

Opening doors to access IMS/DB via Db2

Stan Hoey
Syncsort Inc

November 2019
Session HK



Agenda

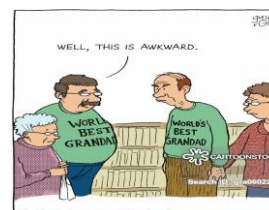
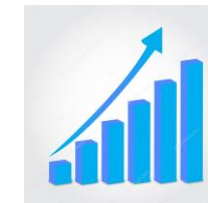
- Introduction
- History
- Options to stay in IMS/DB
- Options to migrate from IMS/DB
- Free beer

Me

- 1974-1996 – IBM UK
 - 1974 – 1986 Operations, Systems Programming
 - 1986 – 1996 Service Management, Db2 Systems programming/SYSADM

- 1996-2013 – Circle Computer Group
 - Db2 training
 - DL/2 Product Manager

- 2013-today – Syncsort Inc
 - DL/2 Product Manager
 - EMEA Customer Support



History lesson, courtesy of Wikipedia

- [IBM](#) designed the IMS with [Rockwell](#) and [Caterpillar](#) starting in 1966 for the [Apollo program](#), where it was used to inventory the very large [bill of materials](#) (BOM) for the [Saturn V](#) moon rocket and Apollo space vehicle.
- The first "IMS READY" message appeared on an [IBM 2740](#) terminal in [Downey, California](#), on August 14, 1968. (18,711 days ago)
- IMS/TM is one of 3 popular transaction managers (with CICS and Tuxedo)
- The chances are that withdrawing money from an [automated teller machine](#) (ATM) anywhere in the world triggers an IMS transaction.
- In 2013 IBM completed a benchmark on IMS Version 13 demonstrating the ability to process 100,000 transactions per second on a single IMS system

There are many reasons to stay with IMS

- RAS (Reliability, Availability, Serviceability)
 - Very mature product
 - 100s of years development effort, new features with every new version
- Scalability
 - Unmatched transaction volumes and throughput
 - Queue and Data sharing, HALDB
 - Performance, FastPath
- Accessibility
 - Java, SQL access, Data sharing
- Tooling
 - IBM and OEMs provide a wide range of monitoring, DBA and AD tools
- Core applications have been running successfully in IMS for decades

Customer feedback from IBM

“The mix of Java and COBOL in the IMS environment is a good way to modernize the core applications on the mainframe.”

Thomas Bauer, Computer Scientist
Fiducia & GAD IT AG



So why look at migrating IMS/DB to Db2?

#1 reason: Cost

- MLC pricing model can make IMS very expensive
 - Few applications are exclusively IMS/DB so difficult to isolate in a dedicated LPAR
 - Small IMS application footprint may have to run on a very large processor
- Add in the cost of tools and it becomes even more expensive
- IMS is not seen as strategic and legacy mainframe environments often miss out on investment
- New IMS versions are usually more expensive and become harder to justify if there is no requirement to exploit new features
- IMS systems may have resulted from an acquisition and don't fit in existing operational architecture

So why look at migrating IMS/DB to Db2?

#2 reason: Skills

- Workforce cost reductions often fall on older and more expensive employees
- Many IMS applications are > 30 years old and original developers are retired (or worse)
- Applications are stable and there are limited opportunities to deep dive into code
- New IMS applications are uncommon in many sites and there's little need to keep skills current
- Scarcity of next generation developers
- Many applications are written in COBOL, PL/I or even assembler and skills not maintained
- There is often fear in changing IMS applications
- Some source components may no longer be available

Whether or not we agree, it's a fact so what are the options?

SQL access to IMS is possible today

From existing IMS documentation

- Use the IMS Universal JDBC driver to connect to an IMS database for writing SQL queries
- Meta-data from the IMS Catalog or the IMS Enterprise Suite Explorer for Development
- Rich set of SQL functions is available
- Some DDL and DML usage rules are imposed

```
CREATE DATABASE SQLDBAC ACCESS HDAM VSAM LIKE SQLDBOR CCSID 'UTF-8'
```

Table 1. Mapping between IMS database elements and relational database elements.

Hierarchical database elements in IMS	Equivalent relational database elements
Segment name	Table name
Segment instance	Table row
Segment field name	Column name
Segment unique key	Table primary key
Foreign key field	Table foreign key

Is rewriting an option?

- For some IMS applications, yes:
 - z/Journal article in 2009:

Type of Element	Elements by Complexity Level				
	Simple	Low	Medium	High	Total
Batch Programs (IMS)		60	68	24	152
Batch Programs (non-IMS)		13	35		48
Online Programs (IMS)		12	36	14	62
Online Programs (non-IMS)		5			15
JCL (unique jobs)	550				550
Elements by complexity	550	90	149	38	827
Conversion Hours (each)	8	24	40	160	
Hours by complexity	4,400	2,160	5,960	6,080	
TOTAL HOURS FOR CONVERSION					18,600

- Programs to be converted

- Conversion effort based on program complexity

Straightforward if sufficient resources, but complex databases or applications are more challenging

The practicalities of rewriting

- A degree of application re-architecture may be required, converting DLI calls to SQL is seldom 1:1
 - Path calls
 - Accessing concatenated segments
- Detailed DLI knowledge is required
 - Common command codes (for example F, L, V, N)
 - Original programming standards (or potentially, lack of same) can be challenging
 - Many changes or temporary fixes may have been made
- Database error handling needs to be recoded
- How will tables databases be handled?
- Some source objects may be unreliable or lost

Lack of detailed DLI knowledge is probably why the rewrite is happening!



Data migration challenges

- IMS database attributes
 - Concatenated key
 - Sparse indexes
 - Variable length segments
 - Logical relationship update rules
 - Logical children variable intersection data
- Data migration
 - Numeric data fields(packed and zoned decimal, signed and unsigned)
 - Date fields
 - Nulls/un-initialised data
 - REDEFINES, OCCURS
 - *“I’ve never seen a character field I didn’t like”*

Big Bang production cutover can be risky

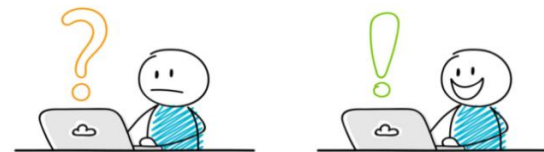


Can tooling help?

- To some extent
 - Simplifying test environments, although not appropriate if applications are static
 - Managing complex production environments
 - Automated database maintenance
 - Automated recovery
 - Easier to manage data issues/SOC7 abends

- IMS Explorer
- IMS Catalog
- IMS Management Console

Helps to mitigate diminishing IMS skills



What about replication?

- This is a good option for some
 - Negligible disruption to current IMS applications
 - Potential workload balancing
 - Reduce batch window
 - Non-disruptive application maintenance
 - Can provide with disaster recovery capability with no data loss
 - Targets can be IMS or other DBMSs
- But not without drawbacks
 - Environment is more complex
 - Source data is still IMS

Can solve some problems but doesn't address IMS skills paucity

Are there automated conversion tools?

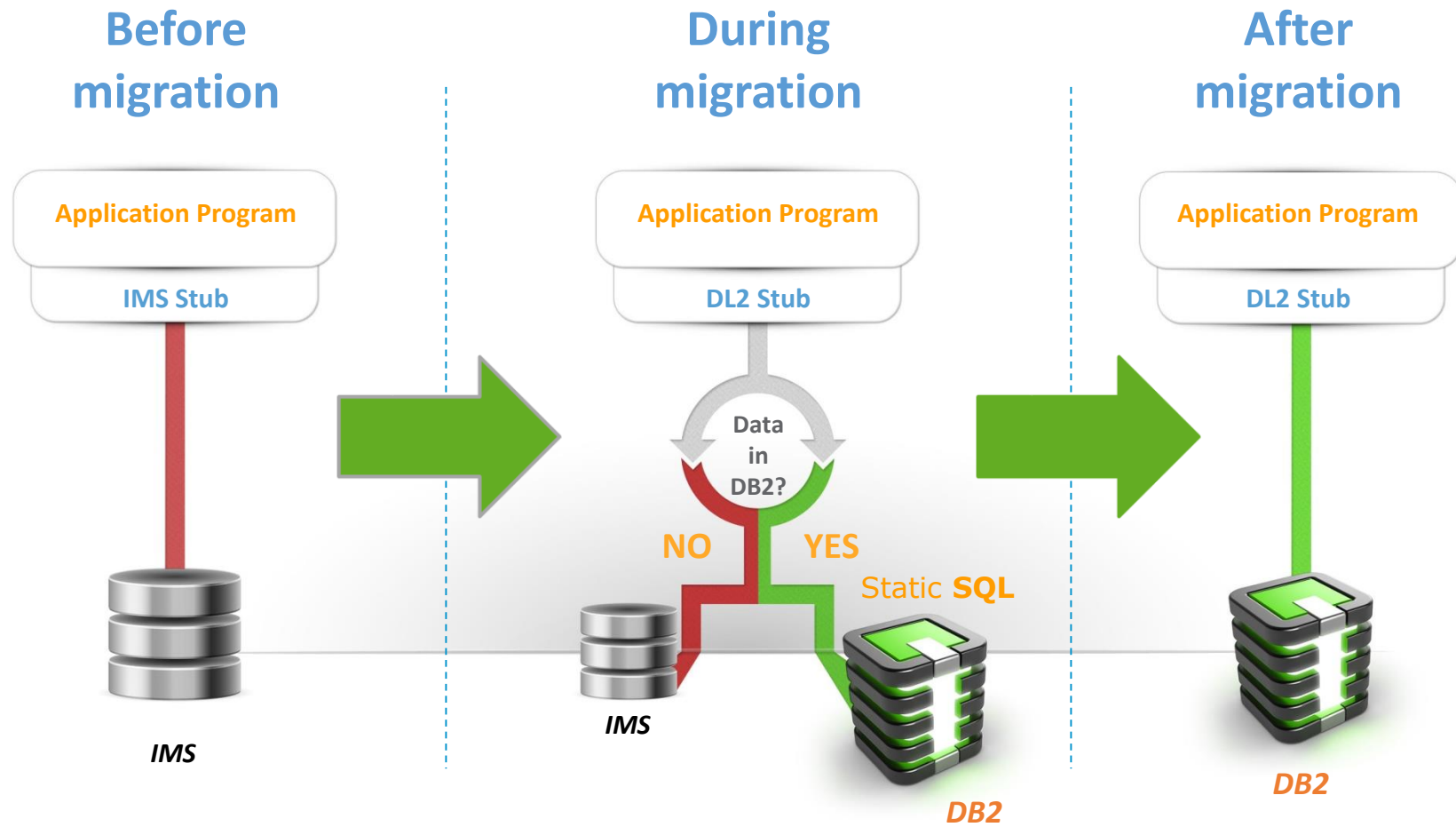
- [Blue Phoenix](#) claims to automatically convert IMS programs to DB2 programs
 - Their website link www.bphx.com takes you to [Modern Systems](#)
 - IMS/DB, VSAM, IDMS and ADABAS are supported sources
 - Mainframe and non-mainframe targets, including Oracle and SQL Server
- [HTWC](#) Based in Italy, claims to provide transparent IMS/DB migration product
 - Non-mainframe targets, including Oracle and SQL Server
- HiRel claimed to automatically convert IMS programs to Db2 programs
 - Converted COBOL code was very difficult to maintain
 - I can find no evidence that the company still exists
- Anecdotally, other companies claim to be able to rehost IMS applications in Linux/Unix

Are there automated conversion tools?

- [Elevate IMS](#) from Syncsort
 - Successfully deployed in many Production systems across the world
 - Major banks, Governments, Manufacturing, Retail and Distribution
 - Entering its 26th year in production deployments
- What makes Elevate IMS different?
 - Lowest risk: **no** application code changes required
 - Lowest risk: migrate one or more databases at the same time
 - Lowest risk: uses standard IBM APIs and interfaces
 - Lowest risk: application programs can access any mix of migrated and non-migrated databases

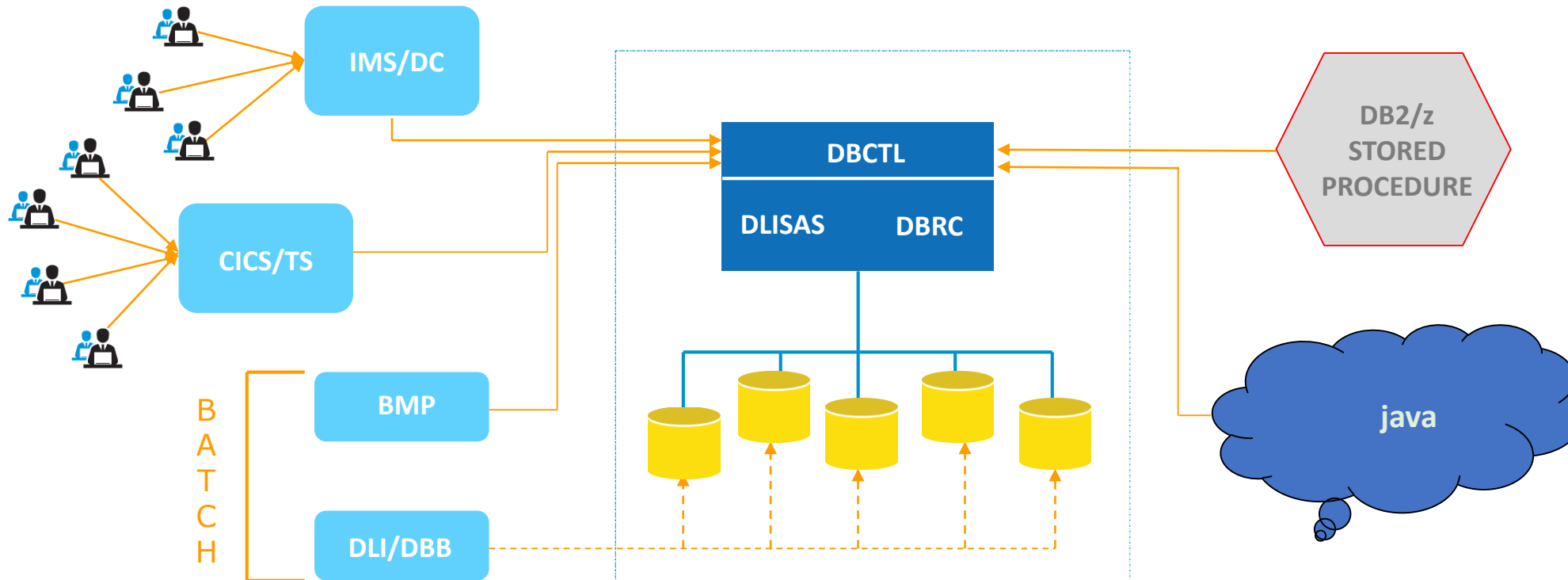
So how does it work?

Elevate IMS run-time overview



- Batch utility relinks application programs
- Relinked programs can access IMS/DB as normal

Elevate IMS – supported environments



APIs

- xxxTDLI
- EXEC DLI
- AIBTDLI
- AERTDLI

Languages

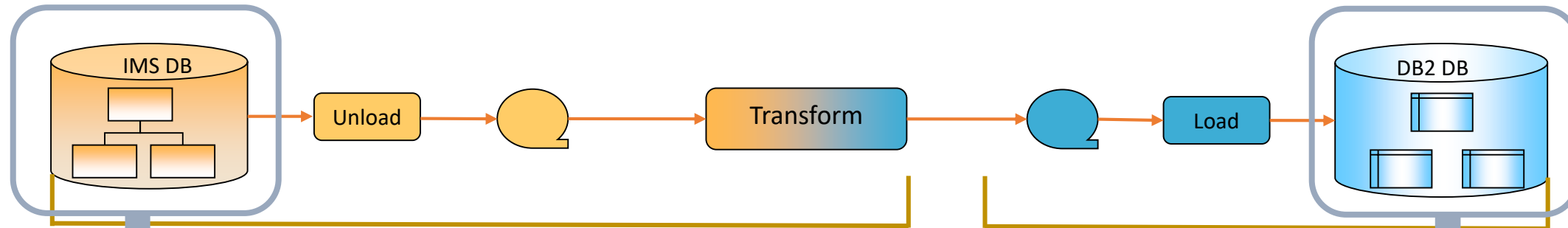
Assembler
COBOL
PL/I and LE/370
Java

Functions

IMS database calls
IMS system calls
IMS command codes

Elevate IMS data migration

Conceptually:



In practice:

- Batch job unloads and reformats IMS data
- Combination of DL/2 utility and system sort
- Optional steps for HERE, FIRST segments
- Generated JCL & control cards

- Batch DB2 load utility job(s)
- In referential integrity sequence
- Generated JCL & control cards

COMPARE

DL/2 can identify and reports inconsistent data prior to migration
Use this to decide optimal solution

What is your objective?

- Do you want to eliminate IMS/DB?

- To reduce costs
- Because of lack of expertise
- To consolidate DBMSs



- Do you want to be in Db2?

- Supporting rapidly changing business requirements, data lakes, ad hoc reporting
- Current IMS data needs to be available to new applications

These questions may look the same but they have entirely different answers

Summary

- IMS is still the premier solution for high-volume transaction environments
 - Unparalleled scalability and performance
 - Continued enhancements ensure it remains relevant
 - For some it is the only possible solution
- But:
 - Diminishing skills and cost means it is often under scrutiny
 - Kicking it into the long grass is not a good strategy
 - There are alternative solutions
- I was only kidding about the free beer



I'll leave you with this:




Please submit your session feedback!

- Do it online at <http://conferences.gse.org.uk/2019/feedback/hk>
- This session is HK



1. What is your conference registration number?


 This is the three digit number on the bottom of your delegate badge

2. Was the length of this presentation correct?

 1 to 4 = "Too Short" 5 = "OK" 6-9 = "Too Long"

1 2 3 4 5 6 7 8 9

3. Did this presentation meet your requirements?

 1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9

4. Was the session content what you expected?

 1 to 4 = "No" 5 = "OK" 6-9 = "Yes"

1 2 3 4 5 6 7 8 9