ISSN No:-2456-2165

Practical Transformation Approach to Holistic Legacy Modernization

Pijush Biswas, Kailash Verma

Abstract:- In the arena of cloud technology, the run and manage of enterprise is driven by optimized computing without compromising the business continuity. The resource cost is optimized to operate as pay per use. So, it becomes imperative to take the legacy applications onto a cloud platform to minimize the TCO and add new features to existing software. To make the cost optimization in cloud platform, the business applications must run in distributed manner in a cloud platform according to best pattern & practice. It is the imperative that we identify all bits and pieces of a business problem in the source legacy environments before we plan a migration move. Here we would discuss the method and process to discover the unknowns as the challenge itself remains the difficult to identify the legacy system component in existing architecture because of dearth of documentation or lack of architecture knowledge on ground who operates these applications.

I. INTRODUCTION

To support large scale application migration use-cases, one need to understand bits and pieces of the applications and when the applications are huge in number and distributed over heterogeneous operating system and programming languages, the task of legacy application migration becomes unmanageable. The formulation of strategy, collection of data, analysing and correlating become huge tasks. There is a need for systematically discover the various component of problem statement so here these propositions. The task of getting insights of existing business modules, system architecture, running databases are not readily available and at time they not available with application owners. The legacy application themselves have gone out of warranty with no documentation left. Only good news that application owners are happy that these applications still support their today's business. When it becomes clear about a problem statement, plan to move can be started. The feasibility is done for next move to move the application into a compatible cloud. How to do we identify the source environment and what process are really make effective legacy modernization practical? Is there a way to identify a practical approach and do we have a defined path?

II. PROBLEM STATEMENT

Here the systems and methods of architecting applications are more targeted to multiple host platform and vendor neutral. The deployment of business applications is virtualized to take benefits of easy shift, lift, pay per use models. Legacy systems and methods are not necessarily one to one to cloud counterpart, as then need careful planning to reorganize, replatform and rearchitect to support target platform before they become available for modernization and migrations.

The migration is very simple problem. It just talks about getting one application from a source environment to a target environment. The source environment in a simpler case is a data centre and in a simple case target environment is another data centre. If the move is planned as one to one, it remains simple move. The move may contain number of servers, networks, hard disks. When there is a business intelligence attached to simple problem, it becomes a medium problem. There are instances of heterogeneous operating system which need replatform, there are obsolete operating system which needs upgrades. When operating system upgraded, the related applications are become candidate for upgrades. It becomes really challenging to understand source application architectures, pre-existing systems, inbound and outbound interfaces, involved databases. These are bits and pieces of the said applications. The problem statement itself needs a discovery and redefine the problem with a reverse engineering.

III. BACKGROUND

A case of legacy application running on mainframe and the application owner wants to put it onto cloud platform to have better run and manage benefits and integrate better with modern applications interface. If we see these applications, they must be having a user interface, they must be having a business logic running at core and off course these applications also putting the processed data into a database. The challenge is how to identify each one of these players:

Let's break the problem statement into 3 categories then:

- Which are the user interfaces? Which are the languages, platform & hosting mechanism?
- Which are the business logic area? Which server process take care of these business rules?
- Which are the databases which holds the data? Does the data have a backup or fail over strategy implemented?

With little or no documentation, these is little or no answer to these problems. But the business to continue and so is the civilizations. To overcome the problem, a discovery process (TADDM) in installed to scan the server process, operating systems, network traffic, CPU performance, memory utilizations, hard disk usage, middleware installed, languages installed platform used, database installed etc. The data are collected by discovery process and merged into a single CMDB workspace for rigorous analysis and extrapolation. The data patterns are matched with experienced administrators to find out a workable architecture. That becomes a basis of further workshops to find gaps in the data and thereafter designing a technical solution.

IV. SOLUTION

Solution component starts with understanding workload, programmable components, printers, scanners or external output devices, input devices like console, service components like batch jobs. The components are scattered and undocumented in terms of upstream, downstream, endpoints, integration points to come up with deployable components like the figure given below:

The existing applications & infrastructures are thoroughly examined by automated scripts installed in the environments. Here we take a problem of legacy modernization of a mainframe application where discovery is run with following angles:

A. Mainframe Application discovery
B. Storage & Middleware discovery
C. Database discovery

A. Mainframe discovery

The process & method to have discovery process for Mainframe zOS DLA process by TADDM:

- Perform the mainframe discovery using TADDM Domain Server
- Execute the mainframe discovery using the zOS DLA process on mainframe servers
- Load the xml output into the TADDM server using the TADDM Discovery console
- Generate the Mainframe discovery report from the TADDM Data management portal
- Enhance the CMDB Analytical database to process the Mainframe discovery data elements and report generation
- Identify the mainframe data elements, storage elements and other hardware devices
- Create database structure in CMDB database with identified data elements
- Create a rollup process to move the mainframe data from leaf tables to master table
- Enhance the autoload to load the mainframe discovery data elements into CMDB for further analysis
- Create a new parser to parse the mainframe data elements from xml file (output of mainframe discovery)
- Create / Enhance a wrapper to load the parsed data into tables and views which are part of report generations

❖ *Accomplishments*

- Identified the TADDM zOS DLA with process document to perform the discovery
- Arranged the Mainframe test server to perform the zOS discovery
- Shared the zOS DLA and documentation with zOS SME to execute the DLA
- Received the xml output files
- TADDM was enhanced to perform the mainframe discovery

B. Storage Discovery

The process and method to discover operating system & middleware on legacy systems by script utility written in scripting language on Windows/Linux/AIX:

➢ Enhancements to capabilities of script utility on Windows/Linux/AIX

- Server-side inquiry through HBA
- Complete SCSI Registry is collected and parsed
- HBA based command line inquiry on Linux
- Configuration data on AIX
- Derived Values (Number of disks, capacity, HBA make, model, firmware,
- Driver, ports, WWPN) –

➢ Incorporate and integrated scripts to collect storage data on physical and LPAR/VIOS.

➢ Develop a new parser on Windows/AIX to load new discovery entities into CMDB analytics engine

❖ *Accomplishments: -*

- Integrated with CMDB to load collected metrics from AIX and Windows
- New report "Deployment Matrix" is added into Custom reports
- Parser and Auto load enhancements are integrated with current CMDB version.

C. Database Discovery

Identifying databases by discovery process to draw the target architecture

➢ *Background*

- Oracle, DB2 UDB, Sybase, MySQL and SQL Server etc data discovery scripts has been integrated with Middleware discovery scripts via DBMORE switch.
- Most of DB scripts are integrated/automated with CMDB analytical tool. The parsers and autoload for high level elements are in place with CMDB.
- Deep dive database discovery is available from DB competency but to review and include in CMDB analysis process & report generations.

➢ *Current Status*

- Independent data discovery scripts exist for all the major DBMS: Oracle, Sybase, DB2, Informix and SQL Server.

ISSN No:-2456-2165

- Oracle, DB2 UDB, SQL, MYSQL data are collected in most of the accounts. Sybase DB info is collected on MPI and SOCC.
- CTL has reported some Informix servers. Current middleware discovery tool and process are working as expected.

A typical analysis and insights are prepared from collected discovery from mainframe component of a given problem are categorized into following distributed heads:

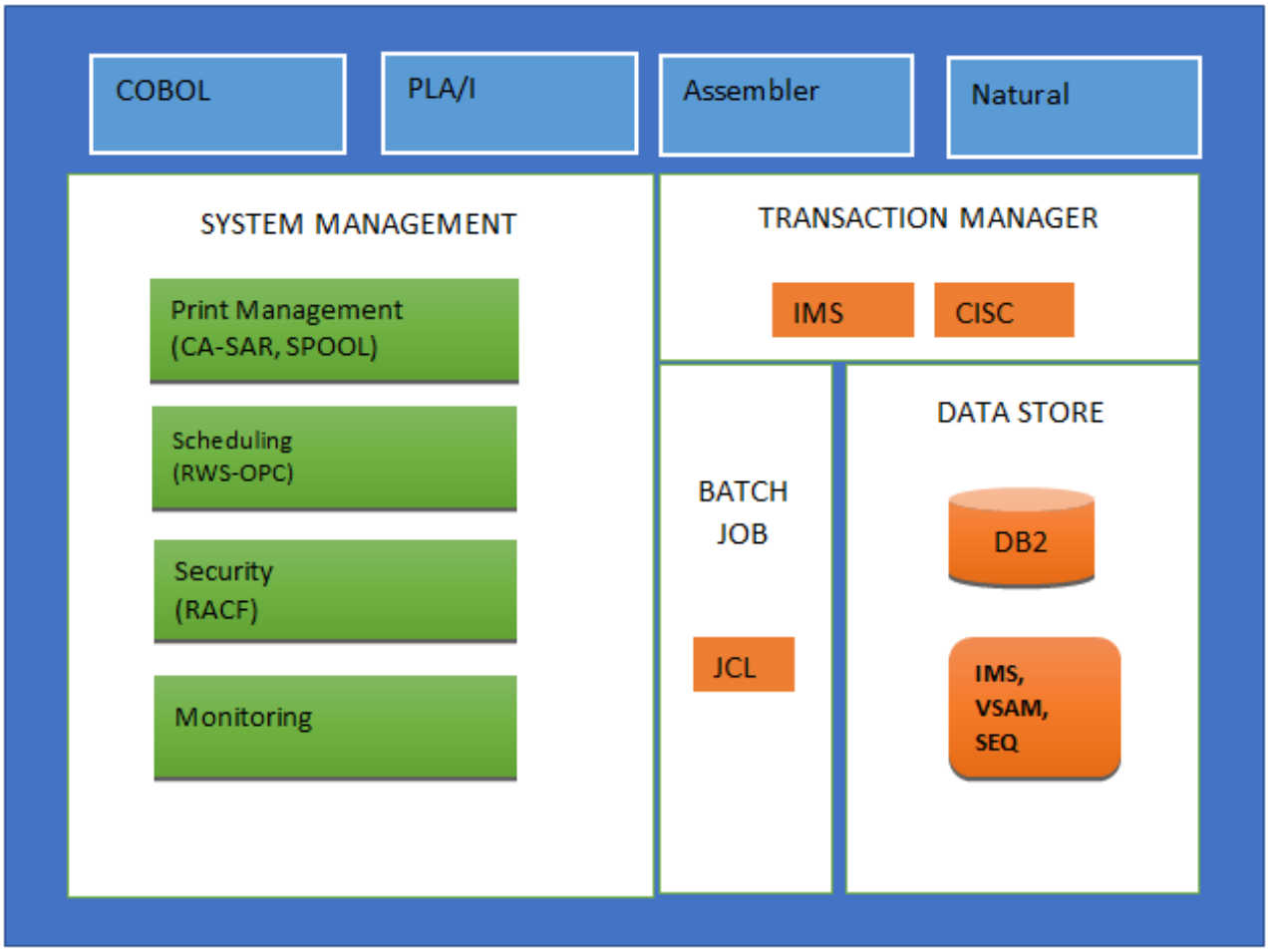

Fig 1

V. CONCLUSION

Identification of the given services are key to success for any kind of mapping from source application to destination application. Either it is data centre vs data centre or data centre vs public cloud.

To get the solution get hosted on cloud, typical services are categorized into BATCH JOB, DATABASE SERVICES & TRANSACTION MANAGER.

When doing a like to like migration, each application is marked for a target design. An emulator for mainframe transaction processing monitors which must be supported on target cloud platform. Since the databases are for the said application is a relational database (DB2) which is can be installed on any virtual machine as IaaS services on supported on clouds. The database can also be replatformed to DB2 LUW, MS SQL Server or Oracle database. Since an emulator support IMS, VSAM and SWQ, the virtual file system would be implemented by cloud services.

The batch jobs are most common part of a mainframe application. The TP manager on the target platform take care of hosting the jobs. The jobs that uses JCL on mainframe are hosted on emulators. Screens elements are forms application by modern language, and the business logic are implemented in a modern language like C#, vb.net, java or nodejs. The business layer are objects are modernized with modern database drivers for respective databases. OLEDB drivers or ADO.NET are suitable for a Microsoft product like .net, C# based services. For java-based application, JDBC database connections drivers are best fit for database communications.

Mainframe's system management tools are replaced by cloud services available on a given target platform or by 3rd party services.

REFERENCES

[1]. Mainframe COBOL JCL DB2 IMS DB VSAM CICS Express Reference - Sibashis Nanda
[2]. Introduction to the New Mainframe z/OS Basics - Mike Ebbers,John Kettner,Wayne O'Brien,Bill Ogden
[3]. Mainframe System: https://ibmsystemsmag.com/IBM-Z/08/2019/effectively-document-mainframe-architecture
[4]. Mainframe on wiki: https://en.wikipedia.org/wiki/Mainframe_computer
[5]. https://ibmsystemsmag.com/IBM-Z/08/2019/effectively-document-mainframe-architecture
[6]. Discovery Component: https://developer.ibm.com/mainframe/products/ibm-application-discovery/
[7]. CISC:https://www.ibm.com/support/knowledgecenter/en/SSTQBD_12.0.2/com.ibm.zsys.rdt.tools.user.guide.doc/topics/component_cics.html
[8]. Mainframe Software List: http://ibmmainframes.com/wiki/software-list.html
[9]. Mainframe components: https://content.pivotal.io/slides/how-to-migrate-applications-off-a-mainframe