ONTOLOGY BASED METHODOLOGY TO ENRICH FUNCTIONAL TEST COVERAGE IN SOFTWARE APPLICATIONS

[1]K. S. A. W. GUNAWARDENA, [2]K. Y. BANDARA, [3]N. D. KODIKARA

[1]Manager Industrial Training, Faculty of Computing, SLIIT, Malabe, Sri Lanka
[2]Head, School of Postgraduate Studies and Research, School of Computing, APIIT, Sri Lanka
[3]Professor, School of Computing, University of Colombo, Sri Lanka
E-mail: [1]ksamang@gmail.com, [2]kosala@apiit.lk, [3]ndk@ucsc.cmb.ac.lk

Abstract - Software development organizations battle in determining the functional test coverage prior to the delivery of software applications to the client and is purely depended on the tester's knowledge of the software. This paper presents a methodology to minimize tester knowledge dependency introducing partial automation of the test design process using ontology and semantic querying.

Keywords - Ontology, Semantic Querying, Functional Testing, Model Based Testing

I. INTRODUCTION

Software is an integral part of most businesses. For decades software engineering researchers and practitioners have attempted in reducing software development costs, delivering in shorter timeframes and improving software quality [7]. Snieska, Daunoriene & Zekeviciene [21] have studied on failure costs providing focus on hidden external failure costs expressed as decrease of part of market and loss of competitive advantage which are felt by the organisations for a long period.Failure costs are the most damaging costs of software; examples include support, maintenance, liability damages, and litigation expenses [21, 20]. Hence software vendors want to invest enough on appraisal costs as that would be the last resort before delivering the release to the client to minimise failure costs. Appraisal costs are costs of testing related to functionality of the application before the delivery of the software application to the client. When delivering the software application to the client, vital questions arise whether the testing is sufficient and sufficient test cases that client may use have been captured? The methodology suggested by this paper presents a solution for these constant nagging questions. Based on years of past experience of the firstauthor with many software applications testing, it is observed that the functional test coverage is decided by humans and it leaves room for different interpretations of the test coverage based on human understanding of the software application. This paper presents a methodology that will eliminate human interpretation of the functional test coverage.

II. ONTOLOGY BASEDMETHODOLOGY

The methodology is two folds; first an ontology-based requirements model is designed from base requirements whereas the second part is building a test cases extraction methodology, which aims at deriving test cases automatically from the ontology-based model. Operators would be used to build the logic for test case extraction methodology as in Bandara, Wang &Pahl[3] and Bandara[4]. A set of algorithms working along with a set of operators would be constructed to extract test cases from ontology specification.

The requirements will be captured as user stories as in present context most software development organisations use agile development methodology where user stories are captured in the product backlog[16, 9,1]. These user stories will be modelled into an ontology using Protégétool and further test cases will be extracted from this model. As a parallel process test cases also will be derived manually. Finally test cases extracted from the model would be compared against the test cases derived manually. The tool called Protégéwas selected to implement ontologies, because Protégé is an open-source tool to work with ontologies and it is the most widely used domain-independent, free, platform-independent tool to develop and manage terminologies, Ontologies, and knowledge bases in a broad range of application domains [12]. Musen[15] describes Protégé becoming the most widely used software for building and maintaining ontologies and its importance to Artificial Intelligence (AI) community. To validate the proposed methodology following criterion can be defined.

$$If\ TC_M = manually\ derived\ test\ cases,$$

$$TC_{TCEM} = test\ cases\ dervied\ through\ test\ case\ extraction\ methodology,$$

$$then,\quad \Delta_{TC} = TC_{TCEM} - TC_M$$

What needs to be proved is that $\Delta_{TC}>0$. There also needs to be a qualitative assessment of the test cases extracted through the model as to how these test cases can be easily understood and used by the testers.

III. THE EXPERIMENTATION AND RESULTS

The initial evaluation of the methodology was accomplished through a simple case study of arriving at a software application for a simple business scenario. This would provide the authors' the assurance of the direction of arriving at the solution is in fact reachable and realistic.

3.1. Overall Business Scenario

A new bank needs to have small application to find out population of a particular city to determine whether to establish an automatic teller machine (ATM) in that city. There will be a defined benchmark figure for a country and the decision to deploy ATMs is based on selection of all the cities that have the population equal or greater than the benchmark figure. They also wanted find out bordering countries and the capital city of that country. This business scenario implementation will be first tried in USA.

For the given business scenario to simplify and show the concept at a higher level instead of test cases, test scenarios will be identified. This is mere evaluation whether the solution is achievable. First using the manual method test scenarios will be identified. As per the first author's past experience, testers arrive at the test scenarios looking at the both requirements specification and developer solution. In this case overall business scenario replaces the requirements specification and the developer solution is shown below

3.2. Developer Solution

Developers decided to develop an application that will first populate the all the countries defaulting to USA as the country in a dropdown. Once the country is selected all the states of the country will be populated in a dropdown with a blank selection and user needs to select a state. When the state is selected all the cities of that state will be populated in a drop down with a blank selection and once the user selects the city, the population of that city and the bordering countries will be displayed. They also wanted to develop a report when a country is selected with the benchmark figure being entered in a textbox, a report button will generate a report listing all the cities that has a population equal or greater than the benchmark figure. At the bottom of the report the bordering countries and the capital city of that country will be printed.

3.3. Manual Test scenarios

By looking at the overall business scenario and developer solution, the following test scenarios can be captured manually.

1. Check whether all the countries appear in the country dropdown?
2. Check whether all the states belonging to the selected country appear in state dropdown?
3. Check whether all the cities belonging to the selected country and a state appear in the city dropdown?
4. Check when the country, state and city is selected, whether the population of that city and the bordering countries will be displayed?
5. Validate benchmark figure text box whether it can only accept numeric values
6. Check when the country is selected and the benchmark figure is entered whether all the cities that has a population equal or greater than the benchmark figure is displayed together with bordering countries and capital city of that country in the report to view the cities where ATMs can be installed?

As opposed manual methodology, the purpose of the evaluation stage of using ontologies is to identify test scenarios from the user stories by not having visibility to the developer solution. Therefore overall business scenario needs to be broken into simple user stories in building triples and data definitions in arriving at the ontology

3.4. User Stories

The following user stories can be arrived looking at the overall business scenario.

1. Country borders a country
2. State is a part of a country
3. City is located in a country
4. A city can be the capital of the country
5. Bank has announced benchmark figures
6. Benchmark figure is defined for a country
7. Atm City is a city that has equal or greater population than the benchmark figure

3.5. Data definitions

The following data definitions can be arrived looking at the overall business scenario.

1. population in an integer that is defined for country, state and city
2. benchmark Population is an integer that denotes the Benchmark Figure
3. name is a string that is defined for a country, state and city to capture the name of a country, state or city

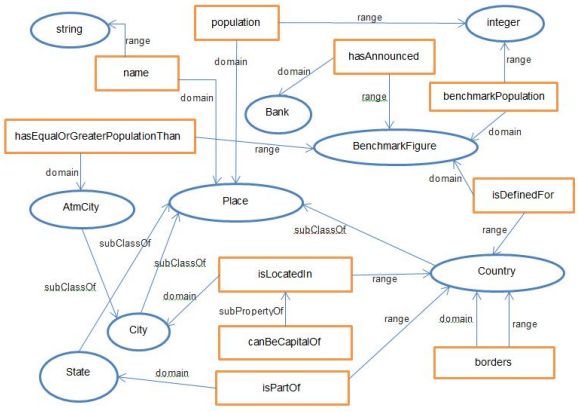


Fig.1. The solution ontology graph

3.6. Constructing the ontology

The fig. 1 represents solution ontology graph of the user stories. An overall class has been defined as Place with Country, State and City being the subclasses of the class Place. Using the Protégétool the ontology was constructed.

3.7. Extracting information from ontology

Using protégé tool the aforementioned ontology can be built and all the data can be further extracted using SPARQL queries. Arias et al. [2] mentions that, SPARQL is a declarative language, which can perform searches and data extraction for extracting information from RDF graphs where RDF is a simple declarative data model of triples to describe resources. Sintek& Decker [19] mentions triple as layered and modular rule language for semantic web and triple can be simply explained as a set of three entities that codifies a statement about semantic data in the form of subject–predicate–object expressions (e.g. "Bob is 35", or "Bob knows John"). Sections to follow will provide queries executed in Protégé and result sets obtained.

3.8. First query

First query was performed looking at triples that have subject, predicate and object relationships as follows

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?subject ?predicate ?object
WHERE { ?predicate rdfs:range ?object .
?predicate rdfs:domain ?subject }

The result test obtained from the above query is further divided in Table 1 and Table 2 separating object properties and the three data properties; population, benchmarkPopulation and name.

Subject	Predicate	Object
Bank	has Announced	BenchmarkFigure
AtmCity	hasEqualOrGreaterPopulatio nThan	BenchmarkFigure
Country	Borders	Country
City	IsLocatedIn	Country
BenchmarkFigure	isDefinedFor	Country
Subject	**Predicate**	**Object**
State	isPartOf	Country

Table 1: Result set of SPARQL query 1 separated with object properties

Subject	Predicate	Object
BenchmarkFigure	benchmarkPopulation	xsd:integer
Place	population	xsd:integer
Place	name	xsd:string

Table2: Result set of SPARQL query 1 separated with data properties

The result set of 1stquery returns almost all the user stories, and if closely observes it has not returned the 4thuser story (a city can be capital of country). When evaluating the data definitions, “Place” has replaced the group “country”, “state” and “city”.

3.9, Second query

Second query is performed to find out all sub classes

SELECT ?subject ?object WHERE { ?subject rdfs:subClassOf ?object}

Subject	Object
AtmCity	City
City	Place
State	Place
Country	Place

Table 3: Result set of SPARQL query 2 of sub classes

The idea of extracting of sub classes listed in the subject column of the above table is to find additional user stories and data definitions to be captured. Each sub class can make an additional user story by replacing the sub class (subject column) in place of the value given for the class in the object column. Now the result set given in Table 1and Table 2can be further expanded as follows where inserted rows are highlighted.

Subject	Predicate	Object
Bank	hasAnnounced	BenchmarkFigure
AtmCity	hasEqualOrGreaterPopulatio nThan	BenchmarkFigure
Country	borders	Country
City	IsLocatedIn	Country
BenchmarkFigure	isDefinedFor	Country
State	isPartOf	Country

Table 4: Updated result set of SPARQL query 1 including sub classes with object properties

Subject	Predicate	Object
BenchmarkFigure	benchmarkPopulation	xsd:integer
Place	population	xsd:integer
Place	name	xsd:string
Subject	**Predicate**	**Object**

Table 5: Updated result set of SPARQL query 1 including sub classes with data properties

Now it can be noted that an implicit user story “AtmCity is located in Country” has been captured and country, state and city related individual data definitions have also been captured together with the place data definition.

3.10. Third query

Third query is performed to find out all sub properties

SELECT ?subject ?object
WHERE { ?subject rdfs:subPropertyOf ?object}

Subject	Object
canBeCapitalOf	isLocatedIn

Table 6: Result set of SPARQL query 3 of sub properties

Similarly as stated for sub classes each sub property can identify additional user story by sub property replacing the property value. There will be no change to data definitions because sub property is related object properties and not data properties. The result set given in Table 4 can be further expanded as follows where inserted rows are highlighted

Subject	Predicate	Object
Bank	hasAnnounced	BenchmarkFigure
AtmCity	hasEqualOrGreaterPopulationThan	BenchmarkFigure
Country	borders	Country
City	IsLocatedIn	Country
City	capitalOf	Country
AtmCity	IsLocatedIn	Country
AtmCity	capitalOf	Country
BenchmarkFigure	isDefinedFor	Country
State	isPartOf	Country

Table7: Updated result set of SPARQL query 1 of object properties which includes sub classes and sub properties

It is interesting to note that earlier stated 4thuser story is now captured (a city can be capital of country) and one more implicit user story of AtmCity can be capital of country is captured. Therefore we can observe even at the requirements level, implicit requirements (directly not stated requirements) also can be captured when representing the user stories in ontology through protégé tool and further extracting information using SPARQL queries. Next step is translating these user stories and data definitions into following test scenarios sequentially looking at the Table7and Table 5.

3.11. Extracted Test scenarios from ontology

Following test scenarios can be captured sequentially looking at the Table8 and Table 6.

1. Check whether there exists a field to enter benchmark figure?
2. Check whether identified Atm Cities have equal or greater population than benchmark Population?
3. Check whether bordering countries are captured for a selected country?
4. Check whether all cities are captured within the selected country?
5. Check whether capital city of selected country is captured?
6. Check whether identified Atm cities are actually belonging to the selected country?
7. Check whether any identified Atm city can be the capital of the selected country?
8. Check whether a benchmark Figure is defined for each country?
9. Check whether all states are identified of a selected country?
10. Check whether benchmark Population is an integer?
11. Check whether Place, City, State and Country have corresponding field to capture the population and whether the corresponding field can only accept integers?
12. Check whether Place, City, State and Country have corresponding field to capture the name and whether the corresponding field is a string type?

IV. COMPARISON AND ANALYSIS OF TEST RESULTS

Table 9 shows the comparison of manual test scenarios against ontology bases test scenarios.

Table 9: Comparison of manual vs. ontology based test scenarios

Manual test scenario	Mapping of ontology based test scenario
1. Check whether all the countries appear in the country dropdown?	Check whether Place, City, State and Country have corresponding field to capture the name and whether the corresponding field is a string type? (**scenario 12**)
2. Check whether all the states belonging to the selected country appear in state dropdown?	Check whether all states are identified of a selected country??(**scenario 9**)
3. Check whether all the cities belonging to the selected country and a state appear in the city dropdown?	Check whether all cities are captured within the selected country (**scenario 4**)
4. Check when the country, state and city is selected, whether the population of that city and the bordering countries will be displayed?	Check whether Place, City, State and Country have corresponding field to capture the population… (**scenario 11**) and Check whether bordering countries are captured for a selected country? (**scenario 3**)
5. Validate benchmark figure text box whether it can only accept numeric values	Check whether benchmarkPopulation is an integer? (**scenario 10**)
Manual test scenario	**Mapping of ontology based test scenario**
6. Check when the country is selected and the benchmark figure is entered whether all the cities that has a population equal or greater than the benchmark figure is displayed together with bordering countries and capital city of that country in the report to view the cities where ATMs can be installed?.	Check whether identified AtmCities have equal or greater population than benchmarkPopulation? (**scenario 2**) and Check whether capital city of selected country is captured? (**scenario 5**)

Now it can be simply observed that the level of details of test scenarios identified though the ontology model is much higher than the approach taken manually. The following test scenarios of ontology approach were not specifically captured in the manual approach

Scenario 1 : Check whether there exists a field to enter benchmark figure

Scenario 6 : Check whether identified Atmcities are actually belonging to the selected country

Scenario 7 : Check whether any identified Atmcity can be the capital of the selected country

Scenario 8 : Check whether a benchmark Figure is defined for each country

The scenario 1 above can be considered as developer implementation decision that has to be checked at the developer level but scenario 8 should be checked at developer implementation level as well as functional test level making sure that benchmark figure related features work for more than one country. Scenario 6 and scenario 7 are purely functional level tests that have been identified primarily captured through

implicit relationships. The key advantage of the ontology based approach is the ability to capture implicit relationships that may be missed in the manual approach. The ontology based approach is more suitable for complex business scenarios. It is also interesting to note that the manual approach is taken by looking at the implementation, i.e. country, state and city fields are treated as drop downs instead of text boxes or any other user interface control, but the ontology approach has merely treated input as a variable that can belong to any user interface control. Therefore beyond doubt experimentation and results have proven that selected methodology is workable and achievable.

V. RELATED WORK

The sub sections discuss only the key areas and the key literature the authors were influenced to arrive at the proposed methodology.

5.1. Ontology based requirements engineering

Importance of requirement engineering has been already identified in several studies [10]. Requirements are the starting point of the software development life cycle and requirements specification is the primary document of communication between the developer and the customer. Therefore, directly linking and extracting test cases from a requirements model should save time and money and decrease the risk of overrun budget and project failure. Challenges in requirement engineering are because of inconsistencies and incompleteness of requirements documents and the root causes of these inconsistencies are involvement of multiple stakeholders and multiple sources [18,13]. To solve the inconsistency problem requirements are removed from the requirements document leading into incompleteness [18]. Zowghi & Gervasi[25] describes this vicious circle as an informal association between consistence, completeness and correctness. Ontology provides a formal representation of knowledge and association between concepts and they are well suited towards finding a solution for this problem [18]. Ontology fits well with the evolutionary approach to the specification of requirements and domain knowledge [11,24]. Ontologies further can support requirement management and traceability [14].

5.2. Model based testing (MBT)

The comparison of the effectiveness (issues found) and efficiency (effort spent) of the manual testing to MBT, Schulze et al. [17] observed that manual testing required less preparation time and had uneven test coverage whereas MBT required more preparation time, but more systematic, and detected more issues. Further Schulze et al. [17] concludes that the manual approach detected more inconsistencies between specified and actual text labels, MBT detected more functional issues. Model based testing is new and evolving methodology for generating the suite of test cases from requirements [8]. This is directly relevant to paper presented. There is a strong relationship between requirements and test cases that generates automatic tests and the test cases are generated from the data model i.e. looking at combinations of test data [8].

The taxonomy of model based testing approaches produced by Utting, Pretschner & Legeard [23], simply describes the process of model based testing. The model of a system under test (SUT) is built from informal requirements or existing specification. This model is called test model and the same can be used to design and develop the system but it is important to keep the test model separately from software development activities. This is to prevent errors from software development model propagating to the test model. Utting, Pretschner&Legeard[23] describes a process where test cases are derived through test specification on one branch and test cases are derived through the model on the other branch and then a comparison happens between the two branches. This process, which provides a concept and a frame to assess the MBT tools, arrived through the paper described by Utting, Pretschner & Legeard [23] strongly influenced the authors to seek a solution using the model based approach.

CONCLUSION AND FUTURE WORK

The proposed methodology would solve the gaps in test designing and tester's understanding, but is dependent on specialist skills on ontology. A senior test engineer can take up the challenge on acquiring the skills or specialised people can be trained on ontology. The success of the methodology can eliminate skilled testers, but will introduce a novel skill on working with ontologies to the industry. Most importantly the contribution through this methodology will target the overall quality and will reduce the failures of the software application in front of the end-user or customer. Also, the ontology based methodology will strongly support latest trends in software development life cycle such as changing requirements or scope and incremental software development. When changes happen only the ontology model needs to be changed and the rest will happen automatically.

The methodology proposed can be implemented as an add-on to the already existing Protégétool thus contributing to the open source AI community. This methodology can also produce an extendable solution that can integrate with already existing test management tools such as "TestLink". "TestLink" is a widely used open source web based test management tool that can easily import or export test cases [22]

REFERENCES

[1] Ambler, C., Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process, John Wiley & Sons, Inc., New York, 2002

[2] Arias, M., Fernández, J.D., Martínez-Prieto, M.A. & de la Fuente, P., "An empirical study of real-world SPARQL queries", arXiv preprint arXiv, Vol.1103 No. 5043, 2011

[3] Bandara, K.Y., Wang, M. &Pahl, C., "An extended ontology-based context model and manipulation calculus for dynamic Web service processes" In Service Oriented Computing and Applications, Springer, pp. 1-20, 2013

[4] Bandara, K.G.Y., "Ontology Based Contextualization and Context Constraints Management in Web Service Processes", Doctoral dissertation, Dublin City University, 2012

[5] Bandara, K.Y., Wang, MX &Pahl, C., "Dynamic integration of context model constraints in web service processes" In International Software Engineering Conference SE'2009, Innsbruck, Austria. IASTED, 2009a

[6] Bandara, K.Y., Wang, M &Pahl, C.,"Context modeling and constraints binding in web service business processes" In Proceedings of the first international workshop on Context-aware software technology and applications, ACM, August 2009, pp. 29-32, 2009b

[7] Basili, V.R., Heidrich, J, Lindvall, M, Münch, J, Regardie, M, Rombach, D &Trendowicz, A., "Linking software development and business strategy through measurement", arXiv preprintarXiv: Vol.1311, No. 6224, 2013

[8] Dalal, S.R., Jain, A., Karunanithi N., Leaton, J.M., Lott, M., Patton G.C.& Horowitz, B.M., "Model-Based Testing in Practice", ICSE '99 Proceedings of the 21st international conference on Software engineering, pp. 285-294, ACM New York, NY, USA, 1999

[9] Düchting, M., Zimmermann, D.&Nebe, K., "Incorporating User Centered Requirement Engineering into Agile Software Development", in Julie A. Jacko (ed), Human-Computer Interaction. Interaction Design and Usability, Lecture Notes in Computer Science Vol.4550, pp. 58-67, 2007

[10] Hall, T., Beecham, S.& Rainer, A., "Requirements problems in twelve software companies: an empirical analysis" in IEEE Proceedings-Software Vol. 149, No. 5, pp. 153-160, 2002

[11] Happel, H.J.&Seedorf, S., "Applications of ontologies in software engineering" in Proc. of Workshop on Sematic Web Enabled Software Engineering (SWESE), November 2006, the ISWC, pp. 5-9, 2006

[12] Jain V.& Singh M., "Ontology development and query retrieval using protégé tool", International Journal of Intelligent Systems and Applications (IJISA), Vol. 5 No. 9, pp. 67-75, 2013

[13] Lamsweerde, A., Darimont, R.&Letier, E., "Managing conflicts in goal-driven requirements engineering" in IEEE Transactions on Software Engineering, Vol. 24, No. 11, pp. 908-926, 1998

[14] Mayank, V., Kositsyna, N.& Austin, M., "Requirements Engineering and the Semantic Web, Part II - Representation, Management, and Validation of Requirements and System-Level Architectures", Technical Report: TR 2004-14, University of Maryland, 2004

[15] Musen, M.A., "The protégé project: a look back and a look forward", AI matters, Vol. 1, No4, pp.4-12, 2015

[16] O'Leary, P., McCaffery, F., Thiel, S.& Richardson, I., "An agile process model for product derivation in software product line engineering", Journal of Software: Evolution and Process, Vol. 24, No. 5, pp. 561-571, 2012

[17] Schulze, C., Ganesan, D., Lindvall, M., Cleaveland, R. & Goldman, D., "Assessing model-based testing: an empirical study conducted in industry" in Companion Proceedings of the 36th International Conference on Software Engineering (pp. 135-144), May 2014, ACM, 2014

[18] Siegemund, K., Thomas, E.J., Zhao, Y., Pan, J.&Assmann, U., "Towards ontology-driven requirements engineering" in Workshop Semantic Web Enabled Software Engineering at 10thInternational Semantic Web Conference (ISWC), October 2011, Bonn, 2011

[19] Sintek, M.& Decker, S., "TRIPLE—A query, inference, and transformation language for the semantic web", The Semantic Web—ISWC 2002, pp. 364-378, 2002

[20] Slaughter, S., Harter D.& Krishnan M., "Evaluating the Cost of Software Quality"Communications of the ACM, August 1998, Vol. 41, No. 8, pp. 67-73, 1998

[21] Snieska, V., Daunoriene, A.&Zekeviciene, A., "Hidden Costs in the Evaluation of Quality Failure Costs", Engineering Economics, Vol. 24 No. 3, pp.176-186, 2013

[22] "TestLink Tutorial: A Complete Guide", viewed 28 October 2017, https://www.guru99.com/testlink-tutorial-complete-guide.html, 2017

[23] Utting, M., Pretschner, A. &Legeard, B., "A taxonomy of model-based testing approaches", Software Testing, Verification and Reliability, John Wiley & Sons, Ltd, Vol. 22 No. 5, 2012

[24] Wouters, B., Deridder, D.& Van Paesschen, E., "The Use of Ontologies as a Backbone for Use Case Management" in European Conference on Object-Oriented Programming (ECOOP2000), Workshop : Objects and Classifications, a natural convergence, 2000

[25] Zowghi D.&Gervasi V., "The Three Cs of Requirements: Consistency, Completeness, and Correctness" in Proceedings of 8th International Workshop on Requirements Engineering: Foundation for Software Quality, (REFSQ'02), 2002

★★★