Evolution of IMS to a complete Enterprise solution

Abstract:

IBM Information Management System (IMS) has catered over the past 4 decades to the evolving IT needs in Mainframe world. For many thousands of companies around the world, including over 95 percent of Fortune 1000 companies, that core data comes from IBM System z mainframe computers running the IBM Information Management System (source – IBM Knowledge Center).

The reliability and transparency of IMS can make it nearly invisible to enterprise architects. As a result, as companies build out new architecture, they may not be using IMS to its fullest advantage. They may not be aware that they can integrate their existing IMS data and transactions into the enterprise architecture, often without any required changes, to grow the business, reduce costs and meet or beat their IT objectives.

In the modern IT landscape, the product service offerings are extended beyond its original scope and boundaries. In the Enterprise architecture, there are new applications developed outside Mainframes which has rich user interface and there is a need for integrating these new applications with IMS where the core business logic and data resides, thereby gaining the best of both worlds.

The objective of this white paper is to explain the key integration and access points which enable IMS assets to provide high value as part of a distributed, multitier architecture. It also focuses on the ways that organizations can integrate IMS assets across the enterprise to save time and money and achieve optimum results—without having to re-architect solutions.

IMS Products:

Considering IMS assets, we must distinguish two areas of IMS, namely Transaction Manager (IMSDC or IMS TM) and IMS Database Manager (IMSDB):

IMS Transaction Manager (TM): This is a message-based manager for online transaction processing designed to manage queuing, security, scheduling, formatting, logging and recovery of input and output messages.

IMS Database Manager (IMSDB) or Data Base Controller (DBCTL): This is a database management system (DBMS) for defining database structure, organizing business data, performing queries against the data and performing database transactions.

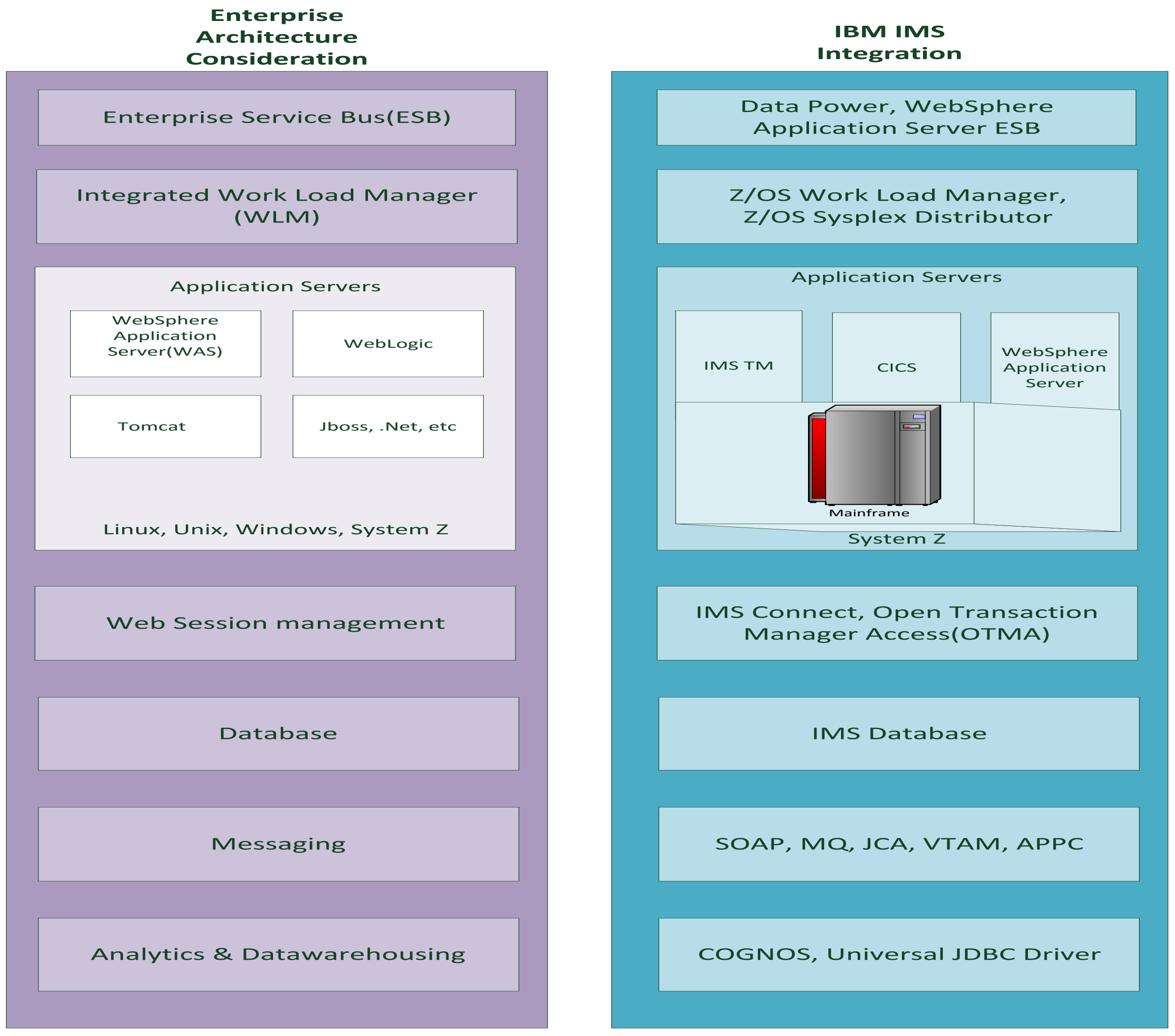
Enterprise Architecture Consideration in IMS:

In a multi-tier architecture, Enterprises must address several key considerations when optimizing their IT architectures. These considerations include architectural elements such as:

- The enterprise service bus (ESB)
- Workload management (WLM) and routing
- Application containers
- Web session management
- Database environment
- Messaging and business intelligence tools

All of which must be integrated so that data can be delivered where, when and how it is needed throughout the organization.

IBM provides corresponding tools, capabilities, integration points and other solution elements in each of these areas, enabling implementation and integration of IMS as part of a multi-tier enterprise architecture.


Enterprise Architecture Consideration
Enterprise Service Bus(ESB)
Integrated Work Load Manager (WLM)
Application Servers
WebSphere Application Server(WAS)
WebLogic
Tomcat
Jboss, .Net, etc
Linux, Unix, Windows, System Z
Web Session management
Database
Messaging
Analytics & Datawarehousing
IBM IMS Integration
Data Power, WebSphere Application Server ESB
Z/OS Work Load Manager, Z/OS Sysplex Distributor
Application Servers
IMS TM
CICS
WebSphere Application Server
Mainframe
System Z
IMS Connect, Open Transaction Manager Access(OTMA)
IMS Database
SOAP, MQ, JCA, VTAM, APPC
COGNOS, Universal JDBC Driver

The following section discusses the various building blocks in IMS which enables the Enterprise architectural considerations that are handled in an IMS implementation.

Application containers:

IMS TM provides many functions within the overall enterprise architecture, including an application container function. This function is produced by IMS through "dependent regions." Each dependent region virtualizes resources for system services, application logic, database calls, message handling, communication, memory and fault tolerance.

When it becomes necessary to perform maintenance on IMS applications to modernize them with new application languages like Java, or to adopt service-oriented architecture (SOA) in the enterprise, IBM Rational Developer for z/OS provides an integrated development environment for making these changes. This highly efficient platform for IMS application development offers a way to modernize with SOA principles and integrate with IMS adapters, produces IMS web services via SOAP and XML plug-ins, incorporates JDBC drivers and simplifies maintenance of IMS applications. The Rational Asset Analyzer(RAA) helps in assessing the current code base to find dead code or redundancies and determine the impact of code changes.

Implementing IMS connectivity solutions:

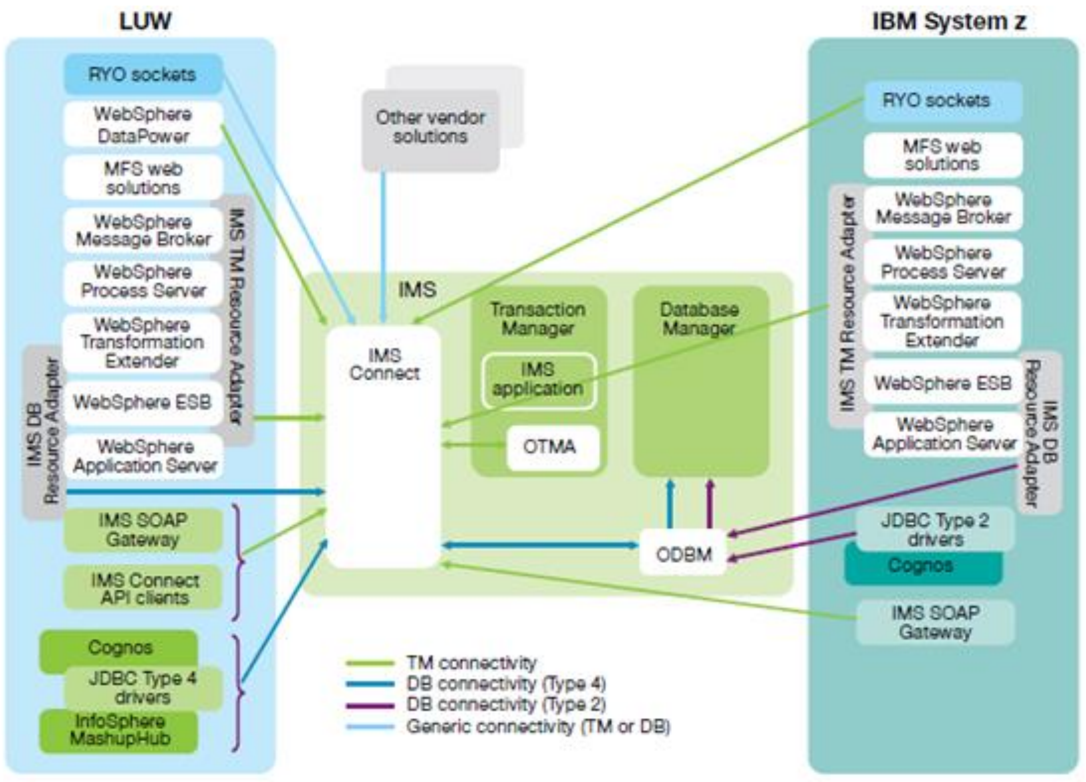
Connectivity solutions for IMS include IMS Connect, enterprise-wide messaging capabilities and IMS Resource Adapters.

IMS Connect

IMS Connect is an integrated feature of IMS that provides high-performance communications for IMS between one or more TCP/IP or local z/OS clients and one or more IMS systems. Most contemporary IMS implementations involve TCP/IP traffic from the Internet, which is delivered to IMS through IMS Connect. As the TCP/IP communicator, IMS Connect opens IMS systems to servers performing distributed processing throughout the enterprise. IMS Connect supports both IMS DB and IMS TM systems.

IMS Connect functions as a standard socket server open to a variety of vendor solutions and works with IMS Open Database Manager (ODBM) to provide standards-based access to IMS data from TCP/IP-enabled platforms. Routing to IMS DB takes different paths depending on the operating system. For Linux, UNIX and Microsoft Windows (LUW), connections come in through IMS Connect to the ODBM and onward to IMS DB. For System z, connections come in through ODBM and then to IMS DB with no IMS Connect involvement.

More than one IMS Connect can be connected to a single IMS image. This is often done to prevent a single point of failure and to add more transaction throughput.



Web session management capabilities built into IMS enable reliable connections from a company's website to an application server and into an IMS data store. IMS Connect receives incoming TCP/IP transactions and communicates them to IMS through the Open Transaction Manager Access (OTMA), a portal in the IMS software. Through IMS Connect and OTMA, IMS can communicate with a wide variety of Java EE-compliant application servers—including

IBM WebSphere as well as other vendors solutions such as Wave, WebLogic, ComCast and JBoss. IMS receives and processes the transactions, generates result and then sends them back to OTMA to give to IMS Connect.

Enterprise-wide messaging capabilities

IMS supports a wide variety of industry standards for messaging protocols, including Virtual Telecommunications Access Method (VTAM), IBM WebSphere MQ, APPC, SOAP, J2C Connector Architecture (JCA) and Representational State Transfer (REST). All but the first three use IMS Connect.

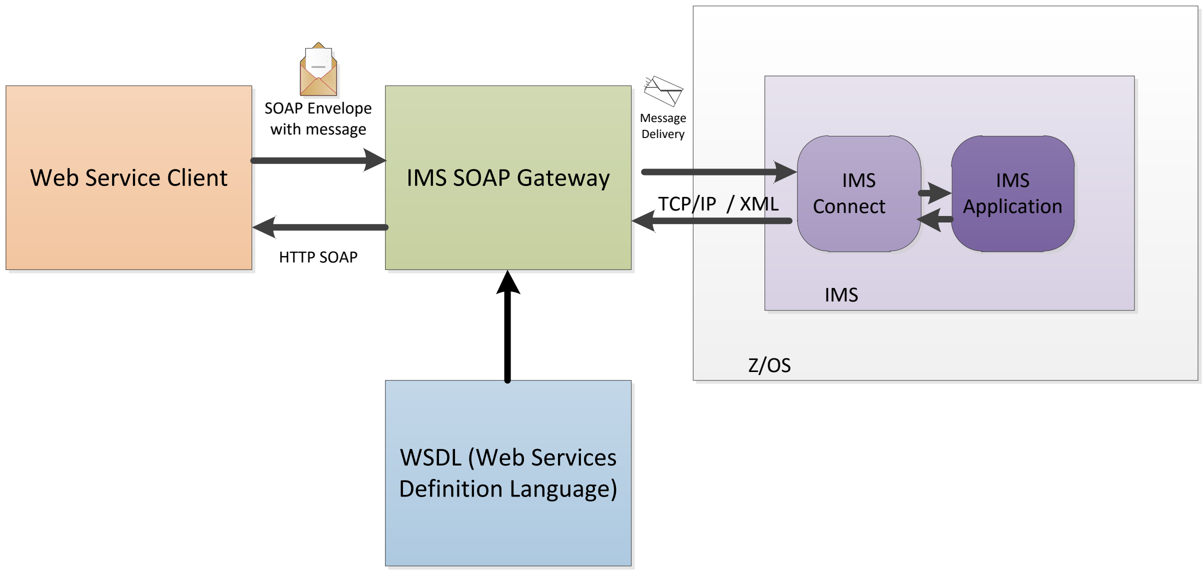
This broad support means that an organization's core mainframe systems can communicate across a multi-tier architecture without the use of proprietary techniques. Enterprise architects have the flexibility to integrate IMS with virtually any other system in the organization. IMS on System z provides an open platform for supporting anything from Web 2.0 agile development environments with RESTful interfaces to enterprise-class middleware. In addition to the messaging protocols, IMS supports a wide variety of message-passing paradigms including synchronous, asynchronous, two-phase, global and local methods.

IMS SOAP Gateway

IBM IMS SOAP Gateway is a Web services solution that enables IMS applications to interoperate outside of the IMS environment through Simple Object Access Protocol (SOAP) to provide and request services independently of platform, environment, application language, or programming model.

The IMS SOAP Gateway enables IMS systems to interact with Web servers using SOAP messages in XML format. The gateway receives these messages from web services, strips off the SOAP headers and passes the communications to IMS Connect. The IMS Connect solution then removes the XML wrapping before sending the messages to IMS TM or IMS DB for processing. The same communication steps run in reverse to send the results back to the web server.

The WebSphere MQ technique enables a system to drop a message into a queue where it can be picked up and processed by a receiving system, and IMS includes an IMS-MQ bridge for this purpose. JCA is supported as part of the communication architecture used by the IMS transaction and database resource adapters. IMS also supports VTAM, a traditional proprietary method of communicating with mainframe systems.



Resource adapters

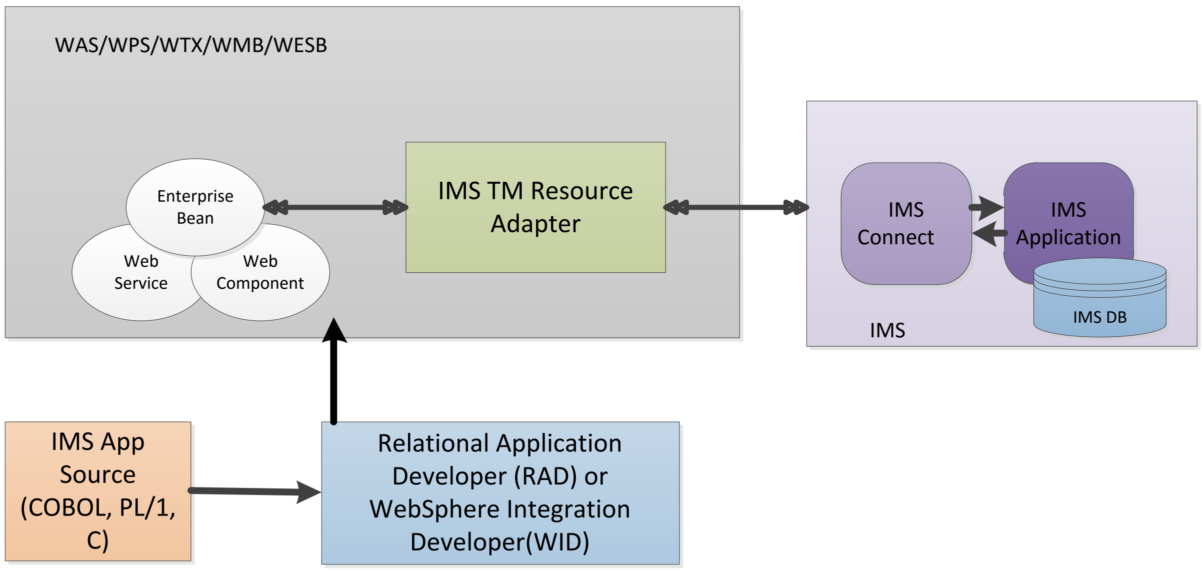
To enable application servers to communicate with IMS mainframe systems, IMS provides a set of resource adapters designed to run on any Java EE-compliant application server. The adapters use the JCA protocol. Because IMS consists of two parts—the IMS Transaction Manager (IMS TM) that runs transactions and the IMS DB that manages data—two resource adapters are required: one for TM and one for DB.

IMS TM resource adapter is used to create Java Enterprise Edition (J2EE) applications to access IMS transactions over the Internet, as well as to make callout requests to external Java EE applications from IMS applications that run in IMS dependent regions.

Using this resource adapter within a WebSphere or Rational development environment, we can:

- Develop components of business processes in support of service-oriented architecture
- Create Java EE applications from Java beans
- Develop service-based applications

The applications can then be deployed on application servers such as WebSphere Application Server, WebSphere Process Server, WebSphere Transformation Extender, or WebSphere Message Broker.



The DB Resource Adapter allows users to send in requests to access and bring back IMS data—for example, an application that is outside of IMS might need to retrieve the customer records for a particular individual. The TM Resource Adapter enables IMS to perform complete transactions such as a checking account withdrawal via ATM, including communication back to the ATM to dispense cash. In both cases, there is no need to change the existing transaction. The organization can keep the data in IMS and integrate it using IMS tooling.

Enterprise Service Bus Integration

An ESB is a mainstay of many large enterprises. ESBs enable organizations to simplify complex environments by integrating applications and data as services and communicating with all of the services in the same efficient way.

For IMS implementations, IBM WebSphere Enterprise Service Bus (WebSphere ESB) provides flexible mediation and hosting of these services. Built on top of WebSphere Application Server, WebSphere ESB helps organizations create services from existing assets and connect service providers with service consumers to create new business systems. It can connect virtually any business application, and supports common connectivity patterns such as simple proxy, service gateway, service translation and service selection.

In addition to WebSphere ESB, a variety of other enterprise services are available through IMS in the form of Java EE-compliant application servers designed for specific tasks. For example, WebSphere Message Broker is a lightweight, advanced ESB that enables the integration of data sources from a wide range of platforms. It can hold up to 40 different protocols, enabling distribution of traffic across those protocols to another service or platform.

Because IMS is fully integrated with WebSphere ESB and other WebSphere applications, it is easy to make IMS an integral part of an SOA and SLA strategy. IMS is critical to these efforts because much of an organization's basic customer and policy data is often located in IMS. With WebSphere ESB, it is easier to go anywhere in the enterprise to find data and use it to achieve business goals.

IMS dependent regions and Java

IMS provides address spaces to execute system and application programs that use IMS services. These address spaces are called "Dependent Regions". The dependent regions are started by the submission of JCL to the operating system. After the dependent regions are started, the application programs are scheduled and dispatched by the IMS control region. In all cases, the z/OS address space executes an IMS control region program. The application program is then loaded and called by the IMS code. Up to 999 dependent regions can be connected to one IMS control region.

Dependent regions can be either message driven (message processing) regions or non-message driven regions (batch). Message processing regions (MPRs) run applications that process messages that come into IMS TM as input, such as from terminals or online programs. When the IMS determines that an application is to run in a particular MPR, the application program is loaded into that region and receives control. The application processes the message and any further messages for that transaction that are waiting to be processed. Then, depending on options specified on the transaction definition, either the application waits for further input, or another application program is loaded to process a different transaction.

IMS provides the following dependent region types:

- Message processing regions, which read and process messages from the IMS message queue, including the following:
  - Message Processing Program (MPP)
  - IMS Fast Path (IFP)
  - Java Message Processing (JMP)

- Non-message driven regions, which are batch-oriented and do not process IMS messages:
  - Batch Message Processing (BMP)
  - Java Batch Processing (JBP)

JMP regions and JBP regions enable Java, object-oriented COBOL, object-oriented PL/I, or a combination of these languages to get control when an IMS transaction is scheduled. This differs from IFP regions and BMP regions, which allow programs written in Assembler, PL/I, C, COBOL, and PASCAL to get control when an IMS transaction is scheduled.

IMS dependent regions support persistent Java Virtual Machines (JVM) to provide the capability to efficiently make calls that interoperate between Java and languages such as COBOL and PL/I. This enables the capability to leverage existing IMS applications as well as extend functionality using Java. Interoperability is supported in MPP, BMP, IFP, JMP, and JBP regions.

Using Java to develop new business logic enables to leverage the skills that are common in the distributed world environment, such as JDBC. In addition, it lowers the cost of running Java applications by taking advantage of System z Application Assist Processor (zAAP) engines, which assist in lowering the chargeable MIPs usage when processing Java.

JBP regions run flexible programs that perform batch-type processing online and can access the IMS message queues for output (similar to non-message-driven BMP applications). JBP applications are started by submitting a job with JCL or from TSO. JBP applications are like BMP applications, except that they cannot read input messages from the IMS message queue. Like BMP applications, JBP applications can use symbolic checkpoint and restart calls to restart the application after an abend.

IMS Tools

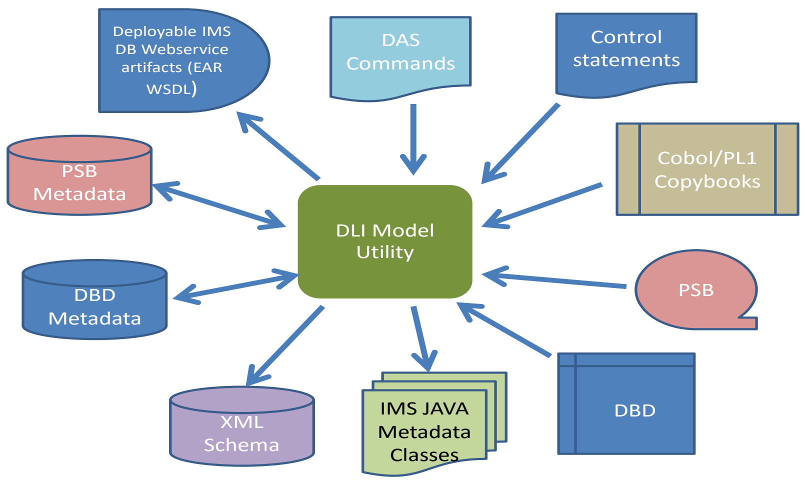
Modernization in IMS Tools has made relational IMS data easier to work with even without the traditional PL/1 or COBOL programming knowledge. This enhancement in IMS tools has bridged the gap in working with relational data and hierarchical data. The major tools that we will discuss are:

- DLI Model utility – Enables IMS data structures viewable in a relational format in GUI interface
- IMS catalog Database - IMS meta data repository
- IMS Explorer utility - Ability to write industry standard SQL queries to access IMS data

DLI Model Utility:

The DLI Model Utility is a plug-in on top of Eclipse. Once the eclipse plug in is download it let us visualize IMS PSBs and DBDs and generate metadata for XML DB support. DL/I Model Utility plug in provides a user friendly interface, simplifies IMS metadata generation, eases IMS Java and XML database application development and access, and offers a visual representation of IMS databases. Users can visualize an entire IMS PSB and DBD in a multipage graphical editor.

It has been used to generate the necessary metadata that is consumed at runtime by the IMS Universal driver, XML-DB, XQuery and IMS DB Web services. This tooling currently uses a bottom-up approach, parsing PSB and DBD source using either Control statements or Wizard panels. User can optionally import COBOL copybook and PL/I Include definitions to define field layouts for each segment.



The drawback when using the DLIMODEL utility is that it can be a challenge to manage. Also, change control is needed when the underlying database definition changes. This method potentially allows the creation of multiple copies of the database metadata, each one of which must be updated with any database structure change.

IMS Catalog Database:

There is a growing need for the IMS metadata that should be easier to manage and can be trusted to reflect the possible changes of the design of IMS databases. This gave way to the IMS catalog Database utility. The IMS catalog holds the metadata for databases and PSBs. It is accessed by using the Java Database Connectivity (JDBC) drivers and is also available to any tool or application. IMS metadata is available in an Extensible Markup Language (XML) format, and also available to standard DL/I applications in a traditional IMS segment format.

When the metadata is updated, the catalog is also updated to reflect the change; therefore, the data is always current. The data is current because the consumers get metadata dynamically from the active IMS catalog High Availability Large Database(HALDB), rather than from static class files. Enhancements to the existing DBDGEN, PSBGEN, and ACBGEN processes allow database and application metadata to be defined to IMS. The IMS catalog is the single, authoritative source of database and application metadata for all client applications.

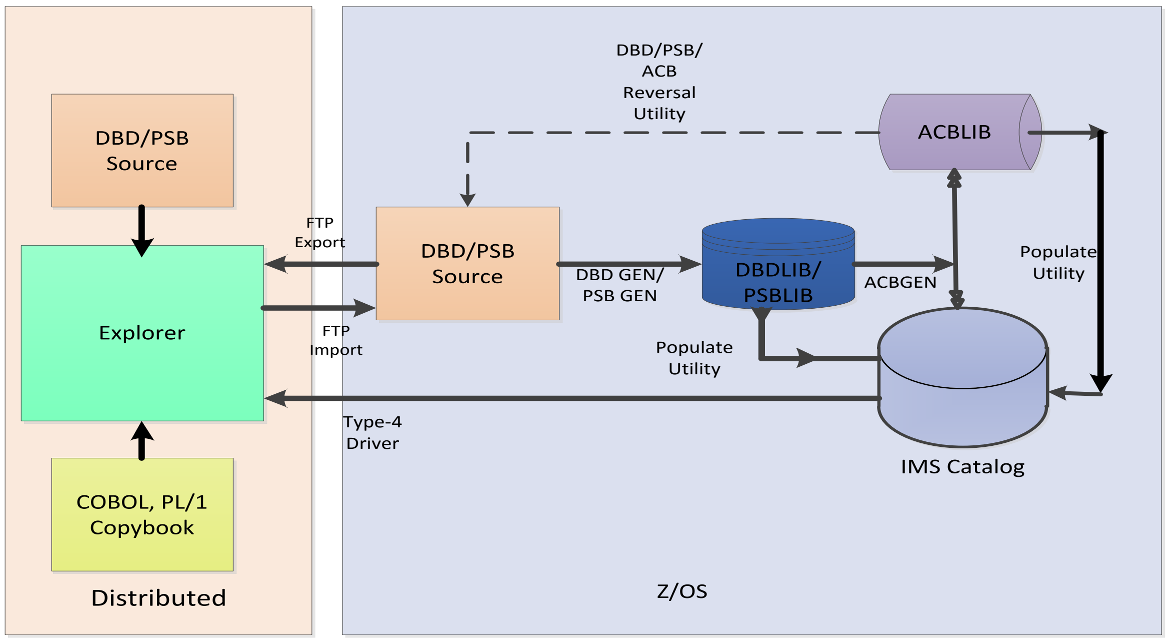
IMS Enterprise Suite Explorer:

The IMS Enterprise Suite Explorer (IMS Explorer) for Development is an Eclipse-based graphical tool that simplifies IMS application development tasks. The tool simplifies tasks such as updating IMS database and program definitions by using standard Structured Query Language (SQL) to manipulate the IMS data.

IMS Explorer graphical editors are used to import, visualize, and edit IMS database and program definitions. IMS Explorer can also be used to easily access and manipulate data that is stored in IMS, by using standard SQL.

The IMS Explorer can also directly import DBDs and PSBs, or can obtain existing catalog information from the IMS catalog via a type 4 connection. The population of the IMS catalog is mainly done from the PSB and DBD sources. The DBD source doesn’t hold all the IMS database segment definitions apart from the Search segments and indexed fields. Additional DBD statements and additional attributes on the FIELD macro can provide more complete metadata information.

Additional metadata can be collected from COBOL copybooks and PL/I or C include members. After updating the metadata information in the PSBs and DBDs in the Explorer, those sources can be sent to the host for the GEN process. This process causes the metadata to be populated into the IMS catalog.



The above figure shows how the Explorer is able to interact with the IMS catalog directly, via a type 4 driver connection. It also shows an interaction with the catalog indirectly, via a File Transfer Protocol (FTP) import and export.

A direct update from the explorer is not available. An intermediate application control block generation (ACBGEN) with population of the catalog is required.

The DLIModel utility is a helpful tool for visualizing. The tool documents all details about the DBDs and PSBs sources. The DLIModel utility is required to produce the com.ibm.ims.db.DLIDatabaseView class for accessing the IMS databases from Java applications on the remote and local locations. This access is through the type 4 and type 2 drivers.

The DLIModel utility is not changed in Enterprise Suite, but its functions are integrated into the IMS Enterprise Suite Explorer. The IMS Explorer uses the centralized IMS catalog on IBM z/OS. The metadata that is currently available in the DatabaseView class is integrated in the IMS catalog along with many more details. The concept of the IMS catalog makes the metadata more dynamic and shared. Its implementation avoids the need for distributing the class to all sites where it might be used in Java programs with DL/I (also via JDBC) access. The access is through the IMS database type 2 and type 4 Universal drivers. The drivers are adapted to integrate this dynamism.

The following features summarize the IBM Enterprise Suite Explorer:

- Incorporates DLIModel functionality Support for migration of DLIModel projects is provided
- Provides the following graphical editors for development and visualization:
  - Program Specification Block (PSB)
  - Database Description (DBD)
- Shows a relational view of IMS data with graphical assistance to build SQL statements

Summary:

IMS delivers high value as a premier online transaction and database management server, with architected interfaces that open up exciting new avenues for leveraging IMS as an integral part of the enterprise in the ever-evolving business world. As a highly scalable platform, IMS on System z can also play an important role in making cloud deployments successful.

Today's enterprises gain competitive advantage by quickly developing and deploying applications that provide unique business services. Whether they are internal applications or web-based customer and vendor services, quick integration and deployment are keys to success. IMS support for complementary standards surrounding IMS connectivity, data representation and application development helps organizations to respond to market changes as quickly and cost-effectively as possible.

Reference:

- A practical guide to IMS Connectivity
- IBM Redpaper on IMS Catalog
- IBM Developer Works – Use Java in IMS