An Efficient Objective Quality Model for Agile Application Development

M.Usman Malik
UET Taxila
Rawalpindi, Pakistan

M. Haseeb Nasir
UET Taxila
Rawalpindi, Pakistan

Ali Javed
UET Taxila
Rawalpindi, Pakistan

ABSTRACT

In the recent past, software industry has seen a paradigm shift in the software development methodologies. Several software development models have been proposed from time to time keeping in view the changing patterns and requirements of customers. Advancements in technology and change in standard operating procedures of corporate sector organizations has led to the development and evolution of several software development models. Currently, agile application development methodology is the most widely use software development model owing to its flexibility. Quality models have also evolved with corresponding to the software development models. Previously several quality metrics models have been developed for waterfall, spiral and incremental development. Work is also being done to propose a quality model which could ensure the quality of agile product being developed. However a robust and standard quality metrics model is lacking for the agile methodologies which can ensure that agile product being developed will fulfill implicit as well as explicit quality characteristics as agreed by the stakeholders. In this paper a quality metrics model for agile application development methodologies has been proposed which ascertains that the quality of the product being developed in agile requirement meet the standard and desired criteria of quality.

General Terms

Agile Application Development, Scrum, Quality Models, Quality Attributes, Quality Metrics

Keywords

Software Engineering, Software Quality Assurance, Quality Metrics Model, Agile Engineering

1. INTRODUCTION

Agile application development methods are flexible approaches towards software development based on the principle of evolutionary and iterative development [1, 2]. Agile methodology is geared towards iterative [3] and opportunistic development approach [4]. In agile method, product is developed in small cycles with each cycle ending with the addition of a Feature highest in the priority list of the client or the product owner. In agile application Development, every cycle or iteration itself can be considered as a mini project or small subset of the project [1] At the end of each iteration, a shippable product is obtained which a fully functional product is containing all the functionalities which were decided in the iteration plan meeting. Agile methodology lays stress on short iterations due to the fact that at the end of iteration N, feedback from the Nth and the previous iterations can be used to improve the N+1 and the next releases which is not the case with waterfall or any other model. Quality metrics model is a set of characteristics which is used to evaluate the quality of the product being developed. Different types of quality metrics models have been developed which verify quality of product by evaluating a metrics against the criteria set to evaluate the product. Numerous quality models have been proposed but none of them completely evaluate an agile product keeping in with the perspective of all the stake holders involved in agile application cycle. In this paper such a quality metrics model has been proposed which evaluates an agile application in terms of quality criteria.

2. LITERATURE REVIEW

Quality assurance has always been a challenge in agile application development owing to the volatile and flexible nature of agile methodologies. Following are some of the famous agile methodologies that are widely used for agile application development.

2.1 Agile Methodologies

Agile methodologies are set of requirement gathering and product development strategies where a product is developed and evolved through interaction between self-evaluating cross-functional teams. In 2001, a brainstorming session was held in which 17 prominent software methodology researchers participated. At the end of the session they unanimously signed a document and named that “Agile Manifest” [5]. It has following major points.

- Individuals and interactions over process and tools.
- Working software over detailed documentation.
- Customer collaboration over contract negotiation.
- Respond to change over following a plan.

Following are some of the famous agile methodologies.

2.1.1 Feature Driven Development

In feature driven development or commonly known as FDD, focus is to develop individual features and then these features are iteratively integrated into the product to achieve overall functionality [6]. FDD is a phased activity. FDD starts from developing an overall model, followed by developing a list of features, then plan by feature, design by feature and ending up at development by feature. In FDD all the stakeholders review and agree upon a feature list [7] and a weekly 30 minute meeting is conducted to review the status of feature and product being developed.

2.1.2 Extreme Programming

Extreme Programming or often called XP is an agile methodology that relies on fast communication, process simplicity and quick feedback, in order to develop a software product [8].

In XP, major focus is on development. Quick development is done and not much time is spent on preparing detailed requirement documents. Rather, requirements and changes are embraced during the development cycle. Dr. Barry Boehm, a prominent researcher and founder of Boehm's quality model said that with the progress of a project, the cost of change in requirements increases exponentially. In order to counter these requirement change issues, extreme programming do not spend large cost on preparing requirement document and strategies, rather concentration is on development and integrating run time requirement changes.

Development in XP is done in very small increments. A customer representative is always present onsite and recommends what is to be developed in the next iteration. After each iteration, customer is asked to review the product, if customer recommends any change, it is implemented immediately and increment is released and integrated into the final product.

2.1.3 Dynamic System Development Method

DSDM is another effective agile application development methodology that focuses on rapid application development rather than adopting any formal requirement engineering technique and spending time and money on requirements gathering.

2.1.4 Scrum

Scrum is considered the best agile methodology for agile application development. In scrum, key focus is to continuously adopt to any requirement changes at run time. It is an extremely flexible development approach and follows the principle of iterative and incremental development. Scrum promotes an environment of collaboration where several cross functional teams work at one place to achieve product targets. [6,9]. Scrum involves several artifacts. An artifact that contains list of all the product features to be developed is called product backlog. Spring backlog is another important artifact that contains features that are going to be developed in the next sprint. Sprint is basically a two to four weeks development cycle in which features agreed in sprint backlog are implemented, tested and released.

2.2 Software Quality Models

A software quality model is a framework of characteristics that is used to ensure that software meets highest standards of quality. In this paper a software quality model has been proposed that can be used as reference in agile application development in order to develop a quality product. But before that, some general purpose quality models widely used in software industry will be discussed.

2.2.1 McCall Quality Model

McCall J. A, was the first to propose a quality model [10]. In his quality model, he has presented several characteristics of software that define quality of a software product. McCall has classified quality characteristics in three major categories.

First is product Revision, which further depends upon maintainability, flexibility and testability of software application. Second category is Product Operation which in turn depends upon integrity, reliability, correctness, efficiency and usability of the product. And the last category is Product Transition that depends upon attributes like reusability, portability and interoperability.

2.2.2 FURPS Quality Model

FURPS quality model [11] is basically composed of functional and non-functional characteristics of software which defines the quality of the software. FURPS was proposed by Grady B.R and Hewlett Packard Co and is often called HP FURPS. In FURPS, 'F' stands for functional requirements and remaining URPS defines non-functional requirements where 'U' represents usability (How user friendly and aesthetically attractive software is), 'R' represents reliability (Ability to recover from a fault), 'P' represents performance and 'S' represents supportability.

2.2.3 Boehm Quality Model

Proposed by Barry Boehm [12], this quality model is somewhat similar to McCall quality model but expands it by adding few additional quality characteristics. This model considers software as a utility and seeks to explain that how an efficient and maintainable software that is more usable is high quality than less usable and maintainable product.

2.2.4 IEEE Quality Model

IEEE quality model [13] contains quality characteristics that are necessary for a maintainable software product. The model basically explains that how an iterative process can be employed to execute and manage software maintenance activities. It also contains certain quality attributes that software must possess to be considered as a quality product. Other qualitative factors include portability, reliability, usability maintainability and several others.

2.2.5 ISO 9126-1 Quality Model

Like FURPS quality model, ISO 9126-1 quality model is also classified into two categories [14]. First category is Quality in use attributes and the second category is Internal and External Quality attributes. Quality in use attributes are those attributes that can be evaluated only when software is released and delivered to the end user. On the other hand internal quality attributes are those that can be measured even without executing the product, whereas external quality attributes are those that cannot be measured without executing the product. ISO 9126-1 quality model contains all of these attributes so that the product can meet internal, external and user oriented standards of quality.

3. QUALITY MODEL FOR AGILE APPLICATION DEVELOPMENT

In section 2, several prominent and widely used quality models have been explained. All of these quality models evaluate quality of software applications that have been developed using traditional software development models. Agile software development is different in essence from traditional software development and all the quality attributes defined in previous quality models are not necessary for agile application development. Quality attributes impact agile application development twice. Firstly, as aforementioned agile application development is an iterative process therefore in order to seamlessly develop future iterations; product should possess certain quality attributes. Secondly, quality of the product is measured when all the increments have been integrated and final product is developed. In this research, a quality model has been proposed that can be used to evaluate quality of a product being developed using agile methodologies. Following are the quality attributes that are critical for an agile application.

3.1 Availability

Availability refers to operative ability of the application. It states that application should always be available for use by end user, developers and testers [15]. Agile application is an iterative process where a customer or representative of end user is always present on site and is consistently evaluating the product. Therefore, in order to gain customer feedback at any point of time during the increment, system should be available and operative all the time.

3.2 Flexibility

Flexibility lies in the heart of Agile Application development. Agile application development is different from traditional approaches in the essence that it embraces change at any point of time. After each increment, product is evaluated and tested by a customer's representative, on site. Customer recommends changes that are adopted immediately. Flexibility refers to the ability of application to undergo changes when required without affecting the overall application [16]. Only a flexible application can adopt change in requirements and functionality; hence, flexibility is a critical quality attribute for agile application development.

3.3 Scalability

In agile application development, core functionalities with high priority are developed in earlier iterations and with the passage of time product evolve and remaining functionalities are added. Keeping in view the above scenario it can be inferred that product needs to scale to the added functionalities in that are added in each iteration. Scalability is the quality attribute that refers to the ability of the application to grow and expand seamlessly [15]. Scalability is a crucial quality attribute for agile applications, since agile applications are based on the principle of evolutionary development.

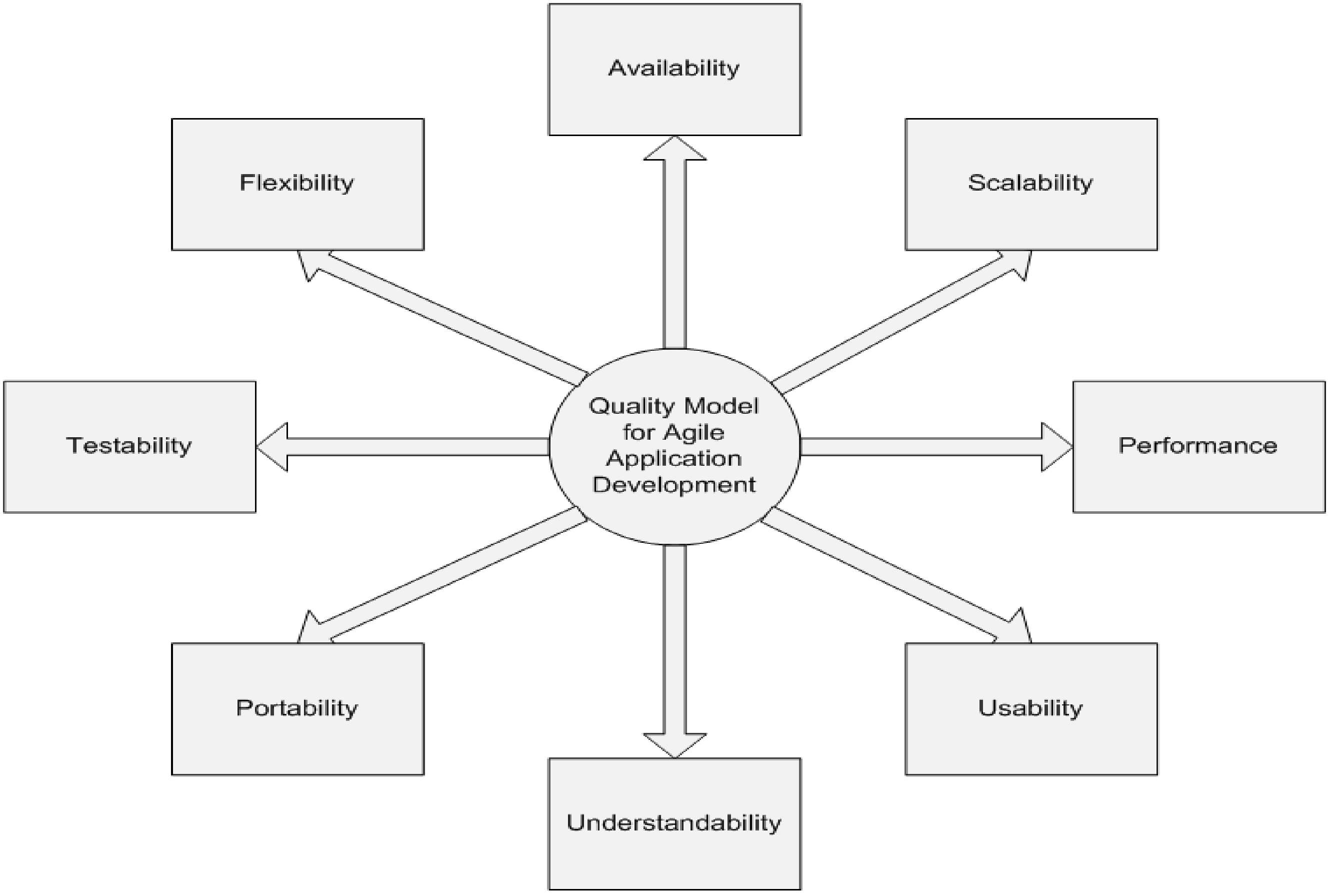

Fig 1: Quality Model for Agile Application Development

3.5 Performance

Performance is one such quality attribute that is present in almost all the famous quality models. Performance is also extremely critical in agile application development because of its rapid application development and testing characteristics. As aforementioned, product is being developed in fast-paced, small iteration where performance of the product is crucial in seamless and flawless execution of the program. Performance refers to processing capabilities of application with respect to time constraints [15] and it is one of the key quality attributes in agile application development.

3.6 Portability

Nothing is constant in agile application development. Requirement changes, hardware changes and platform changes occur often. In such volatile scenario, agile product being developed should be portable enough to be able to run on versatile platforms. Portability is that quality attribute of a system that enables application to operate on multiple hardware or software platforms [17]. Portability is another extremely critical attribute in agile applications and has huge impact on the quality of the product.

3.7 Usability

Usability is another critical quality attributes in agile application development. As aforementioned, a customer is consistently involved in requirement evolution during agile application development; he consistently evaluates system and provides his feedback. For end user to perform better evaluation, application under analysis must be usable [17]. Usability refers to the aesthetics and degree of ease of use of the application by the customer. Usability is extremely important quality attribute with reference to agile application development.

3.8 Understandability

During agile application development, several cross functional teams sit together and work at one place. In order to ensure seamless communication and interaction between all the teams, product developed should be understandable. Understandability refers to the ease with which a customer, tester and developers can explain and understand the usage of the product [17]. It signifies that how easy it will be for a user to perform a specific task and how easily can a tester understand and perform testing of the application which is a continuous process in agile applications. In fig 1, the quality model that has been proposed at the end of the result has been given which contains all the eight quality attributes that play a key role in ensuring overall quality of applications developed using agile methodologies.

4. IMPACT OF QUALITY ATTRIBUTES ON AGILE SDLC

In section 3, a quality model has been proposed with eight quality attributes that are most critical in agile application development. In this section an evaluation of the impact of these quality attributes have been presented. An evaluation matrix has been drawn that shows how helpful each quality attribute is during different individual agile software development life cycle phases. In the end an evaluation graph has been presented that shows the overall impact of each quality attribute on SDLC. By impact of quality attribute on a particular SDLC phase, it is meant that how helpful a particular quality attribute is or how positively, does an attribute impacts particular SDLC phase. For example a flexible application can have very high impact on requirement specification phase because it can accommodate requirement changes in the following iterations of agile process. However, testability does not have much impact on requirement phase. Testability impacts unit and integration testing phase. In order to quantitatively represent the impact, a scale of 1 to 5 points has been chosen where 1 means no impact and 5 means very high impact. Details of evaluation scale are given in Table 1.

Table 1: Evaluation Scale

Point	1	2	3	4	5
Value	No Impact	Little Impact	Average Impact	High Impact	Very High Impact

4.1 Quality Attributes Evaluation Matrix

An evaluation matix has been drawn in Table 2. First column contains quality attributes of quality models for agile applications whereas first row contains phases of SDLC. Intersection of rows and column contain impact of that particular attribute on SDLC phase. Last column of the matrix contains the overall impact of a particular quality attribute on SDLC which is sum of the individual impacts on different SDLC Phases.

Table 2: Quality Attributes' Impact on SDLC Phases

	Requirement Specification	**Design**	**Implementation**	**Unit Testing**	**Integration Testing**	**Release**	**Overall Impact**
Availability	**3**	**2**	**2**	**4**	**5**	**5**	**21**
Flexibility	**4**	**4**	**5**	**4**	**4**	**4**	**25**
Testability	**2**	**2**	**3**	**5**	**5**	**2**	**19**
Scalability	**4**	**4**	**4**	**3**	**3**	**5**	**23**
Performance	**2**	**2**	**2**	**4**	**4**	**5**	**19**
Portability	**2**	**3**	**4**	**5**	**5**	**5**	**24**
Understandability	**4**	**4**	**4**	**4**	**4**	**4**	**24**
Usability	**2**	**2**	**3**	**5**	**5**	**5**	**22**

From the evaluation matrix, it is evident that flexibility, with overall impact of 25, is the quality attribute that has highest impact on the quality of agile application followed by portability and understandability with overall impact of 24.

Evaluation graph of quality attributes has been drawn in Fig 4.1which shows the overall impact of each quality attribute on SDLC. Quality attributes have been mentioned on x-axis while corresponding impact has been mentioned on y-axis.

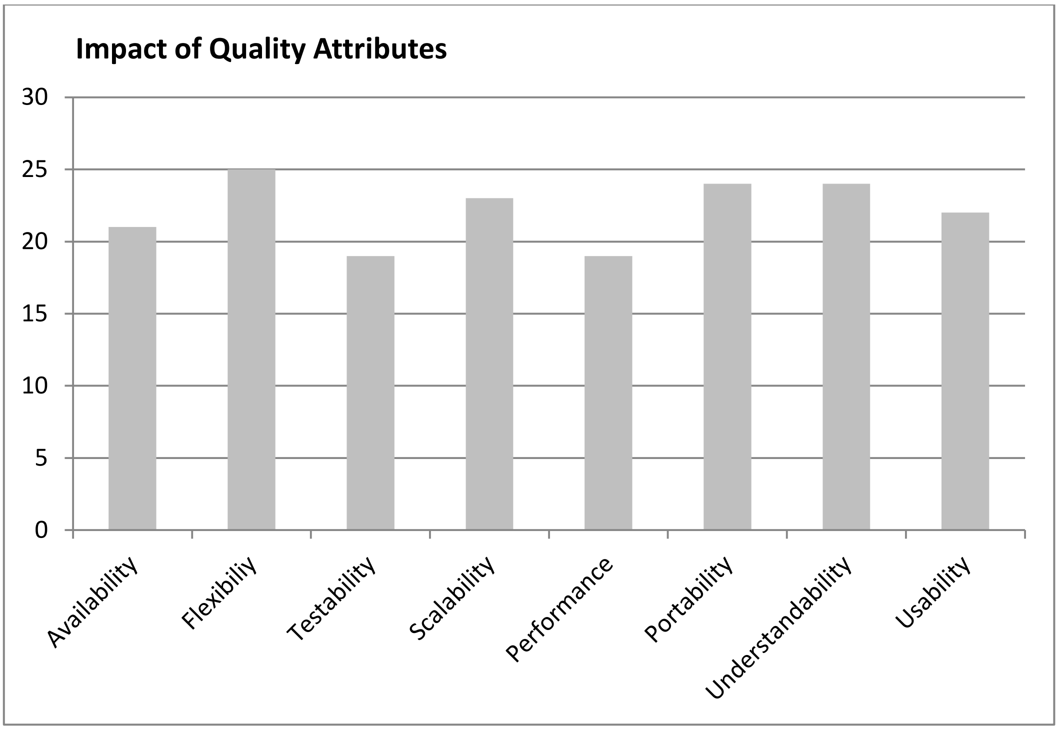

Fig 2: Overall Impact of Quality Attributes on Agile SDLC

5. CONCLUSION

The paper presents a novel quality model for agile application development. Eight quality attributes along with their importance in agile application development, have been mentioned. Also, role of each quality attribute in individual phases of SDLC have been evaluated and presented in tabular form with an overall impact presented in the form of a graph. The research shows that flexibility is the most important quality attribute in agile application development followed by portability and understandability.

6. FUTURE WORK

Further quality attributes can be added in the quality model like maintainability and modifiability and their impact on SDLC. Similarly, criteria for defining impact on SDLC needs improvement and can be fine-grained in the future work.

7. REFERENCES

[1] C. Larman, *Agile and Iterative Development: A Manager's Guide*. Boston: Addison Wesley, 2004.

[2] C. Larman and V. Basili, "A History of Iterative and Incremental Development," *IEEE Computer*, vol. 36, no. 6, pp. 47-56, June 2003.

[3] V. R. Basili and A. J. Turner, "Iterative Enhancement: A Practical Technique for Software Development," *IEEE Transactions on Software Engineering*, vol. 1, no. 4, pp. 266 - 270,1975.

[4] B.Curtis, "Three Problems Overcome with BehavioralModels of the Software Development Process (Panel)," International Conference on Software Engineering, Pittsburgh, PA, 1989, pp. 398-399.

[5] Murauskaite A., Adomauskas V., "Bottlenecks in Agile Software Development using Theory of Constraints(TOC) Principles", Master's Thesis, Gothenburg, Sweden 2008.

[6] Pekka Abrahamsson, Outi Salo, Jussi Rankainen & Juhani Warsta : *Agile software development methods*

[7] Peter Coad, Eric Lefebvre, Jeff De Luca : Java Modeling in Color with UML, Prentice Hall PTR, 1999, Chapter 6.

[8] Kent BeckExtreme *Programming explained*, Addison-Wesley, 1999. *Review and analysis*, VTT Electronics, 2002.

[9] Ken Schwaber, Mike Beedle: *Agile Software Development with Scrum*, Prentice Hall, 2001.

[10] Kumar, A., Kumar, R. and Grover, P. S. 2006. A changeImpact Assessment in Aspect-Oriented Software Systems, In the proceedings of International Software Engineering Conference Russia, (SECR-2006), Dec, pp. 83-87.

[11] Ghezzi, Jazaveri, C. M and Mandrioli. D. 1991. Fundamentals of Software Engineering. Prentice-Hall, NJ, USA.

[12] Boehm, B.W. et al. 1978. Characteristics of software Quality, TRW Series of software Technology, Amsterdam, North Holland.

[13] Hyatt, L. E. and Rosenberg, L. H.1996. Product Assurance Symposium and Software Product Assurance Workshop, Proceedings of meetings, European Space Agency, pp. 209

[14] IEEE 1993. Standard for Software Maintenance, Software Engineering Standards Subcommittee of the IEEE Computer Society.

[15] [15] Dromey, G. R. 1995. A model for software product quality, IEEE Trans. on software Eng., vol.21, no. 2,

pp.146-162

[16] Fusani, M.1995. Quality Models for Software Evolution Instruments International Seminar on Software Measuring & Testing, IEI -CNR /Qualital /SSSUP S.ANNA Pisa, Italia.

[17] IEEE 1993. Standard for Software Maintenance, Software Engineering Standards Subcommittee of the IEEE Computer Society.

9. AUTHOR'S PROFILE

M. Usman Malik has completed his Bachelors in Software Engineering from University of Engineering & Technology in July, 2012. Currently he is enrolled as part time MS Scholar of Software Engineering at the same institute. He provides consulting services, business planning solutions to various IT firms. He also provides research services to IT Industry. His areas of interest are Operating Systems, Mobile Application Development, Digital Image Processing, Agile Software Development, Software Quality Assurance and Software Requirements Engineering.

M. Haseeb Nasir is an MS Scholar of Software Engineering at University of Engineering &Technology, Taxila. He is currently serving as a graduate research assistant at the same institute. He completed his Bachelors in Software Engineering in July, 2013. His areas of interest include Object Oriented Programming, Software Testing and Quality assurance, Mobile Application Development and Agile Methodologies.

Engr. Ali Javed is serving University of Engineering & Technology, Taxila, Pakistan, as an Assistant Professor. He is currently enrolled as a PhD scholar in same institution. He has completed his MS Engineering with Gold Medal from the same institute. He has more than 25 publications in world's most reputed journals. His areas of interest are computer vision, digital image processing, software quality assurance, software project management and mobile application development.