# A Best Practice of Java-Based Applications Migration with Variety of Software Solutions

Yung-Shih Wang
Billing Information Laboratory
Chunghwa Telecom Laboratories
Taipei, Taiwan, R.O.C.
dolaamn@cht.com.tw

Chih-Chin Yang
Billing Information Laboratory
Chunghwa Telecom Laboratories
Taipei, Taiwan, R.O.C.
larkyang@cht.com.tw

***Abstract*—Business operation systems in the design period, not only consider the development of related technologies, but also the selection of different kinds of software platform solutions, such as application server and relational database are particularly important. In addition to the platform operability, cost considerations also affect a large portion of the decision results. If system programmers do not take the different business operation execution environments into consideration in the step of system development, when the system established complete, due to system requirements change or cost considerations and need to adopt other software solutions, migrate business operation systems to different application servers or relational databases will spend extra time and effort for system modifications. This paper proposed an empirical study with an instance of A&A (Authentication & Authorization) Management System (AAMS) to actually migrate between different kinds of application servers and relational databases, and proposed several common principles of system migrations and system design rules for variety of software solutions. Moreover, it reduces the time which it needed and lowers system development expenses significantly during system migration.**

***Keywords—authentication; authorization; migration principle; software solution;***

## I. INTRODUCTION

The system which provides a variety of business operations can be referred to as large-scale information systems. The business operations provided by these systems can deploy to different kinds of heterogeneous platforms, fulfill the requirements demand from all users. Accompany with many type of businesses, using large-scale information systems to provide all kinds of business operations is a standard model for enterprises.

However, it is important to consider the platform compatibility when developing large-scale information systems, such as the Operation Support System (OSS) and the Business Support System (BSS), which not only improve the negotiation advantages between system venders, but also reduce the cost expense under a specific platform.

Beginning of the system design, because of the feature of platform-independent and the advantages of easy-to-develop and easy-to-deploy, Java-based applications are widely adopted by large-scale information systems, accompany with the characters of applications only write once and run anywhere. Moreover, develop the large-scale information systems, not only estimate the underlying hardware architecture implementation costs, but also the system software platform implementation, such as application servers, relational databases, version control servers, and in order to ensure the stable operations of the system to achieve service level agreements, it must add another service performance monitoring system software, and costs of system development is expensive.

For the sake of reducing the cost of system development in large-scale information systems, the system always developed by using the same technologies. However, to decide what kinds of different software solutions are better can based on the characteristics of the business operations and business service level agreements, and finally reach the purpose of lower the system construction costs.

This paper proposed an empirical study with an instance of A&A (Authentication & Authorization) Management System (AAMS), adopted the most existing enterprise software solutions, such as commercial support and open source application servers and relational databases. Additionally, the paper compares the different features about application server WebLogic and JBoss, and relational database Informix, Oracle and MySQL, concentrate on proposed some common principles of system migrations and system design rules for the purpose of reduces the time which it needed and lowers system development expenses significantly during system migration.

Finally, the paper uses a real case about AAMS with another business operation migrates between different type of application servers and relational databases, and provides discussions about the system development costs and analyzes the related migration benefits.

## II. RELEATED WORK

Under the impact of globalization and deregulation, all of the enterprises not only provide better services and resources, but also fulfill the requirements demand from all users. Migration activities range from changing hardware platforms due to their obsolescence, changing operating system, changing architecture, and changing client or server interface. There are more and more system migrations for some reasons, such as system development costs, system performance augmentation, or degree of the commercial support.

Software migration concerns the adaptation or transformation of an existing system to a new technological context. It proposed a detailed report, performed among several companies with the aim of investigating about the experiences in software migration, and analyze on characteristics of projects each company decided to report about, such as OS migration, programming language migration, DB migration, architecture migration and UI migration [1]. For programming language migration, organizations expect high benefits from the application of object-oriented software development, and proposed a migration model based on an empirical study in order to reduce the risk of failure during the migration process [2]. There exist a number of legacy systems, and migrating existing systems to a SOA environment can be a best approach for reusing a legacy system. SOA migration is a complicated task. To ensure the migration performance, some directions on modeling migration approaches are needed. A user-oriented SOA migration model is addressed in [3], and SOA reengineering technical approaches are concluded in [4][5]. Three-tier middleware architecture is commonly used for enterprise-distributed applications. The second tier consists of application servers that receive requests from clients where the computations implementing the business logic are performed, and the third tier contains databases that maintain persistent data for applications [6]. It describes how replication for availability can be incorporated within the second and third tier, meeting incorporation of availability measures in a multitier system poses challenging system design problems of integrating open, nonproprietary solutions to transparent failover. Additionally, the article [7] is aimed to accomplish how the migration takes place between recent databases technological and reliable tool, and the article [8] study the approach and process of data migration and introduce the basic functions of each module in data migration process and the test procedures after data migration.

## III. JAVA-BASED APPLICATION SOFTWARE SOLUTION

### A. Java-Based Large-Scale Information System

Typical large-scale information system architecture is illustrated as Figure 1, it usually composed of the following components:

*1) Client:* Client represents anyone who can operate the business operations provided by the information system.

*2) Application Server:* Application Server provides a platform for Java-based applications, all the business operations can deploy on it to fulfill the requirements from users.

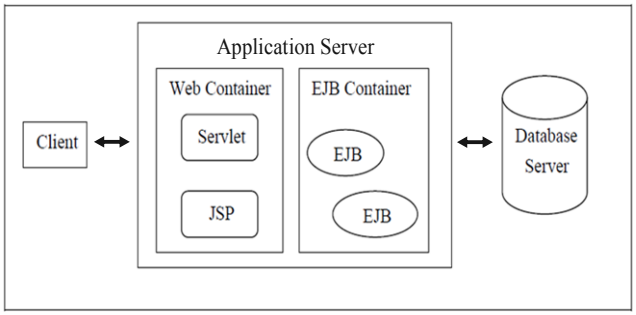


Figure 1. Java-based large-scale information system architecture

*3) Database Server:* Database Server used to store information related to business operations, including business data, historical records or the authentication and authorization properties of users.

Application Server in Java-based large-scale information system is divided into two functional parts, according to the different requests from users:

*1) Web Container:* When a user request is generated, the connection status of the request will determine the main web container, mainly related to the provision of the general web application execution environment, including user input or display output and other user interface processing.

*2) EJB Container:* When a user request is generated, the J2EE functions of the request will determine EJB container, provides the execution environment for enterprise beans. Its purpose is to control enterprise beans and provides system-level services, meanwhile, developers can focus on business logic. EJB container contains several features:

*a) Transaction management:* Provides enterprise beans transaction management.

*b) Security:* Control the permission to call enterprise beans functions.

*c) Life cycle management:* Provides life cycle management for enterprise beans.

*d) Database connection pooling:* Control database connection through connection pool, and provides mechanism to share database connections for enterprise beans.

### B. System Software Solutions

Nowadays, there are more and more different brands of application servers and relational databases adopted by large-scale information systems. Hence, the combinations of application servers and relational databases are more complicated, no matter commercial support or open source software solutions.

*1) Application Server:* Application server plays an important role in large-scale information system architecture, not only provides an execution platform for business operations, but also a communication middleware between users and relational databases. When design an execution environment for business operations, how to choose an appropriate application server is an important issue. Provide 7*24 uninterrupted business operations is the goal for large-

scale information system in enterprises. For this purpose, stability and high availability of business operations is one of the necessary conditions. Therefore, for the business operations continue to execution without interruption, the commercial support becomes one of the preferred application server. In addition to provide more complete running platform features, or even provide monitoring and management program that enable the service running continuously without any interruption to meet the needs of users at all times, such as order operations, billing operations or system management operations. If there is a cost considerations or the system is not required to provide services all the time, it might consider using the open source application server to build the environment, such as report operations or a large number of back-end batch processing operations.

Application server provides only partial J2EE implementation specifications cannot be called a J2EE application server, for example, Tomcat is just a web container, only provides all the J2EE specifications can called J2EE application server. In order to use the full functionality of J2EE, WebLogic and JBoss therefore are used as a platform for this paper. WebLogic and JBoss are complete implement J2EE functions, suitable for AAMS as an execution environment. The comparisons of these application servers are shown in Table I.

TABLE I. COMPARISON OF APPLICATION SERVER

| **Brand** | WebLogic | JBoss |
|---|---|---|
| **Version** | 11g | 6.0 |
| **Distribution** | Commercial | Community |
| **J2EE** | Full Support | Full Support |
| **Cost** | License/ Maintenance Fees | Free |
| **Onsite Support** | Yes | No |
| **References** | More | Less |
| **Class Loading** | Traditional Hierarchical Structure | Module Based |
| **Thread** | Share Thread Pool | Multiple Thread Pool |

*2) Database Server:* Except for application server, database also plays an important role in large-scale information system architecture. There is a variety of relational databases to choose from, including commercial support and open source versions of the databases, according to the system purpose and the cost of system construction. This paper uses three different types of relational databases for AAMS. The difference between these relational databases are shown in Table II.

TABLE II. COMPARISON OF RELATIONAL DATABASE

| **Brand** | Informix | Oracle | MySQL |
|---|---|---|---|
| **Version** | 11.7 FC8 | 9.2.0.8 | 5.5.33 |
| **Distribution** | Commercial | Commercial | Community |
| **Cost** | License/ Maintenance Fees | License/ Maintenance Fees | Free |
| **Onsite Support** | Yes | Yes | No |
| **Concurrent Connections** | High | High | Low |
| **Quantity of Data** | High | High | Low |

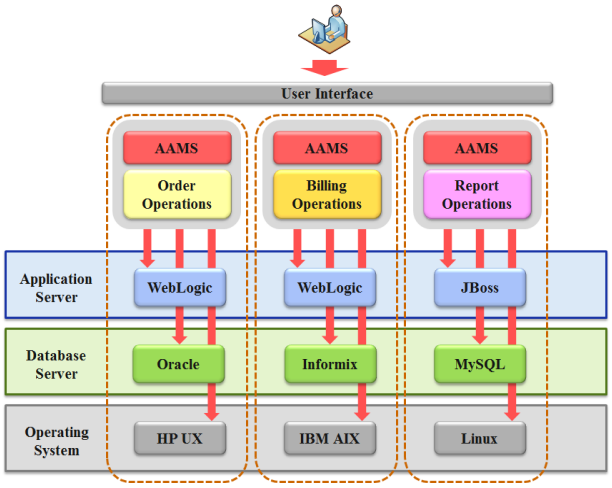


Figure 2. AAMS framework with variety of software solutions

## IV. IMPLEMENTATION WITH MIGRATION PRINCIPLE

This section will describe the issues and discussions about the implementation of AAMS, and apply for some implementation principles to achieve the purpose of migration without any program modifications.

### *A. Implementation Architecture of AAMS*

The implementation of AAMS is written by Java programming language with the migration principles proposed by this paper, so as to achieve cross-platform features. Hence, the entire system can run on different kinds of application servers and it can use different database environments as well. The overall system framework as shown in Figure 2, there is several business operations using AAMS for security control with variety of software solutions. Moreover, the architecture of AAMS as shown in Figure 3, divided into following modules:

*1)* Authentication

The main idea of the Authentication module is to ensure the user's identity, through LDAP, user certification authority and authentication information stored in the database.

*2)* Authorization

The main idea of the Authorization module is giving the rights to users for operating business operations via role management. The user can only operate the specific business operations for different roles, and authorization module provides the mechanism to record the operating history of the users.

*3)* Account Management

Account Management module provides the management of user information, user roles, user working unit, and operation information. Additionally, it also checks user accounts to enhance system security periodically.

### *B. Functionality of Authentication and Authorization*

The AAMS adopts the JAAS (Java Authentication and Authorization Service) framework to achieve cross-platform resources management and is not limited to various manufacturers' unique safety mechanism.

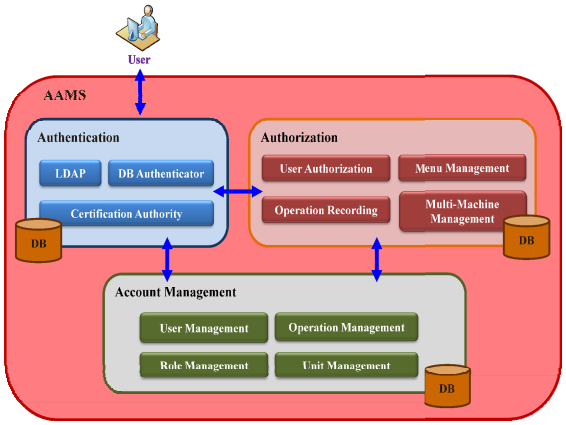


Figure 3. AAMS implementation architecture

How to create a common security model has been an integral part of the design in large-scale information systems. JAAS provides flexible mechanisms to protect the client or server-side Java programs, either client or server are needed to implement login and access control. JAAS verify the user running the program, so as to achieve the purpose to protect system. JAAS is a pluggable authentication framework and provides an easy way to integrate some standard security mechanisms into systems. Applications can easily append on authentication modules by using configuration, the JAAS framework is shown in Figure 4.

### C. *Migration Principles for Application Server*

To achieve business operations migrate between application servers, it proposed several common principles to follow.

*1) System property file:* Each of the different application servers has its own way to configure system properties. All system-related properties will need to reconfigure when application server platform changes. Therefore, if there are shared parameters, it can be independently set to a system property file. When the application server restarts, the system property file will load into the server immediately. Besides, business operation system can migrate between application servers completely without any program modifications. The system property file contains the parameters of the system-related information using name-value parameter format, such as user execution environment, business operation deploy addresses, file download storage path, and even user authentication and authorization methods. The example system property file is described in Figure 5.

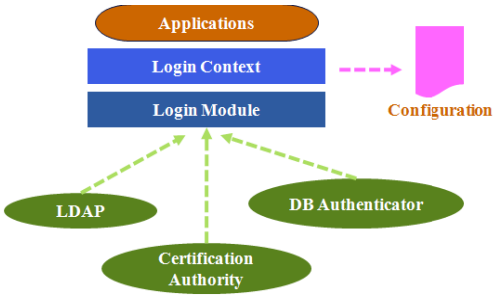


Figure 4. JAAS framework for AAMS

```
SquadA_W1=8006
SquadA_W1s=8007
SquadA_B1=8008
SquadA_B1s=8009
SquadB_A1=8061
SquadB_A1s=8060
SquadB_B1=8062
SquadB_C1=8064
SquadB_C2=8064
SYS_LDAP_CONNECT_TIMEOUT=5
SYS_LDAP_SEARCH_SIZELIMIT=20
SYS_LDAP_SEARCH_TIMELIMIT=20000
SYS_EMPINFO_QUERY_TYPE=DB
SOURCE IP BLOCK=true
SYS_ANW1_FILE_PATH=/home
```

Figure 5. System property file for AAMS

*2) Adopt JNDI (Java Naming and Directory Interface):* Communications between a program and a relational database is via JDBC driver. A JDBC driver is primarily used for accessing relational database via SQL statements. Each relational database vendor provides their own specific JDBC drivers, and the program must use the driver to get the database connection respectively. The AAMS implementation uses JNDI to get the data source connections and the data source connections are setup on applicaion servers with unique JNDI name. Business operation modules deploy on different kinds of application servers, when migrate between these application servers, it only needs to set the corresponding JNDI data sources without any program modifications. Migration can be complete easily without additional effort to spend. Data source configurations in WebLogic and JBoss are shown in Figure 6.

*3) Encode parameters:* Web-based business operation modules identify users through the session ID in the implementation of HTTP GET. The session ID generated by WebLogic and JBoss is totally difference. Sometimes there is a symbol "+" in the string of session ID generated by JBoss. When using HTTP GET to pass parameters through the internet, it will cause an interruption and parameter values obtained with errors. The AAMS encode all the parameters, including session ID, to avoid unexpected errors caused by the process of transferring parameters. The example string of session ID generated by WebLogic and JBoss are compared in Table III.

**WebLogic**

| ☐ | Name | JNDI | Target |
|---|---|---|---|
| ☐ | QA_SECURITY | jdbc.ROLE.roleadm<br>jdbc.SECURITY.security | SquadA_M1,<br>SquadA_W1,<br>SquadA_B1,<br>SquadC_D1,<br>SquadB_A1,<br>SquadA_A1 |
| ☐ | QA_TABLEUTIL | jdbc.TABLEUTIL.security<br>jdbc.TABLEUTIL.roleadm | SquadA_M1,<br>SquadA_W1,<br>SquadA_B1,<br>SquadB_A1 |

**JBoss**

| Name | JNDI | Enabled? |
|---|---|---|
| QA_01 | java:/jdbc.SECURITY.security | ✔ |
| QA_02 | java:/jdbc.ROLE.roleadm | ✔ |
| QA_03 | java:/jdbc.TABLEUTIL.roleadm | ✔ |

Figure 6. JNDI configurations for AAMS

TABLE III. COMPARISON OF SESSION ID

| Application Server | Session ID |
|---|---|
| WebLogic | nS2zTLjFC9ZQRyXJnrLv4wkLHnyRXhJvxV3QzBF B7c0hbyXYMhTp |
| JBoss | S7FzSzV2+ICi6D33NJcnZKyo |

*4) Deployment descriptor:* Do not use the deployement descriptor to carry out security control. Every application server has its own deployment descriptor to describes the classes, resources and configuration of the application and how the server uses them to serve requests. If the program uses the server specific deployment descriptor to execute security control, when the program migrates to another environment, the security control will fail. For example, WebLogic has its own security control in deployment descriptor file named "weblogic.xml" as shown in Table IV. When the program migrates to another application server, like JBoss, the security control will fail in JBoss. Hence, AAMS adopts JAAS to carry out security control to avoid these situations.

TABLE IV. DEPLOYMENT DESCRIPTOR SECURITY CONTROL

| Deployment Descriptor | Content |
|---|---|
| weblogic.xml | <security-role-assignment><br><role-name> </role-name><br><principal-name> </principal-name><br></security-role-assignment> |

## D. Migration Principles for Database Server

To achieve business operations migrate between database servers, it proposed several common principles to follow.

*1) Database reserved words:* To design the database, name of the database tables and fields need to avoid reserved words of different kinds of databases. In addition to some of the commonly used data definition language, such as "create", "alter" and "drop", data manipulation language, shch as "select", "update", "insert" and "delete", data control language, such as "grant" and "revoke", it also consider other words reserved by database. For example, "group" is often used to represent different business operation groups and "desc" is often used as an abbreviation of "description", but these words are all reserved words for database. Therefore, using compound words to name for database tables and fields is the better choice. When migrate between different kinds of databases, it can reduce the chance of a program to modify.

*2) Common data type:* When design data types of table fields in database, it needs to use common data types, do not use the unique style for specific database. If fields need to use unique data types necessarily, then the program needs to determine the database type in advance and choose the right SQL syntax. To take the commonly used time format as an example, Informix and MySQL use data type "datetime" to represent the time from year to second, but Oracle use "date" to express the same time style. If using java class "ResultSet" to get the query data with time data type from database, it should use the method "getTimestamp()" to get the data no matter what the data type is "date" or "datetime". For another example, every database has its own data type to generate primary key values automatically. When a sequence number is generated, the sequence is incremented, independent of the transaction committing or rolling back. The database Informix and Oracle use the data type "sequence", but MySQL use the data type "auto-increment" to increase the sequence. Besides, how to retrieve the generated sequence is not the same for these databases. In AAMS, it mostly using the same type format of different database field types, the part of the data types can not be unified, then the program needs to determine the way in advance, and execute different SQL syntax finally. The comparisons of commonly used data type are discussed in Table V.

TABLE V. COMPARISON OF DATA TYPE

| Data Type | Informix | Oracle | MySQL | Description |
|---|---|---|---|---|
| **Naming Length** | maximum length 24 bytes | | | Informix<=128, Oracle<=30, MySQL<=64, naming length<=24bytes |
| **String** | char(n) | char(n) | char(n) | n<=255 |
| | char(n) | char(n) | **x** | n>255 and n<=2000 |
| | char(n) | **x** | **x** | n>2000 |
| | varchar(n) | varchar2(n) | varchar(n) | n<=255 |
| | lvarchar(n) | varchar2(n) | varchar(n) | n<=4000 and n>255, MySQL<=1000 |
| **Integer** | bigint | number(n) | bigint | n=19 |
| | integer | number(n) | integer | n=10 |
| | smallint | number(n) | smallint | n=5 |
| **Date** | date, datetime year to second | date | date, datetime | when Oracle to Informix then date→datetime year to second, when Oracle to MySQL then date→datetime |
| | datetime year to fraction(3) | timestamp | decimal(17, 3) | when Oracle to Informix then only millisecond reserved |
| **Float** | smallfloat, float, decimal→default: p=16 | number→default: p=38 | float | when Oracle to Informix then number→decimal(32) |
| **Decimal** | decimal(p,s) | number(p,s) | decimal(p,s) | p<=32 and s>=0, Informix p<=32, Oracle p<=38 |
| **Boolean** | Unified char(1) with value 't' or 'f' | | | |

*3) Common SQL syntax:* When query to database, each database has its own special query grammar, such as Informix uses "first" to get the first number of records, Oracle uses

"rownum" to limit the interval to retrieve data items, and MySQL uses "limit" to limit the query data items, each database has its own manners to filter the records. AAMS uses the SQL syntax as simple as possible, does not use a unique database query grammar. If the program has to use unique query grammar, it must be determine the database type through the program in advance, and decide what kinds of grammar to query. It can also use the program language, such as "for loop", to control the number of records retrieve from the database, and reduce the opportunities to modify the program.

*4) Common SQL style:* When using conditional sentences in SQL, like "WHERE" clause, there is difference between using single quotation and double quotation in different databases. For example, Informix and MySQL allows the "WHERE" clause to use double quotation, but Oracle will report an error in the same situation, Oracle only permit to use single quotation for "WHERE" clause. So as to achieve the program migrate between different databases, AAMS uses single quotation in "WHERE" clause to unified query syntax in order to reach the purpose of cross-database queries.

*5) System time:* In AAMS, it often needs to record user login and logout time information. Each database has a different function to automatically generate the system time, such as Informix and Oracle use "sysdate" to generate the system time, but MySQL uses the function "sysdate()" to generate the system time. AAMS does not use the function provided by database to generate the system time, it obtains the system time by using Java implementation, like Java class "Date()", so as to avoid inconsistency way to get the system time between databases.

## V. MIGRATION BENEFIT ANALYSIS

In this section, it uses a real case about AAMS with another business operation migrates between different type of application servers and relational databases, meanwhile, it provides discussions about the system development costs and analyzes the related migration benefits.

Take AAMS with order operations for example, AAMS and order operation are original deploy on commercial support application server WebLogic accompany with relational database Oracle in operating system HP-UX. The construction cost including hardware and software of application server is up to 47.2 million dollars, and the cost of relational database is up to 28 million dollars. For another system migration solution, AAMS and order operation are deploy on open source application server JBoss accompany with relational database MySQL in Linux like operating system CentOS. The construction cost including hardware and software of application server is down to 5 million dollars, and the cost of relational database is down to 2 million dollars. Hence, the system migration follows the principles proposed by this paper will be more easily without any system modifications, and the system cost saving will up to nearly 90% compared to original solution. The detailed migration comparison of variety of system solutions is shown in Table VI.

TABLE VI. COMPARISON OF MIGRATION SOLUTION

| | **Before Migration** | | **After Migration** | |
|---|---|---|---|---|
| | **Hardware** | **Software** | **Hardware** | **Software** |
| AP | HP-UX | WebLogic | CentOS | JBoss |
| | 10 hosts ($2,600,000/host) Total: $26,000,000 | 80 cores ($265,000/core) Total: $21,200,000 | 10 hosts ($500,000/host) Total: $5,000,000 | Total: $0 |
| | $47,200,000 | | $5,000,000 | |
| DB | HP-UX | Oracle | Linux | MySQL |
| | 1 host (32 cores) Total: $10,400,000 | 32 cores ($550,000/core) Total: $17,600,000 | 1 host Total: $2,000,000 | Total: $0 |
| | $28,000,000 | | $2,000,000 | |
| Unit: NT | | | | |

## VI. CONCLUSION

How to reduce software development and construction costs for a large-scale information system is very important. According to the purpose of the system, the different software solutions are adopted. Therefore, it plays an important role to develop the system across different application servers and database servers. The paper used AAMS as an example, proposed some common principles of system migrations and system design rules. For the most software solutions adopted by enterprises, including commercial support or open source application servers and relational databases, it can reduce the time waste and lower the system construction cost significantly during system migration through the principles proposed by this paper. In the future, it will be more application servers and relational databases added to raise the migration flexibility, and make the common principles of migration more robust.

## REFERENCES


[1] M. Torchiano, M. D. Penta, F. Ricca, A. D. Lucia and F. Lanubile, "Software Migration Projects in Italian Industry: Preliminary Results from a State of the Practice Survey," International Conference on Automated Software Engineering, pp. 35-42, Sept. 2008.

[2] D. Auer and H. Dobler, "A Model for the Migration to Object-Oriented Software Development with Special Emphasis on Improvement of Acceptance," International Conference on Technology of Object-Oriented Languages and Systems, pp. 132-143, Nov. 2000.

[3] Z. Zhang, H. Yang, D. Zhou and S. Zhong, "A SOA Based Approach to User-Oriented System Migration," International Conference on Computer and Information Technology (CIT), pp. 1486-1491, July 2010.

[4] A. Umar and A. Zordan, "Reengineering for Service Oriented Architectures: A Strategic Decision Model for Integration versus Migration," The Journal of Systems and Software, pp. 448-462, 2009.

[5] J. Ziemann, K. Leyking, T. Kahl and W. Dirk, "Enterprise Model Driven Migration from Legacy to SOA," Software Reengineering and Services Workshop, 2006.

[6] A. I. Kistijantoro, G. Morgan, S. K. Shrivastava and M. C. Little, "Enhancing an Application Server to Support Available Components," IEEE Transactions on Software Engineering, pp. 531-545, July 2008.

[7] M. Elamparithi, "Database Migration Tool (DMT) - Accomplishments & Future Directions," International Conference on Communication and Computational Intelligence (INCOCCI), pp. 481-485, Dec. 2010.

[8] L. Xing and Y. Li, "Design and Applications of Data Migration System in Hetertogeneous Database," International Forum on Information Technology and Applications (IFITA), pp. 192-195, July 2010.