Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN EVOLUTION SCHEME FOR BUSINESS RULE BASED LEGACY SYSTEMS

[1]HAN LI, [1]HE GUO, [2]HUI GUAN, [3]XIN FENG, [4]YANG XU, [4]HONGJI YANG

[1] Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, Liaoning, China

[2] Department of Computer Science and Technology, Shenyang University of Chemical Technology, Shenyang 110142, Liaoning, China

[3] People's Liberation Army 66444, Beijing 100042, Beijing, China

[4] Software Technology Research Laboratory, De Montfort University, Leicester LE1 9BH, England

ABSTRACT

Business rules are compact statements that depict important aspects of business processes. For most enterprises, business rules are embedded in the information systems. As change is inherent in software, information systems turn into legacy ones and their documentations may not reflect the actual business logics. Thus, business rules in legacy systems become significant investments, and it is necessary to evolve legacy systems without simply getting rid of the embedded business rules. This paper studies the scheme of business rule-based legacy systems evolution. To locate valuable functionalities, reengineering techniques are used to comprehend legacy system. Then business rules are extracted from these functionalities by means of information flow analysis and decomposition slicing. Since service-oriented architecture is flexible to reuse components and connect components with changes in business, business rules are stored and managed by a service-oriented business rule management system. A case study is illustrated to show the scheme can preserve valuable business rules, facilitate evolving business rule-based legacy systems, and involve non-technical users to business related software evolution.

Keywords: *Business Rule, Legacy System, Evolution, Service-Oriented, Reengineering*

1. INTRODUCTION

Since software is in a constant flux, many information systems become legacy ones. Although most legacy systems are implemented with obsolete techniques, they are strongly intervened with existing business processes, and tightly coupled to business logics which results from many years of patching and fixing. Thus, enterprises are facing an unenviable dilemma. On one hand, they want to get rid of legacy systems in order to align their information systems with new techniques and business logics. On the other hand, legacy systems are important enterprise assets which include business knowledge, such as enterprise data and policies. So legacy systems cannot be simply replaced, and the key lies in the discovery, application and evolution of the embedded business rules.

A business rule refers to the logic that governs business operation. By capturing legacy business rules, enterprises are able to create systems fully aligned with their actual business requirements, and business professionals may be involved to supervise the process of legacy system evolution.

Recent years, many researchers focus on the research of business rules of legacy systems. Based on program transformation, [1] achieved business rule abstraction and organizing, and made legacy CRM system more flexible to cope with further changing requirements. In [2], a framework was given to recover business rules from large legacy systems by integrating the advantages of various normal recovery solutions. In [3], a tailored solution approach which extracted prime business rules from large legacy system was proposed and implemented as a system. Moreover, [4] presented a method to obtain business rules based on dependence-cache slicing. Experiments showed the usage of dependency-cache slicing might improve the precision of business rule abstraction. Since Service-Oriented Architecture (SOA) holds many technology and business benefits, migrating software systems towards SOA attracts more attentions. In [5], the transformation of functionality of CRM system into web services was investigated. In [6], a case study was given to evolve legacy system towards SOA, which provided a practical guideline for legacy system evolution. Based on wrapping technique, a black-box modernization approach was depicted to

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

transform legacy interactive functionality to services [7]. Additionally, [8] developed a sequence mining algorithm in order to identify hidden logics in e-learning legacy systems.

As mentioned above, this paper focuses on business rules of legacy systems, and aims at evolving business rule-based legacy systems towards SOA. Besides, since most software systems or applications adopt object-oriented technique, object-oriented systems are considered as the research object.

The remainder of the paper is organized as follows: Section 2 overviews the proposed scheme. Section 3 depicts the process of the proposed scheme including system comprehension and partition, business rule extraction and service-oriented migration. In Section 4, a realistic business rule-based legacy system is used as a case study to show the feasibility of the proposed scheme. Section 5 draws the conclusions.

2. SCHEME OVERVIEW

The objective of this research is to extract reusable legacy assets from the underlying legacy system and re-host them in service-oriented architecture. Based on reengineering techniques and domain knowledge, a scheme is proposed for business rule centered legacy system evolution. Figure 1 illustrates the framework of the proposed scheme, which is linked by an evolution process model and comprises 5 main modules as follows:

System Comprehension: It contains the methods of understanding legacy systems including pre-processing, static analysis and dynamic analysis.

System Partition: It includes the methods of dividing legacy systems into subsystems. It is a preparation for extracting business rules.

Business Rule Extraction: It is the kernel of the proposed scheme, which comprises the methods of locating and generating business rules.

Service-Oriented Migration: It involves the methods of representing business rules and deploying a service-oriented business rule management system.

Domain Knowledge Base: It comprises a set of design patterns, architectural patterns and expert knowledge of particular domains. It supports the execution of other modules.

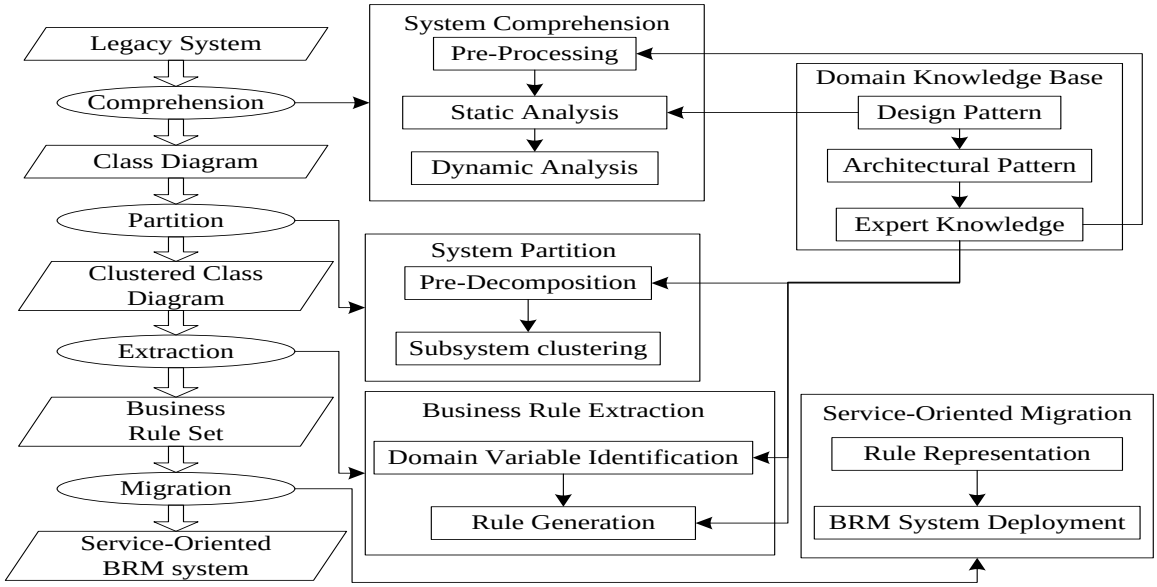

Figure 1: Framework Of The Proposed Scheme

As shown in Figure 1, the evolution process model is a 4-phase process as follows:

Comprehension: It is concerned with legacy system understanding. It outputs a class diagram which primarily target on valuable legacy functionalities that may be considered for inclusion into the target system.

Partition: It takes charge of decomposing class diagrams into different subsystems. The outputs are clustered class diagram.

Extraction: It carries out extracting business rules from corresponding subsystems, which results in a group of business rules in business rule syntax [1].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Migration: It is responsible for integrating web service technique with business rules. The output is a service-oriented BRM system.

3. BUSINESS RULE-BASED LEGACY SYSTEM EVOLUTION

Business rules are the kernel of business rule-based legacy systems. So identifying business rules and achieving target systems based on these existing business rules are significant procedures during legacy system evolution.

3.1 System Comprehension

Since most legacy systems have been developed over a long time and the business rules are spread across the whole system, system comprehension is considered as a prerequisite for legacy system evolution.

Pre-Processing aims at identifying and documenting which legacy functionalities are compatible with the requirements of the target system. Except for source code and technical documents, expert knowledge is required to specify these useful parts. Pre-processing comprises 2 parts including information collection and functionality selection. Information collection is to specify resources and understand business context. It contains a list of relevant activities such as skimming technical documents, studying use cases, discussing with maintainers and reviewing source code. Functionality selection is to identify reusable functionalities. It is iterative and involves business and technical knowledge. Business experts recognize domain concepts which are candidates for target business services. Software engineers judge which designs and implementations are candidates for corresponding business concepts. As a result, functionalities which are potential inputs to the target system are highlighted and classes that related to the same functionality are regarded as a component.

Static Analysis is a possible and suitable technique to compute the potential role a class plays in a design pattern, because classes always match against the design patterns proposed by system designers or developers in Object-Oriented development. Thus, it is applied to obtain the relationships among classes within a component. The process is summarized as 4 steps:

Step1. Design Pattern Instance Determination

A set of design pattern instances are provided for specific reusable functionalities.

Step2. Initial Class Diagram Allocation

According to the instance of a design pattern, an initial class diagram is developed for what is expected in the source code.

Step3. Design Pattern Refinement

Enumerating element names in the class diagram and refine them by searching them in the source code. Renaming, remodeling, extending and replacing are the activities to refine class diagram.

Step4. Actual Class Diagram Detection

Taking the refined class diagram as the declaration form, all classes and their methods are detected to generate candidate class diagrams that match the refined design pattern for a specific functionality.

Dynamic analysis provides a mechanism for Component and Connector view which illustrates runtime behavior among various functionalities. It only retrieves actual interacted components. Therefore, the dynamic relationships of components are captured and recorded. Following depicts the steps of dynamic analysis.

Step1. Task Scenario Selection

In terms of the occurrences of classes and interfaces in use cases, a set of common task scenarios are selected.

Step2. Components and Connector Aggregation

Observing which procedures are executed when a specific task scenario is executed and further tracks its chain reaction to discover how components are interacted at run-time.

Step3. Sequence Diagram Generation

The dynamic relationships among components are modeled using component-level sequence diagrams.

3.2 System Partition

As legacy systems are always huge in size, it is useful to break a complex system down into a set of small and manageable subsystems which respectively contains relevant components.

Pre-Decomposition: Legacy systems always contain a large number of cooperating components, and these components are often organized into identifiable clusters, namely, subsystems. That is, components which contribute to the same business logic are included in a subsystem. Hence, according to domain-expert knowledge, a legacy system is preliminarily decomposed based on the discrepancy of business logic that embedded in various subsystems. Figure 2 shows the structure of a legacy system after pre-decomposition.

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

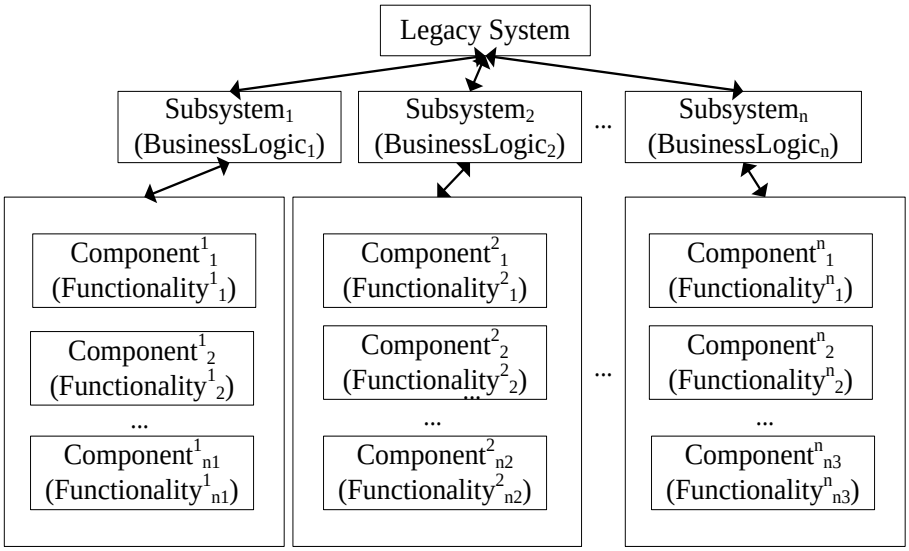

Figure 2: Structure Of A Legacy System After Pre-Decomposition

Subsystem Clustering: Partition clustering starts with an initial set of cluster centers, and relocates entities by moving them from one cluster to another. It is a proper method to optimize the subsystems generated by pre-decomposition. In order to efficiently explore the extraordinary large solution space of all possible partitions, Genetic Algorithm (GA) based clustering is applied. The inputs of the clustering algorithm are preliminary subsystems and the dependency graph of all classes contained in these subsystems. Following represents the main steps of GA-based clustering algorithm.

Step1. Fitness Function Selection

Fitness function is used to quantify the quality of a cluster. Considering both intra and inter connectivities among classes, modularization quality (MQ) is selected as the fitness function which is shown in Eq. (1), where $A_i=\mu_i/N_i^2$ is the intra-connectivity of subsystem i which has N_i classes and μ_i intra-edge dependencies, and $E_{i,j}=\varepsilon_{i,j}/2N_iN_j$ ($i\neq j$) and $E_{i,j}=0$ ($i=j$) depicts the inter-connectivity between subsystem i and j which respectively contains N_i and N_j classes with $\varepsilon_{i,j}$ inter-edge dependencies.

$$MQ=\begin{cases}\dfrac{\sum_{i=1}^{k}A_i}{k}-\dfrac{\sum_{i,j=1}^{k}E_{i,j}}{\frac{k(k-1)}{2}} & k\geq 1\\ A_1 & k=1\end{cases} \quad (1)$$

Step2. Encoding Scheme Selection

In a chromosome, the value of a gene represents the index of a subsystem that the class belongs to. The position of a gene depicts the index of a class.

Step3. Population Initialization

In order to improve the efficiency of GA, domain knowledge is taken as a reference. That is, compared with randomly generated population, the value of a gene considers the result of pre-decomposition.

Step4. Genetic Operator Execution

The execution comprises selection, crossover, mutation and replacement. Tournament selection strategy is used to choose parental members for reproducing population. Classes that depend on each other in a chromosome are passed together to the offspring during crossover. Mutation only happens when the offspring improves the quality of the chromosome. MQ is utilized to determine which chromosomes should be replaced by new ones.

Step5. Termination of Clustering

The terminating conditions satisfy when the genetic operators result in specified number of generations or the value of the fitness function of the best chromosome remains the same.

3.3 Business Rule Extraction

Business rules which are stated as statements that define or constrain some aspect of the business [9] are the kernel investments for enterprises. Within each subsystem, business rules are extracted as follows:

Domain Variable Identification: Since business rules always creating, maintaining and utilizing data, and data are primarily represented by variables in source code, variables that relates with business rules becomes the clue of business rule extraction. Business rule related variables are defined as domain variables comprising pure and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

derived domain variables. Pure domain variables directly map to objects in source code. Derived domain variables depend on other domain variables. In order to identify domain variables information-flow algorithm is applied to each methods in a class. First, input, output and variables that apparently map to objects of source code are considered as initial domain variables. Next, other domain variables are computed by relation ρ in information-flow algorithm [10].

Rule Generation: Although legacy systems are partitioned into corresponding subsystems. It is still complex to locate business rules within a subsystem. Thus, according to the domain variables identified, decomposition slicing is performed to carve out the minimal subset of program statements without affecting the implied business logics. Decomposition slicing does not depend on statement numbers. It produces a set of slices which contains all the identified domain variables. Based on the slices, business rules are generated based on business rule syntax [1].

3.4 Service-Oriented Migration

A Business Rule Management System (BRMS) is a software system used to define, deploy, execute, monitor and maintain the variety and complexity of decision logics used by operational systems within an organization. Adopting BRMS in information systems can decrease development cost and shorten the cycle of development and maintenance. So the extracted business rules are integrated into BRMS. Moreover, in order to easily deploy and integrate new business requirements, Web Service is applied to construct BRMS. That is, components in the BRMS are implemented as service suppliers or consumers.

Rule Representation: Business rules represented in business rule syntax is similar to natural language. As natural language is difficult to be directly converted into standard XML format, Entity Relationship Diagram (ERD) is involved as an interpreter for the conversion. The process of rule representation contains 2 steps.

Step1. ERD Conversion

ERD is a graphic that shows the relationships among entities in a database. It is intuitive and easy to be comprehended by both engineers and business professionals. The Business Rules Group studied the relationship between entity relationship models and business rules. Based on the method which converts business rules into entity relationship models [9], business rules are transformed into a set of database tables which are visually represented by ERD.

Step2. XML Conversion

As the target system is service-oriented, business rules represented by ERD requires to be converted into XML format. RuleML [11] is an open language standard for rule interchange and markup. It bases on XML and is commonly used in Web Service. Thus, it is applied to represent business rules. According to the types of business rules, various elements in entity relationship model are mapped to corresponding elements in RuleML.

Service-Oriented BRMS Deployment: Business logics are separated from abundant and complex legacy functionalities. In order to avoid mussily distributing business logics in legacy systems and facilitate further business rule evolution, the extracted business rules are stored and deployed in a service-oriented BRMS. Figure 3 shows the architecture of the service-oriented BRMS.

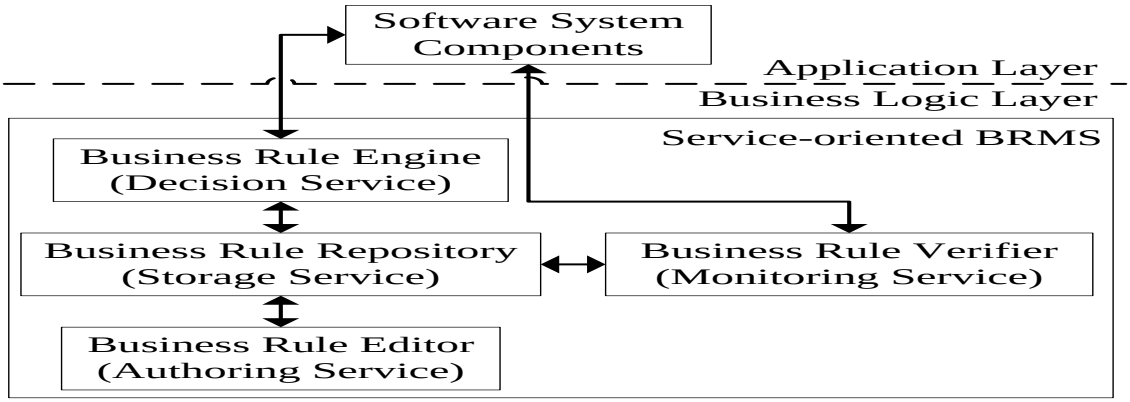

Figure 3: Architecture Of The Service-Oriented BRMS

For the target system, the entire service-oriented BRMS is considered as an independent service which provides business logic service for components in application layer. Inside the service-oriented BRMS, its 4 modules are respectively made available as different rule services which loosely couple with each other.

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. CASE STUDY

In order to ensure the feasibility of the proposed scheme, the early version of a Customer Relationship Management (CRM) system, which manages the customers of an online shopping system, is chosen as the case study. For online shopping systems, the success of a retailer tightly relates to the quality of its customer care. Thus, the CRM system must contain a set of business rules used to classify and serve customers. The CRM system is a web-based legacy system which is developed by ASP and Microsoft Access.

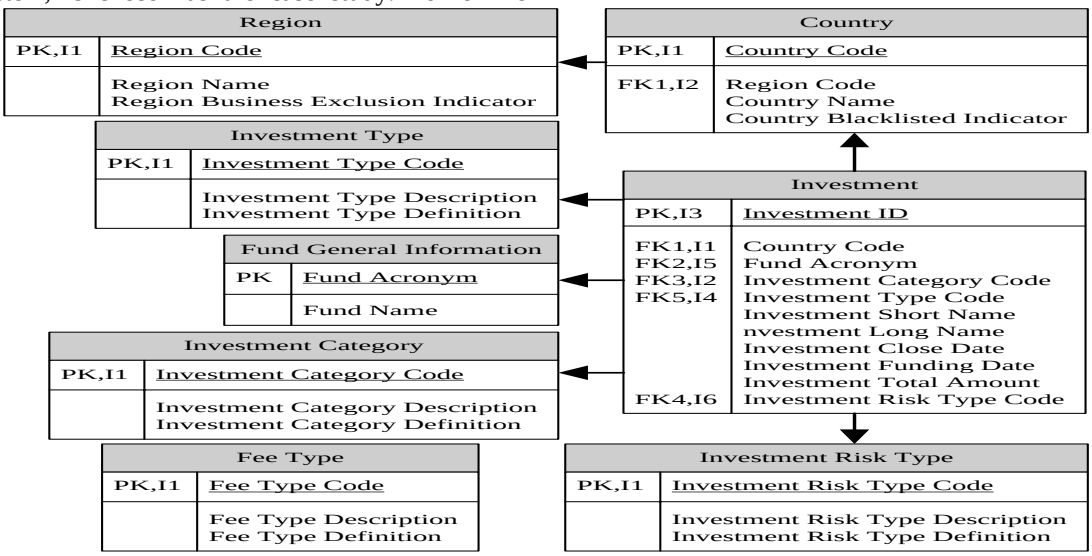

Figure 4: Partial Logical View Of The Case Study

During system comprehension, pre-processing firstly chooses Access database during information collection and analyzes tables in Access database. Figure 4 shows partial logical view of the case study, which contains tables that possess investment relevant data. In addition, each arrow in Figure 4 represents a 1:*n* relationship.

Then, a RFM (Recency, Frequency and Monetary) analysis method is applied to comprehend the relationships among tables and data stored in the tables, and with the help of business professionals customers are classified into 6 groups consisting of superstar, golden customer, typical customer, exceptional occasions customer, everyday shopper and dormant customer. Taking superstar as an example, it includes customers who are the most loyal and hold highest frequency and investment.

Since ASP codes separate representation and logic processing, it is easy to locate description codes and concentrate on business logics by identifying variable handling and business relevant codes. Then the relationships among these business logic relevant codes are captured by static and dynamic analysis. As the CRM system relatively simple, system partition is omitted. That is, the entire CRM system is considered as one subsystem, and the next step is to identify business rules. For instance, when considering variable *dct* which maps to object discount as a domain variable, a business rule, which means if the purchase of an order is over 1000, a discount is given to the order and the discount rate is 5%, is captured as shown in Figure 5.

```
ON (discount)
IF (order exists) AND (order.purchase>1000)
THEN discountrate=5%
```

Figure 5: An Example Of Business Rule

In order to deploy the extracted business rules in the service-oriented BRMS, business rules are transformed into XML format with the help of ERD. In the case study, 17 tables are established to express the entity relationship model. For example, table Rule Enforcement is used to record the type, starting and ending date of rules. As shown in Figure 6 the business rule in Figure 5 is represented in RuleML.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

```
<Implies>
 <if>
  <And>
   <Atom>
    <Rel>exist</Rel>
    <Var>order</Var>
   </Atom>
   <Atom>
    <Rel>purchase</Rel>
    <Var>order</Var>
    <Ind>over 1000</Var>
   </Atom>
  </And>
 </if>
 <then>
  <Atom>
   <Rel>discountrate</Rel>
   <Var>order</Var>
   <Data>5.0 percent</Data>
  </Atom>
 </then>
</Implies>
```

Figure 6: The Sample Business Rule In Ruleml

At last, these business rules in XML format are stored in business rule repository of the service-oriented BRMS and serves as services. Based on the service-oriented BRMS, business rules can be evolved according to further business requirements. In terms of these evolved business rules, corresponding business services are able to be implemented by wrapping corresponding legacy codes or developing new services.

5. CONCLUSION

This paper presents a scheme to assist the evolution of business rule-based legacy systems. To concentrate on valuable legacy functionalities, legacy systems are understood and partitioned to subsystems by applying reengineering techniques. Information-flow analysis and decomposition slicing are applied to identify domain variables and business rule relevant code segments. In terms of domain expert knowledge, implied business rules are captured and are deployed in a service-oriented BRMS. The feasibility of the scheme is proved by a simple case study. In brief, the scheme is able to assist business rule-based legacy system evolution. By applying software reengineering and knowledge from business experts, it not only facilitates the identification of valuable functionality and business rules, but also gives an opportunity for non-technical business users to take part in the evolution of legacy systems. Furthermore, by integrating service-oriented architecture and business rule management with legacy system evolution, business rule-based legacy system can be organized by a flexible architecture which may facilitate further evolution.

REFERENCES:

[1] Y. Xu, H. Yang, and I. Amin, "Business Rule Based Program Transformation for CRM System Evolution", Proceedings of IEEE International Conference on Information Reuse and Integration, Institute of Electrical and Electronics Engineers Computer Society, September 16-18, 2006, pp. 244-247.

[2] X. Wang, J. Sun, X. Yang, and Z. He, "A Framework of Business Rules Recovery from Large Legacy Systems", *WSEAS Transactions on Information Science and Applications*, Vol. 3, No. 3, 2006, pp. 576-583.

[3] C. Wang, Y. Zhou, and J. Chen, "Extracting Prime Business Rules from Large Legacy System", Proceedings of International Conference on Computer Science and Software Engineering, IEEE Computer Society, December 12-14, 2008, pp.19-23.

[4] G. Xie, "Business Rule Extraction from Legacy System Using Dependence-Cache Slicing", Proceedings of International Conference on Information Science and Engineering, IEEE Computer Society, December 26-28, 2009, pp. 4214-4218.

[5] Y. Xu, Q. Duan, and H. Yang, "Web Services Oriented Customer Relationship Management System Evolution", Proceedings of the 13th IEEE International Workshop on Software Technology and Engineering Practice, Institute of Electrical and Electronics Engineers Computer Society, September 24-25, 2005, pp. 39-48.

[6] F. Cuadrado, B. García, J. C. Dueñas, and H. A. Parada, "A Case Study on Software Evolution towards Service-Oriented Architecture", Proceedings of the 22nd International Conference on Advanced Information Networking and Applications, Institute of Electrical and Electronics Engineers Computer Society, March 25-28, 2008, pp. 1399-1404.

[7] G. Canfora, A. R. Fasolino, G. Frattolillo, and P. Tramontana, "A Wrapping Approach for Migrating Legacy System Interactive Functionalities to Service Oriented Architectures", *Journal of Systems and Software*, Vol. 81, No. 4, 2008, pp. 463-480.

[8] Z. Zhang, D. Zhou, H. Yang, and S. Zhong, "A Service Composition Approach Based on Sequence Mining for Migrating E-Learning Legacy System to SOA", *International Journal of Automation and Computing*, Vol. 7, No. 4, 2010, pp. 584-595.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[9] The Business Rules Group, "Definiting Business Rules-What Are They Really?", Report, 2000.

[10] X. Wang, J. Sun, X. Yang, Z. He, and S. R. Maddineni, "Application of Information-flow Relations Algorithm on Extracting Business Rules from the Legacy Code", Proceedings of the 5th World Congress on Intelligent Control and Automation, Institute of Electrical and Electronics Engineers Inc., June 15-19, 2004, pp. 3055-3058.

[11] J. A. Randall, W. Duminda, and B. M. James, "Using RuleML to Specify Cross-Domain Information Flow Control Policies", Proceedings of IEEE International Conference on System of Systems Engineering, IEEE Computer Society, May 30-June 3, 2009, pp. 1-6.