Manual Testing and Conceptual Data Modeling in Software Evaluation a Brief Summary

Dr Suneeta Sachin Rachanaikar, Sachin M Rachanaikar

***Abstract*— Software development initiates from requirements gathering and analysis then continues with design, coding and concludes with testing. With possible continuation there after by taking inputs from review which is supported by software testing. Testing is an important phase that too manual testing truly verifies software for expected outcome. Over the years researchers have contributed in manual testing field. In this article an attempt is made to reconnoiter efforts in the area of manual testing. Further study is made towards not only knowing so far efforts but possibility of enhancing manual testing. In doing so presence of conceptual data model (CDM) is taken into consideration to relate it with manual testing. Existing literature is the main source for possibility of making hypothesis of efficient manual testing by associating it with CDM.**

***Index Terms*— Conceptual Data Model. Manual Software Testing.**

I. INTRODUCTION

There is an abundance of testing approaches in the discipline of software engineering today. Over the last few decades many of these have come to be used and adopted by the industry as solutions to address the increasing demand for assuring software quality. Software testing is an important but expensive part of the software development process. Much research has been aimed at reducing cost and time taken to software testing. The two main reasons for testing the software are to make a judgment about quality or acceptability and to discover problems. Testing is concerned with errors, faults, failures, and incidents. A test is the act of exercising software with test cases. A test has two distinct goals to find failures and to demonstrate correct execution. Testing work can be divided into automated and manual testing. Automated software testing means the automation of software testing activities [1]. These activities include the development and execution of test scripts, the verification of testing requirements, and the use of automated test tools. Testing a software product forms a considerable expense but so do the costs caused by faults in the software product. Software Testing Techniques are scripted manual/automated, black box/user acceptance testing, exploratory, unit/Developer/White Box, regression, Ad Hoc, exploratory. Usability testing is another important verification type. Over the last few decades testing styles have come to be used and adopted by the industry as solutions to address the increasing demand for assuring software quality. Usability testing refers to the evaluation of information systems that involves participants who are representative of the target user population [2]. Software test as an important way to verify that software system quality level has received widespread attention of the society. Testing is a branch of Computer Science/Mathematics and is objective, rigorous and comprehensive. Testing techniques must have a logico-mathematical form in other words one right answer. Testing validates the product and measures development progress. Testing determines whether development processes are being followed. Testing is essential for the process improvement. Testing an application is done to detect differences between existing and the required condition and evaluate the features of the software application. Testing activities performed by the testers without the help of any software testing tools is called as manual testing. Manual testing is better suited for finding new and unexpected errors. The establishment of test automation is a risky investment project. Decision on what to automate and what to test manually should be defined early in the project to avoid problem [3]. Software development has phases in it among these phases testing is one important phase as it validates the developed software. Modeling is considered to be a way for the software development. Modeling affects all the phases of software development. Various modeling techniques exist CDM is part in the modeling activities and might influence manual testing. This article makes an attempt to relate manual software testing and CDM. Rest of the paper is organized as section II is about background justifying importance of manual testing and CDM, survey of major research contributions in the field of manual software testing, CDM and then existing proposal that relate manual testing and CDM is mentioned in section III, section IV makes statement of possible CDM and manual testing association from the inputs obtained from our study of background and existing works and the article concludes with review note summary. Initially we surveyed software testing which includes automated and manual testing both and CDM works so far. Then we focused on manual testing and CDM so among more than 200 articles on the subject initial filtering is done to reduce to 75 articles which focus only on manual technique and CDM. These 75 articles are further refined based on which relate to our study and are mentioned in the article.

Manuscript received Sep 24, 2016

Dr Suneeta Sachin Rachanaikar, Researcher residing in California, USA
Sachin M Rachanaikar, Business Analyst, Modeln Inc USA

II. BACKGROUND

Testing is probably the most important cost factor in software development. It is sometimes estimated to constitute 50–60% of the total effort spent to create a software system in typical cases. Model based testing is specifically tailored for small applications, embedded systems, user interfaces, and state-rich systems with reasonably complex data as discussed in the work of Ibrahim et al [4]. Alternating Variable Method approach to search-based test input generation is described in the work carried out by Kempka et al [7]. Two different techniques orthogonal arrays and the Allpairs algorithm are used to identify all the pairs for creating test cases. Broadly approaches like combinatorial based, input enhancement based, graph based, case study oriented testing, exploratory exist in the literature concentrating software testing in general. Testing techniques include random testing, functional testing, control flow testing, data flow testing, mutation testing, regression testing. Modeling makes software development efficient in that data modeling influence is even important. Efforts on CDM for software development are in discussion in the literature. CDM maps requirements and meets business expectations if modeled appropriately. In any software development user expectation is given utmost importance. While carrying out manual testing user approval is one of the key factor. CDM representation to elaborate business requirements might help in the review process of manual testing. Thus it is required to explore manual testing and CDM to formulate methodology to make manual testing efficient.

III. RESEARCH CONTRIBUTIONS

In order to analyze and understand manual testing our study began with software testing and constrained to manual testing which involves different steps in it. Based on manual testing automation is decided in many instances. Contributions in the field of manual testing and in conceptual data modeling are studied separately while studying other methods too. Existing efforts in both fields are giving hint that both can be related i.e. manual testing and CDM to improve performance of manual testing. Accordingly research articles on manual testing, CDM and then few works relating manual testing and CDM are focused. Next paragraphs elaborate contributions in these fields. Taipale et al follow process of building a theory from case study research and even emphasize data collection and data analysis in [3]. Common aspects are found among load testing, performance testing and stress testing [5]. Load testing definition is unified with the existing load testing interpretations as well as performance and stress testing interpretations, which are also about load testing. Creating an automated test is usually more time consuming than running it once manually. The cost differential varies depending on the product and the automation style [15]. Author describe two expert-based and one user based usability method, heuristic evaluation, the cognitive walkthrough, and the think aloud in [6] and even focus on testing health technologies.

Testing is the last chance during development to detect and correct possible software defects at a reasonable price [33].Correlation between the parameters which are more sensitive for software performance is given by authors in [8]. In addition ideal measurement values are given to the developer at coding phase with an appropriate confidence bound value for each parameter. Author has discussed most relevant challenges in software testing, testing process, test criteria and testing types like component based testing, object oriented testing [9]. Jiang and Hassan described verifying the test data against fixed threshold values, searching through the test data for known problem patterns and automated detection of anomalous behaviors and mentioned open problems [10]. Further authors have surveyed techniques that are used in the three phases of load testing the load design phase, the load execution phase, and the load test analysis phase. Authors focus on recommendation system principle, software test case design and its property and collaborative filtering algorithm [11]. On demand services have become available to perform manual testing as needed without introducing delay and are often many times faster than equivalent automated tests [13]. Test case prioritization, input to it and the objectives of prioritization in addition rapid releases are emphasized by Hemmati et al [19]. Systematic review of software test automation benefits and limitations in academic literature and survey of the practitioners' view of software test automation benefits and limitations has been carried out by Rafi et al [20]. Symbolic execution provides a way to automatically generate inputs that trigger software errors ranging from low level program crashes to higher level semantic properties, generate test suite that achieve high program coverage and provide per path correctness guarantees [21]. Authors have worked on most frequent error types when manually deriving test cases from an activity diagram or state machine [22]. Authors have shown that optimizing whole test suites towards a coverage criterion is superior to the traditional approach of targeting one coverage goal at a time [23]. Manual testing practices, classification and session strategies are discussed by Itkonen et al [29]. Designing the test cases beforehand and writing them down in a test case specification document is only one way of applying defect detection strategies [30]. Role of experience is considered for effective testing in order to develop successful testing strategies and tool support [31]. The organization model used by a company has remarkable effects on the strong and weak points of its testing process [32].

Conceptual modeling continues to evolve as researchers and practitioners reflect on the challenges of modeling and implementing data intensive problems that appear in business and in science [12]. Conceptual modeling can support cognitive argumentation humans create the semantics for concepts in order to obtain a more realistic and formal view of reality. In business architecture the use of conceptual modeling techniques aims to structure the approach to modeling and understanding holistic business knowledge and give business managers the insight they need to make better decisions [14]. Authors propose a Conceptual Modeling Quality Framework in [16]. This framework is useful for evaluating not only the end result of the conceptual modeling process, the conceptual representation, but also the quality of the modeling process itself. Author has argued that component specifications need conceptual data models both on component and interface level and that those models should be closely related [17]. An enterprise model comprises conceptual models of software systems, object or component models that are integrated with conceptual models of the surrounding action systems, business process models or strategy models [18]. Validation metrics are established during the requirements phase of the conceptual model development and incorporate numerical and experimental uncertainty. The conceptual model is based on CRUD (create, read, update, and delete) operations. The template based on the ABC model (application = business logic + CRUD) was useful for requirement specification of business logic as mentioned in the work by authors [24]. Author focus on how to effectively translate a requirement specification into a conceptual model [25]. Software life cycle models provide a conceptual scheme for rationally managing the development of software systems. Such a scheme could therefore serve as a basis for planning, organizing, staffing, coordinating, budgeting, and directing software development activities including testing [26]. Approach proposed by Blanco and Tuya uses a conceptual data model as the basis for designing the test model according to the system specification [27]. When a means of testing is provided for all constructs of the conceptual model the testing is said to be exhaustive. A matrix form can be used to demonstrate the completeness of the testing [28].

IV. CONCEPTUAL DATA MODEL IN MANUAL TESTING

Data modeling a way to structure and organize data is widely applied in different industries because it can be used easily by databases. Early modeling and analysis help companies to understand their needs and problems with potential solutions. In real business world since the goal of modeling always changes the data modeling turns out to be very important especially in the early designing phase. Also during the process communication and precision are two key benefits that make a data model crucial. A conceptual data model identifies the highest level relationships between the different entities. It includes the important entities and the relationships among them. CDM is the first step in constructing a data model in top-down approach and is a clear and accurate visual representation of the *business of an organization*. CDM visualizes the overall structure of the database and provides high-level information about the subject areas or data structures of an organization. CDM discussion starts with main subject area of an organization and then all the major entities of each subject area are discussed in detail. Major contributions in software testing field justify efficient software testing process is crucial. In this direction CDM based approach to software testing can enhance software testing. Review process makes software testing more efficient. Examining any project related work is called review. Types of review include management reviews, technical reviews, code reviews, test case reviews (formal, informal). Conceptual data model could be used to perform review which improve software testing. This possible relationship of CDM approach of software testing as contained with respect to software testing is shown in figure 1.

Conceptual data model is created by gathering *business requirements* from various sources like *business documents*, discussion with functional teams, *business analysts*, smart management experts and end users who do the reporting on the database. Data modelers create conceptual data model and forward that model to functional team for their review.

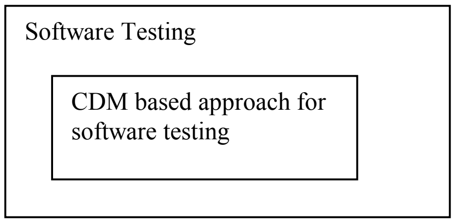

Figure 1 Block diagram relating CDM and software testing

Conceptual data model based approach for manual test case plan, test case analysis, test case design and test case generation is planned to be proposed in our next work as shown in figure 2. Need for CDM based approach is it is critical to find bugs before production and exhaustive testing is tough.

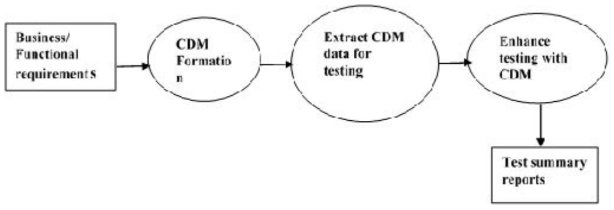
Business/ Functional requirements
CDM Formation
Extract CDM data for testing
Enhance testing with CDM
Test summary reports

Figure 2. CDM based manual testing

CONCLUSION

Conventional functional testing is start point from where quality of software is evaluated and to ensure this manual testing is the only option based on which further automation testing is developed. In this direction existing literature is the main input to justify the importance of manual testing. Betterment of manual testing is an ongoing area of research among experts. Performance analysis, conceptual data model and manual testing terminologies combined might impact the process of verification and validation of the software product. CDM analysis might help in reviewing manual testing activities like test case plan, design, and execution which might involve interaction with business users.

REFERENCES

[1] Dustin E, Rashka J, Paul J, Automated software testing: introduction management, and performance. Addison-Wesley, Boston, 1999.
[2] Andre W. Kushniruk, Vimla L. Patel ,James J. Cimino, Usability Testing in Medical Informatics: Cognitive Approaches, 1091-8280/97/$5.00 0 1997 AMIA, Inc, 1997 to Evaluation of Information Systems and User Interfaces, 1091-8280/97/$5.00 0 1997 AMIA, Inc, 1997.
[3] Ossi Taipale, Jussi Kasurinen, Katja Karhu, Kari Smol ander, Trade-off between automated and manual software testing, Int J Syst Assur Eng Manag (Apr-June 2011) 2(2):114–125 DOI 10.1007/s13198-011-0065-6, 2011.
[4] Ibrahim K. El-Far and James A. Whittaker, Model-based Software Testing, Encyclopedia on Software Engineering, Wiley, 2001.
[5] Zhen Ming Jiang, Ahmed E. Hassan, A Survey on Load Testing of Large-Scale Software Systems, 0098-5589 (c) 2015 IEEE., 2015.
[6] Monique W.M. Jaspers, A comparison of usability methods for testing interactive health technologies: Methodological aspects and empirical evidence, International journal of medical informatics, 2009.
[7] JosephKempka, PhilMcMinn, DirkSudholt, Design and analysis of different alternating variable searches for search-based software testing, Theoretical Computer Science, 2015.
[8] Charmy Patel, Ravi Gulati, Identifying Ideal Values of Parameters for Software Performance Testing, 978-1-4673-9354-6/15/$31.00 ©2015 IEEE, 2015.
[9] Antonia Bertolino, Software Testing Research: Achievements, Challenges, Dreams, Future of Software Engineering(FOSE'07), IEEE, 2007.
[10] Zhen Ming Jiang, Ahmed E. Hassan, A Survey on Load Testing of Large-Scale Software Systems, DOI

10.1109/TSE.2015.2445340, IEEE Transactions on Software Engineering, 2015.
[11] Jianxin Ge, Jiaomin Liu, Software test cases recommendation system research based on collaborative filtering, 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing , 2016.
[12] Veda C. Storey, Juan C. Trujillo, Stephen W. Liddle, Research on conceptual modeling: Themes, topics, and introduction to the special issue, Data & Knowledge Engineering, Volume 98, 2015.
[13] G. Philip Rogers , Paul Miles , Manual testing's newfound place in the automated testing world, IEEE AUTOTESTCON, 2015
[14] Amjad Fayoumi, Pericles Loucopoulos, Conceptual modeling for the design of intelligent and emergent information systems, Expert Systems With Applications, 2016.
[15] Vivek kumar, Comparison of manual and automation testing, International Journal of Research in Science And Technology, 2012
[16] H. James Nelson , Geert Poels, Marcela Genero, Mario Piattini, A conceptual modeling quality framework, Software Qual J, 20:201–228, 2012.
[17] Jörg Ackermann, Using a Specification Data Model for Specification of Black-Box Software Components, Enterprise Modelling and Information Systems Architectures, Vol. 2, No. 1, May 2007.
[18] Ulrich Frank, Multi-Perspective Enterprise Modeling: Foundational Concepts, Prospects and Future Research Challenges, Springer, Software & Systems Modelling, Volume 13, Issue 3, pp 941–962,2014.
[19] Hadi Hemmati, Zhihan Fang, Mika V. Mäntylä, Bram Adams, Prioritizing manual test cases in rapid release environments, Software testing, verification and reliability, DOI: 10.1002/stvr.1609, 2016.
[20] Dudekula Mohammad Rafi, Katam Reddy Kiran Moses, Kai Petersen, Mika V. Mantyla, Benefits and Limitations of Automated Software Testing: Systematic Literature Review and Practitioner Survey, 7th International Workshop on Automation of Software Test, DOI: 10.1109/IWAST.2012.6228988, 2012.
[21] Cristian Cadar, Koushik Sen, Symbolic Execution for Software Testing: Three Decades Later, Communications of the ACM, Volume 56 Issue 2, Pages 82-90,February 2013, 0.1145/2408776.2408795.
[22] Michael Felderer, Andrea Herrmann, Manual Test Case Derivation from UML Activity Diagrams and State Machines: a Controlled Experiment, Information and Software Technology, Volume 61 Issue C, Pages 1-15, May 2015, 10.1016/j.infsof.2014.12.005.
[23] Gordon Fraser, Andrea Arcuri, Whole Test Suite Generation IEEE Transactions on Software Engineering,Volume 39, Issue 8, Aug. 2013, 10.1109/TSE.2013.6
[24] Takeshi Chusho, Jing Li, Conceptual modeling for web applications and definitions of business logic for end-user initiative development, Proc. The IADIS International Conference on Information Systems 2014 (IS2014), 2014, pp.184-192.
[25] Yeol Song, Yongjun Zhu, Hyithaek Ceong, Ornsiri Thonggoom, Methodologies for Semi-automated Conceptual Data Modeling from Requirements, LNCS 9381, pp. 18–31, 2015, DOI: 10.1007/978-3-319-25264-3_2.
[26] Walt Scacchi, Process Models in Software Engineering, Encyclopedia of Software Engineering, 2nd Edition, John Wiley and Sons, Inc, New York, December 2001.
[27] Raquel Blanco, Javier Tuya, A Test Model for Graph Database Applications: An MDABased Approach, A-TEST'15, August 30-31, 2015, http://dx.doi.org/10.1145/2804322.2804324
[28] Man-Yee Chan and Shing-Chi Cheung. Applying white box testing to database applications. Technical Report HKUST-CS9901, Hong Kong University of Science and Technology, Department of Computer Science, February 1999.
[29] Juha Itkonen, Mika V. Mäntylä, Casper Lassenius, How Do Testers Do It? An Exploratory Study on Manual Testing Practices, Third International Symposium on Empirical Software Engineering and Measurement, 2009.
[30] Juha Itkonen, Mika V. Mäntylä,Casper Lassenius, Defect Detection Efficiency: Test Case Based vs. Exploratory Testing, Proceedings of International Symposium on Empirical Software Engineering and Measurement, pp. 61–70, 2007.
[31] Armin Beer, Rudolf Ramler, The Role of Experience in Software Testing Practice, Software Engineering and Advanced Applications, SEAA '08. 34th Euromicro Conference, 2008.
[32] Jarmo J. Ahonen, Tuukka Junttila, Markku Sakkinen, Impacts of the Organizational Model on Testing: Three Industrial Cases, Empirical Software Engineering, 9, 275–296, 2004.
[33] NATALIA JURISTO, ANA M. MORENO, SIRA VEGAS, Reviewing 25 Years of Testing Technique Experiments, Empirical Software Engineering, 9, 7–44, 2004.

Dr Suneeta Sachin Rachanaikar
Researcher residing in California, USA, PhD holder in Information and Communication Engineering from Anna University of Technology Chennai. Presented and published 8 research articles in International journals and International conferences. Having a decade experience in teaching with inclination on research. Reviewer for International conferences and International Journals. Defined process for software performance analysis as part of research work and relating software evaluation with modeling further.

Sachin M Rachanaikar
Business Analyst, Modeln Inc USA. Associated with analysis and manual testing as part of the work for more than five years with strong domain knowledge of more than a decade. Involved in interaction with business users while evaluating pharma product. Experienced working with SAP SD in addition.