	0	IMS1	DFSSAM04	
<input checked="" type="checkbox"/>	AOBMP						65535	0	0	IMS1	TS2IAOB0	
<input type="checkbox"/>	AOP						4	0	0	IMS1	TS1IAOP0	
<input type="checkbox"/>	BHA1						65535	0	0	IMS1	PMAPIK13	
<input type="checkbox"/>	BHA2						65535	0	0	IMS1	PMAPIK23	
<input type="checkbox"/>	EMHTX2						0	0	0	IMS1	EMHPSB2	
<input type="checkbox"/>	EMHTX3						0	0	0	IMS1	EMHPSB2	
<input type="checkbox"/>	DEBSTRAN		MULT	N	N	1	65535	0	0	IMS1	DEBS	

Start Transaction

Q

SCHD

SUSPEND

TRACE

Start/Stop multiple transactions.

Explorer for Administration – Transaction Details

IBM Tools Base Administration Console for z/OS View Configure admin IBM

Resources

Search

IMSplex View

- CSDMEC20
 - PLEX1
 - IMS1
 - Transactions
 - Programs
 - Databases
- EC01663
 - PLEX1
 - IMS1
 - Transactions
 - Programs
 - Databases

Transaction: EMHTX2

IMS Attribute	Value	IMS Attribute	Value	IMS Attribute	Value
Transaction Code	EMHTX2	PGM Name	EMHPSB2	Routing Code	EMHTX2
Status	STOQ STOSCHD	Status	✓	Status	✗
Comment Mode	SNOL	BMP Type	N	Program	EMHPSB2
Conversational	N	Definition Type	MOBLKS	Inquiry	N
Fast Path	E	Dynamic Option	N	Definition Type	MOBLKS
Class	1	Fast Path	E	Region	
Limit Count	0	Language Interface		Last Access Time	
Message Queue Count	0	Member	IMS1	Time Created	2012.301.15.13.49.22
Global Queue Count		Region type	PP	Last Import Time	
Member	IMS1	Local Scheduling Type	PARALLEL	Last Update Time	
Attny		Last Time Accessed		Model Type	
ACI Command Support	N	CC Text		Model Name	
APPC LU Name		Completion Code	0	Completion Code	0
CC Text		Generated PSB	N	Member	IMS1

Related Databases

Database Name	Database Type	Status	Access Type	Area Name	Definition Type	Member	Part Name	Last Access Time	Completion Code
MSDBLM01	MSNR	✓	EXCL		MOBLKS	IMS1			0
MSDBLM02	MSNR	✓	EXCL		MOBLKS	IMS1			0
MSDBLM03	MSNR	✓	EXCL		MOBLKS	IMS1			0
MSDBLM04	MSNR	✓	EXCL		MOBLKS	IMS1			0
MSDBLM05	MSRF	✓	EXCL		MOBLKS	IMS1			0
MSDBLM06	MSRD	✓	EXCL		MOBLKS	IMS1			0
MSDBLM07	MSRD	✓	EXCL		MOBLKS	IMS1			0
MSDBLM08	MSNR	✓	EXCL		MOBLKS	IMS1			0
DEDLNZ1	DEDB	✓	UPD		MOBLKS	IMS1			0
INDR0101	ASCL	✓		INDR0101		IMS1			0

Finished retrieving child resources

Transaction details view. Here you can drill into the transaction from the main page by double clicking the transaction row EMHTX2 and you see the transactions details and its related programs, routing codes and databases.

Notice that there is context sensitive help on this page with regard to each of the panels, see later slides for examples.

Explorer for Administration – Hover Help

IMS Attribute	Value
Transaction Code	EMHTX2
Status	STOQ, STOSCHD
Commit Mode	SNGL
Class	1
Limit Count	0
Message Queue Count	0
Global Queue Count	
Member	IMS1
Affinity	
AOI Command Support	N
APPC LU Name	
CC Text	

Transaction: EMHTX2

Scheduling class used to determine which message regions can process the transaction

Hover help is available to help assist you with unfamiliar attributes. Clicking the “i” in the top right transaction panel would also display a help panel with more details for transaction attribute explanations. Notice that the hover help and the “i” help will only be related to the panel you are in to reduce the need to search numerous pages from the IMS book.

Explorer for Administration – Context Sensitive Help

CSDMEC20 > PLEX1 > IMS1 > Transactions > EMHTX2

Transaction: EMHTX2		Related Program		Related Routing Code	
IMS Attribute	Value	IMS Attribute	Value	IMS Attribute	Value
Transaction Code	EMHTX2	POM Name	EMHPSB2	Routing Code	EMHTX2
Status	STOQ, STOSCHD	Status	✓	Status	✗
Commit Mode	SN	Definition Type	N	Program	EMHPSB2
Conversational	N	Dynamic Option	MOOBLKS	Inquiry	N
Fast Path	E	Fast Path	N	Definition Type	MOOBLKS
Class	1	Language Interface	E	Region	
Limit Count	0	Member	IMS1	Last Access Time	
Message Queue Count	0	Region type	IFP	Time Created	2012.301.15.13.49.22
Global Queue Count	N	Local Scheduled Type	PARALLEL	Last Import Time	
Member	IMS1	Last Time Accessed		Last Update Time	
Affinity		CC Text		Model Type	
AOI Command Support	N	Completion Code	0	Model Name	
APPC LU Name		Generated PSB	N	Completion Code	0
CC Text		Member	IMS1	Member	IMS1

Click to see status help

Help

Transaction status: Unavailable

The selected transaction is unavailable. This condition has multiple causes, including a stopped transaction, an uninitialized transaction, or a serious error that resulted in an applicationabend.

The status code for the transaction provides more details about why the transaction is unavailable. This information is retrieved from the LcStat field of the **QUERY TRAN** command.

- NOTINIT status code**
The NOTINIT status code indicates that the transaction is not initialized.
- STOQ status code**
The STOQ status code indicates that the transaction is stopped for queuing and can no longer be queued globally.
- STOSCHD status code**
The STOSCHD status code indicates that the transaction is stopped for scheduling and cannot be scheduled globally.
- QERR status code**
The QERR status code indicates that an I/O error occurred.
- LISTO status code**
The LISTO status code indicates that the transaction is stopped for scheduling because of unavailable data.

Related information:

Database Name	Database Type	Status	Access Type	Area Name	Definition Type	Member	Part Name	Last Access Time	Completion Code
---------------	---------------	--------	-------------	-----------	-----------------	--------	-----------	------------------	-----------------

69 03- IMS13 Integration: 69

Notice next to the red icon the status codes shown indicating why its a red color. If you want to know more about these particular status's or why its marked red click on the icon and on the right side of the browser a help panel appears with related help to this transactions status. Click on one of the transaction status links in the help panel and drill further to see corrective actions.

Explorer for Administration – Programs View

Search

IMSPlex View

- [-] CSDMEC20
 - [-] PLEX1
 - [-] HWS1
 - [-] IMS1
 - [-] Transactions
 - [-] Programs
 - [-] Databases
- [-] ECO1663
 - [-] PLEX1
 - [-] IMS1
 - [-] Transactions
 - [-] Programs
 - [-] Databases
 - [-] HWS1

CSDMEC20 > PLEX1 > IMS1 > Programs

PGM Name	Status	BMP Type	Definition Type	Dynamic Option	Fast Path	Language Interface	Member	Region type	Local Scheduled Type	Last Time Accessed	Generated PSB
AD2CONV	✓	N	MOBLSKS	N	N		IMS1	MPP	SERIAL		N
AD2TP	✓	N	MOBLSKS	N	N		IMS1	MPP	SERIAL		N
APOL1	✓	N	MOBLSKS	N	N		IMS1	MPP	SERIAL		N
AUTOGSAM	✓	Y	MOBLSKS	N	N		IMS1	JBP	SERIAL		N
AUTPSB1	✓	Y	MOBLSKS	N	N		IMS1	BMP	SERIAL		N
AUTPSB1H	✗	N	MOBLSKS	N	N		IMS1	MPP	PARALLEL		N
AUTPSB1I	✗	N	MOBLSKS	N	N		IMS1	MPP	PARALLEL		N
AUTPSB11	✓	N	MOBLSKS	N	N		IMS1	JMP	PARALLEL		N
AUTPSB2	✓	Y	MOBLSKS	N	N		IMS1	BMP	SERIAL		N

Similar to transactions there is the ability to display information and status's about programs with the double click function that will take you into a details and relationship view.

Explorer for Administration – Databases View

Search

IMSPlex View

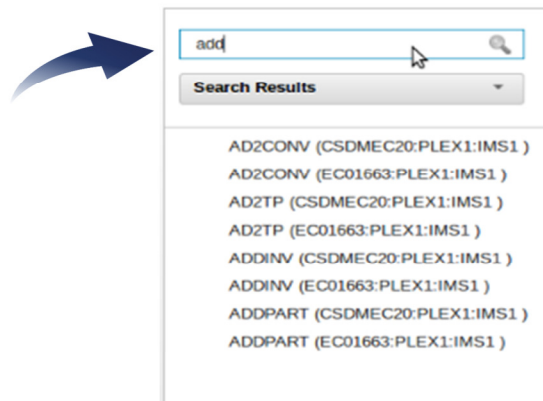
- [-] CSDMEC20
 - [-] PLEX1
 - [-] HWS1
 - [-] IMS1
 - [-] Transactions
 - [-] Programs
 - [-] Databases
- [-] EC01663
 - [-] PLEX1
 - [-] IMS1
 - [-] Transactions
 - [-] Programs
 - [-] Databases
 - [-] HWS1

CSDMEC20 > PLEX1 > IMS1 > Databases

Database Name	Database Type	Status	Access Type	Area Name	Definition Type	Member	Part Name	Last Access Time	Completion Code
AUTOOB	DLI	✓	UPD		MODBLKS	IMS1			0
AUTOOBH		⚠	UPD		MODBLKS	IMS1			0
BANKATMS		⚠	EXCL		MODBLKS	IMS1			0
BANKFNCL		⚠	EXCL		MODBLKS	IMS1			0
BANKLDGR		⚠	EXCL		MODBLKS	IMS1			0
BANKTERM		⚠	EXCL		MODBLKS	IMS1			0
BE2PCUST	DLI	✓	EXCL		MODBLKS	IMS1			0
BE3ORDER	DLI	✓	EXCL		MODBLKS	IMS1			0
BE3ORDRX	DLI	✓	EXCL		MODBLKS	IMS1			0

Similar to transactions there is the ability to display information and status's about databases with the double click function that will take you into a details and relationship view.

Explorer for Administration – Search



If you know the name of the transaction, program or database you can search for it in the top left corner of the browser where there is a text field. Even if you don't know exactly the name, you can search character by character and the search will narrow down eventually to the name of the resource you are searching for. Clicking on it will take you to the respective details view for that resource. IE if it's a transaction you searched on, you will see the transaction details for that transaction and related programs and databases.

System Prerequisites

- **Software requirements**
 - IMS Tools Admin Console (no-charge)
 - IMS Version 12 or later
 - IMS Connect
 - Common Service Layer OM and SCI
 - Supported web browser
 - Firefox, Internet Explorer, Safari