Database Migration based on Trickle Migrations Approach

Khin Aye Nyeint , Khin Mar Soe
University of Computer Studies, Yangon
ilpyaepyae@gmail.com,kmsucsy@gmail.com

Abstract

The banking industry has undergone a major change in recent years. Global competition has forced the industry to be more agile and customer focused in all its services. Banks can no longer function in isolation, but have to operate cutting across physical boundaries. Scalability, maintainability, and security are the upcoming challenges for the banking industry. In addition to the risks associated with replacement, there are also several risks banks face if they decide to postpone a core system replacement. Those banks continuing to run antiquated systems are often challenged by inefficiency. This paper is proposed to addresses the problem of database migration used in legacy banking system by using Trickle Migrations Approach. In our proposed problem area, banking, the legacy database is (ACE Database) and it will be migrated to (Flexcube) database.

Key words: *Data Migration, Trickle Migration*

1. Introduction

Evolution in information and communication technology (ICT) has radically influenced the business world – how transactions are being carried out, how information is being exchanged, and how business collaborations are being handed. The banking sector is no exception as it plays a vital role in facilitating all kinds of business-related financial transactions.

Thus, there is a growing need for the banking sector to keep pace with the emerging requirements of the business sector by adopting appropriate technology for its effectiveness. The emphasis today is on providing banking services anywhere, anytime, to anybody, with the fundamental objective of enhancing customer outreach and flexibility in transactions. Towards this end, the adoption of Internet banking, mobile banking, core banking.

2. Related Work

The related works are discussed in this session.

Legacy Information System (ISs) should be migrated into computing environments and architectures that will avoid future legacy IS problems. Although this vision goes far beyond that described above, it illustrates the presence of continual change and aspects of next generation ISs towards which current (legacy) ISs might be required to migrate [1].

Information System (IS) technology should support continuous, iterative evolution. IS migration is not a process with a beginning, middle, and end. Continuous change is inevitable. Current requirements rapidly become outdated, and future requirements cannot be anticipated. The primary challenge facing IS and related support technology is the ability to accommodate change (e.g., in requirements, operation, content, function, interfaces, and engineering) [2].

3. Background Theory

Data migration is the process of transferring data between computer storage types or file formats. It is a key consideration for any system implementation, upgrade, or consolidation. Data migration is usually performed programmatically to achieve an *automated migration*, freeing up human resources from tedious tasks. Data migration occurs for a variety of reasons, including server or storage equipment replacements, maintenance or upgrades, application migration, website consolidation and data center relocation. To achieve an effective data migration procedure, data on the old system is mapped to the new system utilizing a design for data extraction and data loading. The design relates old data formats to the new system's formats and requirements. Programmatic data migration may involve many phases but it minimally includes *data extraction* where data is read from the old system and *data loading* where data is written to the new system.

After loading into the new system, results are subjected to data verification to determine whether data was accurately translated, is complete, and supports processes in the new system. During verification, there may be a need for a parallel run of both systems to identify areas of disparity and forestall erroneous data loss. Automated and manual data cleaning is commonly performed in migration to improve data quality, eliminate redundant or obsolete information, and match the requirements of the new system.

Data migration phases (design, extraction, cleansing, load, verification) for applications of moderate to high complexity are commonly repeated several times before the new system is deployed.

3.1. Categories of Migration

Data is stored on various media in files or database, and is generated and consumed by software applications which in turn support business processes. The need to transfer and convert data can be driven by multiple business requirements and the approach taken to the migration depends on those requirements. Four major migration categories are:

Storage migration: A business may choose to rationalize the physical media to take advantage of more efficient storage technologies. This will result in having to move physical blocks of data from one tape or disk to another. The data format and content itself will not usually be changed in the process and can normally be achieved with minimal or no impact to the layers above. Hence called Storage migration.

Database migration: Similarly, it may be necessary to move from one database vendor to another, or to upgrade the version of database software being used. The latter case is less likely to require a physical data migration, but this can happen with major upgrades. In these cases a physical transformation process may be required since the underlying data format can change significantly. This may or may not affect behavior in the applications layer, depending largely on whether the data manipulation language or protocol has changed.

Application migration: Changing application vendor will inevitably involve substantial transformation as almost every application or suite operates on its own specific data model and also interacts with other applications and systems within the enterprise application integration environment. Furthermore, to allow the application to be sold to the widest possible market, commercial off-the-shelf packages are generally configured for each customer using metadata. Application programming interfaces (APIs) may be supplied by vendors to protect the integrity of the data they have to handle.

Business process migration: Business processes operate through a combination of human and application systems actions, often orchestrated by business process management tools. When these change they can require the movement of data from one store, database or application to another to reflect the changes to the organization and information about customers, products and operations.

3.2. Reasons to Perform Data Migration

- Acquisition and merger of business unit/organization that triggers process change in organization.
- To improve efficiency, performance and scalability of software application.
- To adapt new changes in terms of technology, Market practice, operational efficiency, regulatory requirement results in better customer service.
- To bring down operational cost and efficiency by streamlining and removing bottleneck in application process or when different data centers were moved to cluster in one location.

3.3. Data Migration Strategies

Even though from some points of view data migration seems problematical and difficult to apply, no one actually doubts in its necessity. More or less regularly almost all companies which operate on data have to update their systems, applications, platforms. It's a natural situation which originates from the fact that not only computer systems are changing, but also so is business. What it means is the fact that a need for data migration might not only be a consequence of data storage system becoming outdated, but also the result of changing business condition.

Ensuring the best data quality, also through efficient data migration, is crucial to responsible management in 21st century world. Data migration,

being a method for letting data originated from one source be compatible with another which it's going to be loaded into, is simple only at first sight. The deeper one knows the problem, the more questions he has, beginning with the most important one - how to make company suffer least because of data migration. In fact, it depends on chosen strategy. Basically, there are two different strategies, two different approaches to data migration. And they differ mainly in the way migration is proceeded.

Two types of migrations strategies are Big bang Migration and Tickle Migration.

3.4. Big Bang Migration

The first strategy, called big bang migration, is someway uncompromising. In a word, it suggests shutting all applications and databases immediately, stopping the work and putting all force in data migration. In fact, it really seems to be a good option, because only this way guarantees that the migration lasts as short as possible. Moreover, it almost eliminates the risk that something unpredicted happens during the process. On the other hand though big bang migration might be destructive to organization's work. Especially in cases of companies which depend on data in a real time, being cut from data might be a real problem.

There is, of course, a way to minimize the negative influence of big bang migrations. In most companies which decide to choose this type of data migration strategy, the process is being initialized after work hours or on holidays. This way, cutting off the access to data may cause the least problems.

Table 1. Big bang migration

Big bang migrations	
Advantages	**Disadvantages**
• No one has to operate their business in two different operating systems. • Training needed only on the new system, not a changeover	• Failures in one part of the system can cause problems and failures in others. • Users have to learn the new system immediately – this could result in a dip in performance

3.5. Trickle Migration

In fact, the work within most companies last 24 hours long, during weekends and public holidays as well (even if employees actually have a day off, systems can't enjoy a break). Thereupon, managers often cannot afford to turn off the systems even during holidays. However, data migration still has to be run anyway. Fortunately, there is an option to migrate data without a need to shut the whole system. It's called trickle migration and is performed during the normal work of all involved systems. The idea behind trickle data migration is not to shut the whole system at once, but operate only on its chosen areas so that all other could be accessible at the moment of migration. This way, employees keep continuous access to data, even though migration might last even 24/7.

Table 2. Trickle migration

Trickle migrations	
Advantages	**Disadvantages**
- can be run during weekends, holidays, etc.	· obligatory organization systems' downtime · risky

4. Proposed System

Proposed System includes the following tasks to complete the migration.

1. Migration of old database to new database.
2. Serving of new transactions of existing accounts during migration.
3. Creation of new accounts during migration eg. Deposit, check balance, withdraw.

4.1. Migration of Old Database to New Database

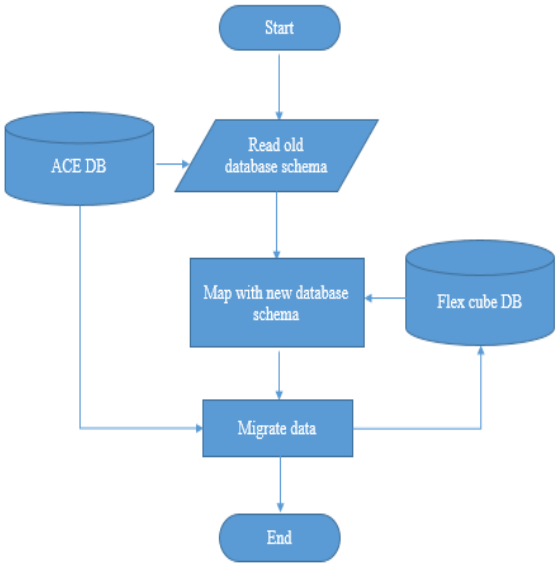

Figure 1. Database Migration Flow

At the start of database migration, the system read the old database schema from the old database (ACE database) and map the data with new database schema. Then, the system migrate the old database format to new database format. Detail steps of migration processes are as follows:

4.2. Migration of Customer Account

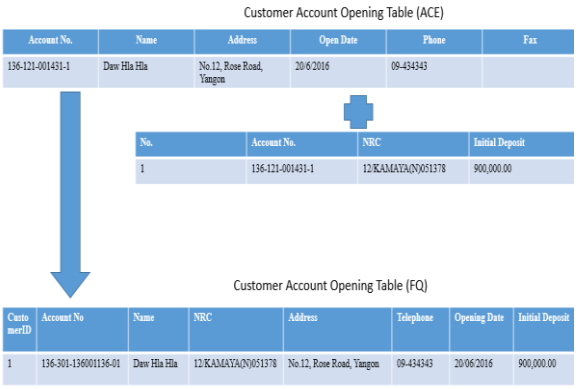

Figure 2. Migration of Customer Accounts

Algorithm for Migration of ACE Account to Flex-cube Account

BEGIN

Set T1 = list of columns in account table from ACE;

Set T2 = list of columns in account table from Flexcube;

Var st = Account No in T1;

split st into 4 columns → C1,C2,C3,C4;

Branchcode of Flexcube Account in T2 ← C1;

Convert account type of C2 → new format of flexcube account type;

Type of Flexcube Account in T2 ←new format of flexcube account type;

Remove C3;

Account No of Flexcube Account ← account holder no. of core system;

Account ownership of Flexcube Account ← "0" +C4;

END

4.3. Migration of Daily Drawing Remittance Listing

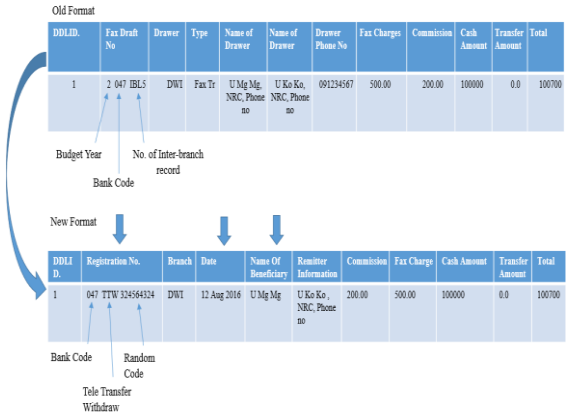

Figure 3. Migration of Daily Drawing Remittance Listing

Algorithm for Migration of Daily Drawing Remittance Listing (DDRL)

BEGIN

Set T1 = list of columns in DDRL table from ACE;

Set T2 = list of columns in DDRL table from Flexcube;

Sr_No in T2 ← Sr_No in T1;

Var st = Fax Draft No in T1;

Split st into 3 columns → C1, C2, C3;

Remove C1;

Bank code of Registration No in T2 ← C2;

Tele Transfer Withdraw of Registration No in T2 ← TTW;

Random code of Registration No in T2 ← Get random generated code;

END

4.4. Migration of Hire Purchase Position Listing

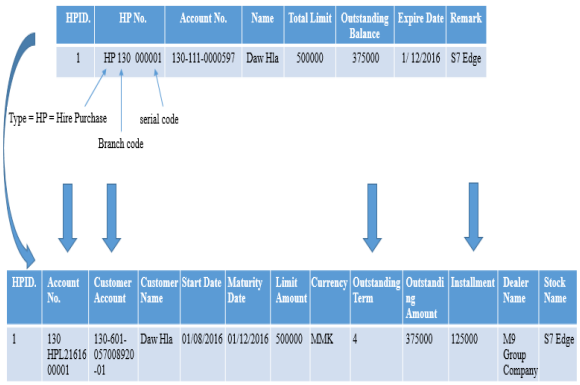

Figure 4. Migration of Hire Purchase Position Listing

Algorithm for Migration of Hire Purchase Position Listing

BEGIN

Set T1 = list of columns in HP table from ACE;

Set T2 = list of columns in HP table from Flexcube;

HPID in T2 ← HPID in T1;

Var st = HPNo in T1;

Split st into 3 columns → C1, C2, C3;

HPNo in T2 ← C2 + "HPL" + random code;

Installment in T2 ← totalLimit in T1 – Balance int T1;

Outstanding_Term in T2 = totalLimit in T1 / Installment in T2;

}

END

4.5. Migrating of Listing for Internal Remittance Outstanding

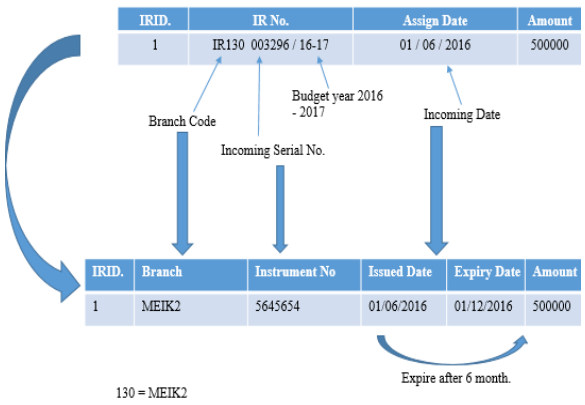

Figure 5. Migrating of Listing for Internal Remittance Outstanding

Algorithm for Migrating of Listing for Internal Remittance Outstanding

BEGIN

Set T1= list of columns in IR table from ACE;

Set T2 = list of columns in IR table from Flexcube;

Sr_No. in T2 ← Sr_No. in T1;

Var st = IR No. in T1;

Split st into 2 columns; C1,C2

Remove years from st;

Branch in T2 ← C1;

Instrument No in T2 ← C2;

END

4.6. Objective of the Proposed System

- To study database migration in banking.
- To use Trickle migration approach in bank database migration.
- To ensure that the current software is up-to-date and that any new software is installed and transferred successfully.
- To successfully test, administer and implement new computer databases within an organization and coordinate any moves to a new system.

4.7. Database Schemas

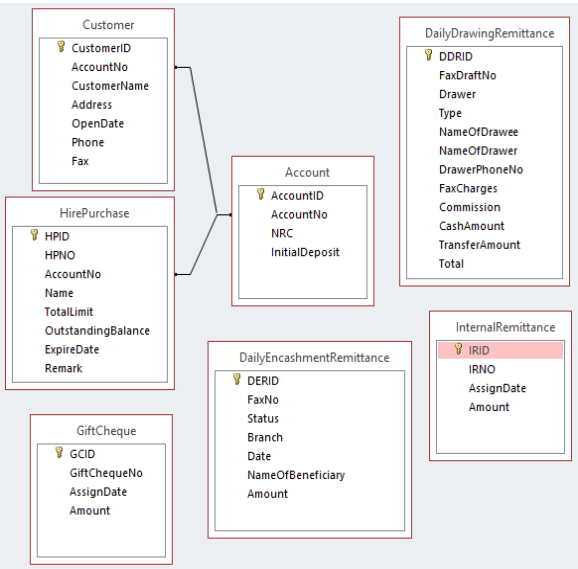

Figure 6. Old DB Schema (ACE DB)

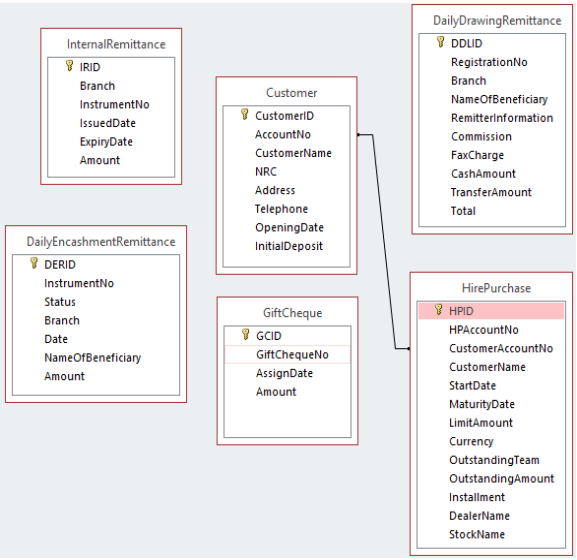

Figure 7. New DB Schema (Flexcube DB)

5. Conclusion

Data migration is a routine part of IT operations in today's business environment. Even so, it often causes major disruptions as a result of data quality or application performance problems and it can severely impact budgets. To prevent these problems, organizations need a consistent and reliable methodology that enables them to plan, design, migrate and validate the migration.

References

[1] Brodie, M.L. and S. Ceri, "On Intelligent and Cooperative Information Systems,", International Journal of Intelligent and Cooperative Information Systems 1, 3 Fall.

[2] Huff, K.E. and O.G. Selfridge, "Evolution in Future Intelligent Information Systems," Proceedings of the International Workshop on the Development of Intelligent Information Systems, Niagara-on-the-Lake, April 2007.

[3] Goldman, N. and K. Narayanaswamy, "Software Evolution through Iterative Prototyping," in Proceedings of the 14th International Conference on Software Engineering, Melbourne, Australia, May 2000.