Business Rule Extraction from COBOL applications

Enterprise Applications typically contain a substantial amount of Business Knowledge, which has been constructed, enhanced and tuned over many years. Typically such Legacy Enterprise Applications have little or no up-to-date documentation, and original authors and designers are no longer able or available to re-document the system.

Manual re-documentation of such code is often a lengthy, expensive and error-prone process. Re-engineering these applications from an incomplete or inaccurate understanding has obvious pitfalls.

Softwaremining’s BRE Toolkit uses sophisticated search algorithms to sieve through complex program code and isolate the Business Rules embedded within it. The system allows assignment of descriptions in natural language to the set of variables referenced within the identified rule. This feature can be used to reproduce Business Rules in a Natural Language form (i.e. pseudo-English).

One of the main features of the technology is the ability to apply the unique softwaremining Language Translation Technology to the extracted Business Rules. The Rules can thus be exported as Java, C# or COBOL code snippets, ready for incorporation onto newly built applications.

Features

- Application Overview and Code Inventory – reports and ease of navigation through flowcharting
- Program level flowcharts
- Heuristics assisted identification of important “rule” variables
- Definition Rules – automatic extraction
- English Rules – Translation of COBOL rules from COBOL to Pseudo-English
- Java and C# Rules: translated program components can be used directly in a modernized application.
- Data Dependencies - Identification and visual representation
- HTML Documentation - Business Rules, program level and application level documentation in HTML.
Code Inventory & Documentation

The system enables easier maintenance of the Application, providing easy access to information such as:

• Identifying entry points to the applications (Root Programs)
• Identifying dependencies between program modules
• Identifying dependencies on copy files
• Identifying data-structures passed between calls, and how are they transformed.

The main screen in the BRE Toolkit provides an overview of the application, including:

• A list of all programs, which copy files they reference and what other programs they call;
• Copy Files - which programs references them;
• Inter-program flowcharts;
• Facility to navigate between programs

The program flowcharts and Data-Division hierarchies are appropriately documented. The documentation can then be saved in HTML/SVG format.

Data View Page

BRE Toolkit allows inspection of the system from “Data” point of view. It shows the hierarchical view of the structures, identifies programs referencing each structure, as well as how the programs interlink. Other useful features include searching for: most used field (can be used as a starting point for business-rules), unused fields (to be removed), large number fields (candidates for BigDecimal / BigInteger when rewriting in Java). COMP fields (extra-care required during data-migration).
**Heuristics – Rule Extraction starting point**

Softwaremining Business Rule Extraction provides a mean of identifying the operations and relationships associated with “Variables-Of-Interest” (VOI).

The identification of VOI is not a trivial task within large and complex legacy system. The original application developers usually can provide a good starting point, the subject matter expert will be also be invaluable. However, often such resources have long left the organization, and other approaches will be required for identification of VOIs.

The “Heuristic Search” screen facilitates identification of VOIs, by providing “usage patterns” searches.

For example, a simple “usage pattern” is to identify variables used in mathematic operations (COMPUTE, ADD, SUBTRACT, …) as the result of (financial) calculations are often important.

A slightly more complex pattern would be to identify variables involved in Mathematics, displayed on screen (ACCEPT / DISPLAY / CICS SEND / RECEIVE), and Read or Stored to files/database. Another pattern may include variables occurring in Mathematics and Reporting.

The “Heuristic Search” screen facilitates such searches.

On successful identification of any such variables, a “rule” template is build, ready for the next “extraction” stage.
**Rule Composer Page**

The rule composer page provides a means to organize, extract and document Rules associated with the identified Variables of interest.

Here rules can be categorized, and processed. Rules have the following characteristics:

- List of applicable programs
- Indicate for whether they apply entire application database, or merely the programs that were identified at rule creation time
- Description – what business processes is achieved (to be filled in by Subject-matter-expert).
- Date-Time created/ Exported.

Additionally the system can be configured to include / ignore certain statement types. For example, the inclusion of file/database read/write statements may not be important in a rule for calculation of an Order-Value, and such statements may be excluded during rule extraction.
Intelligent Extraction of Definition Rules

Definition rules can be defined as statements which provide a definition of a term (variable-of-interest) and a quantitative relationship between it and other variables. The terms in question are those responsible for achieving a particular business objective. E.g.

“order value is calculated by multiplying price of an item by quantity ordered”.

Given a variable-of-interest, the BRE Toolkit can automatically find all the relevant statements and which goes into definition of this rule.

For example, given the following code:

```
01        MOVE 123 TO ITEM-PRICE.
02        MOVE 10 TO I.
03        COMPUTE TEMP = 1 * J.
04        MOVE 40 TO J.
05        ACCEPT ORD-QTY.
06        COMPUTE TEMP = ITEM-PRICE * ORD-QTY.
07        MOVE TEMP TO ORD-VAL.
```

And looking for Definition rules for ORD-VAL (order value) the system can automatically derive the following statement as relevant:

```
01        MOVE 123 TO ITEM-PRICE.
02        MOVE 10 TO I.
03        COMPUTE TEMP = 1 * J.
04        MOVE 40 TO J.
05        ACCEPT ORD-QTY.
06        COMPUTE TEMP = ITEM-PRICE * ORD-QTY.
07        MOVE TEMP TO ORD-VAL.
```

Considering typical COBOL programs contain thousands of lines of code, such an automatic process can save significant amount of time on any business-rule extraction and re-documentation project.

The rule-definition facility traces application flow-logic backwards through all the possible routes, to identify all the most relevant operations on all variables involved in setting the value of Variable-of-Interest (ORD-VAL in above example).

Next, using the in-built Natural-Language translation facility, the above statements can be turned automatically into:

```
Item Price = 123
read Order Quantity. (from screen)
Temporary Variable = Item Price * Order Quantity.
Order Value = Temporary Variable
```
**Business Rule Extraction Page**

Most COBOL programs perform a variety of tasks. Eg. An Order-Processing program may calculate invoice amounts, update inventory records, have data-entry screens etc. SoftwareMining’s Business Rule Extraction Strategy works by providing Data and Statement-Category filters, and Advanced Search Algorithms to extract the relevant information from the programs.

All the Business Analyst / developers have to do, to start the business rule extraction process, is to:

- Specify a Variables of Interest; telling the system to find all explicit / implicit (Thru redefines, and hierarchies) references to the Variable of Interest.
- Select Statement filters; for example, and a rule interested in calculation of Order-Values may not be interested in screen-handling, database handling, statements.

The application of the above filters results in identification of code addressing Variable of Interest, as well as all the other variables referenced within the listed code segment.
The following code snippet is produced as a result of business rule extraction on “Order Value” (variable ORD-VAL) on COBOL program:
Translation of COBOL Rules to Natural Language

The BRE Toolkit also facilitates the association of Natural Language Description with variables names. The descriptions are used, together with in-built heuristic based Natural Language substitution algorithms, to generate specifications in Pseudo-English from the extracted business rule.

Using Acronyms for automatic generation of Description

To ease the generation of variable descriptions, Softwaremining’s BRE Toolkit can scan through all the variable names and export the set of common acronyms used throughout the application. A subject-matter expert can then work off-line to describe what each of the acronyms mean. For example, DOB means Date Of Birth, VAL means VALUE and …

The description list can then be imported and used to automatically generate descriptions for all the variables.
Translation of COBOL Rules to Java Rules

Often the aim of Business Rule Extraction is to produce specification for the rewrite of an application in Java or C#. Softwaremining’s COBOL to Java Translation system is an integrated part of the Business-Rule-Extraction Toolset. The translation module can be used to produce the business rules in **Maintainable and Legible** Java or C# code, together with all XML definitions for data-bears, Object-Relational Persistence Beans and the underlying framework.

### Business Rule Code

```java
/**
 * calcDiscount
 */

protected void calcDiscount (){  
    if (orderMaster.getOrdVal() > iData.getDisThrshld() ||  
         orderMaster.getFirstTimeClient().equals(orderMaster.firstTimeClientClntIsNew))  
    {  
        orderMaster.setOrdVal(orderMaster.getOrdVal()  
                               * iData.getWsDisctPrctnt());
    }

    if (clientInfo.getLocalClnt().equals(clientInfo.localClntClntIsLocal))  
    {  
        if (orderMaster.getDeliveryOption().equals(orderMaster.deliveryOptionNxtDayDlvry))  
        {  
            orderMaster.setOrdVal(orderMaster.getOrdVal()  
                                  + iData.getStdLclDelvChrg());
        }  
        else  
        {  
            orderMaster.setOrdVal(orderMaster.getOrdVal()  
                                  + iData.getNxtDayLclDelvChrg());
        }
    }  
    else  
    {  
        orderMaster.setOrdVal(orderMaster.getOrdVal()  
                              + iData.getOvrseaDelvChrg());
    }
}
```