International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 1 Issue 9, November- 2012

An Approach To Develop Invoice Validation Framework

Prof. M. Emmanuel
Pune Institute Of Computer Technology

Sucheta Thakur
Pune Institute Of Computer Technology

Abstract

Invoice processing is becoming complex day by day .It is quite difficult and time consuming to check these complex codes. Invoice format changes over a period of time affecting an existing functionality of a product. Currently, no automated way is present for validating an existing rule set and newly implemented rule sets which can compare invoices from raw data files to what is loaded into a system. Validating an invoice is an important activity carried out during invoice processing. In today's scenario, there are some invoices which are still error prone due to its complex nature. This paper presents a novel approach to develop a framework for analyzing regression in processing an invoice.

Index Terms— Regression Testing, JUnit, Validation

1. Introduction

Validation is one of the essential in software testing. Manual testing is usually a hard and time consuming and expensive activity. Testing is considered as one of the most costly development processes, sometimes exceeding fifty per cent of total development costs [4]. Most of the times, testing is often poorly performed or skipped by practitioners. This situation leads to industry wide deficiency in testing, and motivates for automated testing as one of the possible solution to overcome this problem [3].

The process of developing large software systems is a complex and error prone process. If fault occurred at an early stage of software development, is identified and removed, it will reduce the validation cost [6]. Automated regression testing is an effective way of doing regression test run after each bug fix [12].

The paper is arranged in following manner. Section I gives necessity of validation in software engineering. Section II gives literature survey of an invoice and its validation approach. Section III defines proposed work followed by problem description. Section IV gives conclusion.

2. Literature survey

In [7], validation of model based approach and exploring automatic model validation techniques has been discussed.

In [8], various test selection techniques for analyzing regression has been mentioned. Rothermel and Harrold described regression testing techniques as: "Given a program P, a modified version P', and a set T of test cases used previously to test P, regression analysis and testing techniques attempt to make use of a subset of T to gain sufficient confidence in the correctness of P' with respect to behaviours from P retained in P'."

There are some applications where lots of transaction takes place at backend and no human interface is present to test those transactions. A framework is proposed that can easily be used to complete an end to end testing process [9]. Customer's loyalty should be retained in today's world. Invoice processing should be correct and there should not be any disruption during processing. Though the formats change, yet it is necessary to ensure stability of an application when it runs in production environment [9].

Invoice contains a list of charges for services or products rendered. They are used in all places where services or products are provided. In some environments invoices are delivered right away whereas in some cases it involves different methods of payments [11]. An electronic invoice is a transaction document containing billing information in an electronic format. They are the evolution of traditional paper invoices [2].

There are still certain errors while generating telecom bills [1]. One of the factors is the size and structure of invoices which is very complex. There are other reasons that cause errors in invoice processing. Some of the factors that cause billing errors can be multiple charges for the same call. Service charges based on volume might not be applied correctly. Some charges may vary from city to city, so taxes may be applied incorrectly. According to a survey, telecommunications expenses rank as one of the five expenses for most companies. Though it has evolved from decades, yet it lacks data validation [1].

In [2] importance of accounting validations has been mentioned. Accounting validations is validations of several amount, quantities mentioned in an invoice. Invoice can be represented in any format i.e. Electronic Data Interchange, text file or xml file, depending on the vendor .Author has mentioned the processing of an invoice in xml format. In [14] author gives validation of the document by creating DTDs and validated it against the schema. The basic structure of Electronic Data Interchange is described in [10].

Regression testing is applied at unit, integration and system levels to reveal different types of failures. During software development life cycle, unit testing, integration testing, and system testing are the types of software testing applied. Unit testing is a process of testing each software module. Most existing regression testing techniques focus on unit testing [15].

JUnit provides two methods, setUp method and tearDown method. The setUp method creates objects and performs tasks needed to run a test and the teardown method destroys the fixture. JUnit automatically invokes the setUp and tearDown methods before and after each test method is executed. A test suite is a collection of test methods which run tests. JUnit invoke the method suite and runs each test method in the suite. A test suite can contain several test methods.

3. Proposed work

In this paper, the framework proposed will validate an invoice. Whenever an invoice format changes, there can be a modification in the working of previous functionality. There may be a case that an accounting data may not persist in a database correctly thus leading to wrong calculations.

3.1 Problem Description

The system is defined as:

System S = {I, O, F} where,

I: Input set,

O: output set,

F: Function set

Set I = {IF, X} where 'I' is an input set which consists of an invoice load file and an xml file containing an expected values.

IF =Invoice file

X=xml file where

X = {S1………Sn} where 'S' is the scenarios to be validated 'n' is number of scenarios.

Each S = {e, q, m} where 'e' is an expected result, 'q' is query and 'm' is message.

XML file will contain a set of different scenarios which will validate against an actual database value.

O = {VR}

O is an output set which consists of a validation result. Validation results will be either true or false. In case, an actual value matches with an expected value, VR = T.

VR = {T, F} is validation result

F is a set of function.

F = {α, β, γ, δ} where

Svg = α(X) is α schema value generator.

Avg = β (q) is β actual value generator.

Evg = δ (e) is δ expected value generator.

Vr = γ (Avg, Evg) is γ is validation result.

Let R be a set.

R= {T, F} where 'R' is result set which will give a value either True or False

Let I be a set.

I= {I1, I2………Ij} is invoice format where 'I' is an invoice format.

Assuming, there are 'j' different kinds of formats, relationship between 'R' and 'I' sets is: I ϵ R

Since an EDI is divided into header, summary and detail,

I1= {H1, S1, D1}; I2= {H2, S2, D2}…….. Ij = {Hj, Sj, Dj}

Header part H1= {h11, h12, h13… h1l} of invoice 'I1' consists of 'l' number of header fields.

Summary part S1= {s11, s12……………. s1m} where S1 consists of 'm' number of summary fields.

Detail part D1= {d11, d12, d13……….d1n} where D1 consists of 'n' number of summary fields. 58

The description can be represented in logical terms as:

$$H_1 = T \ni \forall h_{1x} = T \quad \text{where } 1<=x<=l…………(1)$$

'l' represents number of header fields.

$$S_1 = T \ni \forall s_{1y} = T \quad \text{where } 1<=y<=m…………(2)$$

'm' represents number of summary fields.

$$D_1 = T \ni \forall d_{1z} = T \quad \text{where } 1<=z<=n…………..(3)$$

'n' represents number of summary fields to be validated.

From above equations, for every invoice, validation is true if header, summary and detail level of an invoice is true. This can be represented as:

$$I_j = T \ni H_j \wedge S_j \wedge D_j = T \forall h_{1x}, s_{1y}, d_{1z} \in (H_j, S_j, D_j)$$

where 1= < j < = n and 'n' is number of invoice formats considered in test suite.

3.2 Design

The following diagram depicts proposed design of validation model .For validating an invoice, test suite will start, which will load an invoice. After an invoice persists in database, test scenarios will start.XML file is an input to a framework which will extract expected values. After extracting an expected and an actual value, validation will start. If the values are correct then test scenarios will pass.

Invoice in different formats will be considered. Different invoice will have different methods of extracting invoice in database. Test cases will be in JUnit. Setup method load properties, common to all invoices and the teardown method resets all properties.

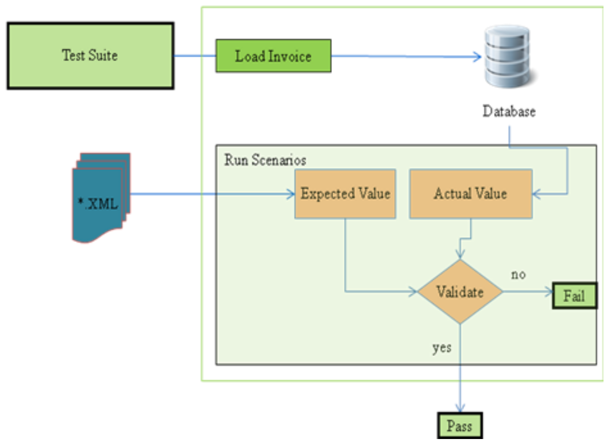

Figure 1. Proposed design of validation model.

3.3 Implementation

An example of file has been presented in fig. 2. The file contains number of elements, forests etc. These are tags recognized by framework and values contained in this field, will be extracted as an expected values, which an invoice should contain while loaded in database. Considering an example in fig 2, the given file will contain header, summary and detail information of an invoice. Since there are lots of data in each segment of an invoice, we will consider those fields first which are prone to errors.

```
<?xml version="1.0"?>
<Assertions>
 <Assert>
      <Description>Validate Header
      </Description>
   <ActualValue> sql query to validate Header
   </ActualValue>
   <CheckMessage PARAM="True">
 <AssertMessage>Incorrect Field name
  </AssertMessage>
    </CheckMessage>
          <ExpectedValue>
              <Rows>
                <Row>
                   <Datafield>expected values
                   </ Datafield >
                  </Row>
              </Rows>
           </ExpectedValue>
     </Assert>
  </Assertions>
```

Fig. 2 Structure of XML file.

Finally, the algorithm mentioned gives out a validation steps. The input to a framework as mentioned above contains header, summary and detail level information.

Algorithm: To validate the results.

```
For i:=1 to AssertList do
Sql[i] = getSql
Fire sql in database and retrieve result 'resultset'
IF(count_of_ resultset>0)
PRINT Description
List_of_expected_value=Get The List of Expected values
For j:=1 to List_of_expected_value.size
List_of_rows=Get The List of rows
For k:=1 to List of Rows.size
List_of_row=Get The List of row
For l:=1 to List_of_row.size
expected value=row.getValue
actual value = resultset
IF  expected value=actual value
PRINT description
Else if CheckMessage  = true
PRINT CheckMessage
END IF


END IF
```

Figure 3. Proposed Algorithm

4. Conclusion

The solution of a problem is proposed with an idea of detecting errors at an early stage of invoice processing. If an error is detected at an early stage of development, then it will reduce the cost of processing invoice. The design of the framework will validate different invoice formats with different rules.

5. References

[1] Dan Parnas & Oleg Test,"How Telecom Invoice Automation Can Reduce Costs". In "Epiphany Supply Chain Solutions, Inc."

[2] Rodrigo García-Carmona, Álvaro Navas, Félix Cuadrado, Boni García,Hugo A. Parada G., Juan C. Dueñas ."A Model-Based Approach for the Management of Electronic Invoices".

[3] Bruce Posey (2002): Just Enough Software Test Automation. Prentice Hall PTR, Upper Saddle River, NJ, USA.

[4] Boris Beizer (1990): Software testing techniques (2nd edition.). Van Nostrand Reinhold Co., New York, NY, USA. M. Young, "The Technical

Writer's Handbook". Mill Valley, CA: University Science, 1989.

[5] Luciano Baresi, Mauro Pezze," An Introduction to Software Testing", Electronic Notes in Theoretical Computer Science 148 (2006) 89–111

[6] Neil Boyette, Vikas Krishna, Savitha Srinivasan," Eclipse Modeling Framework For Document Management".In DocEng'05, November 2–4, 2005, Bristol, United Kingdom.

[7] Gregg Rothermel, Mary Jean Harrold, "Analyzing Regression Test Selection Techniques",In IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 22, NO. 8, AUGUST 1996.

[8] Tuli Nivas." Test Harness and Script Design Principles for Automated Testing of Non-GUI or Web Based Applications". In ETSE'11, July 17, 2011, Toronto, ON, Canada Copyright 2011 ACM 978-1-4503-0808-3/11/05.

[9] Yildiray Kabak,Asuman Dogac," A Survey and Analysis of Electronic Business Document Standards" .In ACM Computing Surveys, Vol. 42, No. 3, Article 11, Publication date: March 2010.

[10] Eduardo B. Fernandez,Xiaohong Yuan,"An Analysis Pattern for Invoice Processing," Approach for the Management of Electronic Invoices".In ISBN 2009,August 28-30,Chicago,IL,USA.

[12] Gerard Meszaros, " Agile Regression Testing Using Record & Playback".In OOPSLA 2003, Oct 26-30, 2003, Anaheim, California.ACM 1-58113-751-6/03/0010.

[13] David C. Yen, Shi-Ming Huang, Shi-Ming Huang, " The impact and implementation of XML on business-to-business commerce".In Computer Standards & Interfaces 24 (2002) 347–362.

[14] B. Bouchou, D. Duarte, M. Halfeld Ferrari Alves D. Laurent,"Extending tree automata to model xml validation under element and attribute constraints".

[15] Yuejian Li,Nancy J.Wahl."An Overview of Regression Testing".In Software Engineering Notes vol 24 no 1.ACM SIGSOFT.January 1999 Page 69.