BAB II
LANDASAN TEORI

II.1 Kualitas Perangkat Lunak

Pengertian dari kualitas dalam perangkat lunak adalah "tingkat dimana suatu sistem, komponen, atau proses dapat memenuhi kebutuhan pengguna atau harapan pengguna [3]. Armand Vallin Feigenbaum menjelaskan dalam sudut pandangnya mengenai kualitas bahwa "Kualitas adalah keputusan pengguna, bukan keputusan bagian teknisi, bukan keputusan bagian pemasaran, juga bukan keputusan bagian manajemen. Itu didasarkan pada pengalaman sesungguhnya pengguna dengan produk dan jasa, diukur terhadap persyaratannya, dinyatakan atau tidak dinyatakan, sadar atau hanya dirasakan, secara teknis operasional atau seluruhnya subjektif, dan selalu mewakili target yang bergerak dalam pasar yang kompetitif." Lalu Feigenbaum menarik kesimpulan dengan pengertian dari produk dan kualitas layanan adalah total produk komposit dan karakteristik layanan pemasaran, rekayasa, manufaktur dan pemeliharaan dimana produk dan layanan yang digunakan akan memenuhi harapan pengguna [4].

"Memenuhi kebutuhan pengguna" adalah definisi inti dari kualitas menurut penjelasan Feigenbaum, dimana kualitas berhubungan erat dengan kepuasan pengguna, kualitas itu *multi dimensional* (harus didefinisikan secara komprehensif), sangat jelas bahwa kualitas tidak hanya mencakup manajemen dari produk dan jasa tapi juga pengguna dan harapan pengguna, berdasarkan pemaparan Feigenbaum.

II.2 Pengukuran

Dalam setiap disiplin rekayasa teknik pengukuran menjadi salah satu hal yang sering digunakan dalam penelitian, begitu juga pada disiplin teknik rekayasa perangkat lunak. Guna mendapatkan suatu hasil penelitian dari objek yang diteliti, maka dilakukan perhitungan dengan menggunakan tahapan tahapan dan perhitungan tertentu sehingga didapatkan suatu nilai yang dapat menggambarkan bagaimana kondisi atau tingkat suatu variabel dalam objek yang dilakukan

penelitian. Sedangkan pengertian dari pengukuran itu sendiri adalah proses menetapkan nomor atau kategori untuk suatu entitas untuk menggambarkan suatu atribut dari entitas tersebut [5].

II.2.1 Pengukuran Perangkat Lunak

Pengukuran dalam rekayasa perangkat lunak banyak memiliki hubungan dengan memberikan nilai numerik perhitungana pada atribut-atribut dari produk perangkat lunak atau proses perangkat lunak [6]. Pengukuran perangkat lunak dapat digunakan dengan membandingkan nilai pengukuran dengan perangkat lunak lainnya atau dengan standar yang telah disetujui dan berlaku di seluruh organisasi sehingga, bisa didapatkan suatu kesimpulan mengenai kualitas perangkat lunak tersebut. Untuk mengukur perangkat lunak terdapat dua cara yang dapat digunakan yaitu:

1. Membuat prediksi umum tentang sistem.

   Dengan mengukur karakteristik komponen sistem dan kemudian menggabungkan pengukuran tersebut, bisa didapatkan estimasi umum dari beberapa atribut sistem seperti jumlah kesalahan dalam sistem.
2. Mengidentifikasi komponen anomali.

   Pengukuran dapat mengidentifikasi komponen individu yang karakteristiknya menyimpang dari beberapa norma. Seperti mengukur komponen untuk menemukan karakteristik yang memiliki kompleksitas tertinggi, dan berasumsi karakteristik ini cenderung memiliki lebih banyak kesalahan, fokus pada komponen ini selama proses pemeriksaan [6].

Sedangkan dalam kualitas perangkat lunak, kualitas perangkat lunak diukur dengan metode penjumlahan dari keseluruhan kriteria dalam suatu faktor sesuai dengan bobot (weight) yang telah ditetapkan [7]. Sebuah formula pembobotan menunjukkan hubungan tertentu antara faktor dan atribut kriteria, setiap kriteria diberi nilai pembobotan untuk menunjukkan persentase kontribusi terhadap tujuan keseluruhan faktor. Formula bobot signifikan ketika menentukan skor dan ketika menentukan kebutuhan. Ketika menentukan skor, skor kriteria dikalikan dengan nilai bobot dalam menghitung skor faktor. Nilai bobot yang lebih tinggi

ditekankan pada nilai kriteria yang lebih tinggi untuk mencapai tujuan faktor penilaian **n**ilai bobot yang lebih rendah tidak telalu ditekankan pada pencapaian nilai kriteria yang tinggi. Ketika menentukan persyaratan, nilai pembobotan menunjukkan jumlah penekanan pada karakteristik perangkat lunak (kriteria) dalam mengembangkan produk [7].

II.2.2 Metrik Perangkat Lunak

Dalam pengukuran yang dilakukan pada penelitian tentunya dibutuhkan suatu metode dan skala pengukuran yang tepat agar dapat diketahui hasil dari pengukuran memiliki nilai yang sesuai dengan kondisi nyata objek penelitian. Metode dan skala objek yang digunakan dalam pengukuran tersebut merupakan komponen dari metrik. Sedangkan pengertian dari metrik adalah sebuah skala pengukuran dan metode yang digunakan untuk pengukuran [5]. Maka dalam penelitian ini akan digunakan istilah metrik sebagai alat ukur.

Menurut Ian Sommervile produk metrik merupakan metrik yang berfokus dengan karakteristik dari perangkat lunak itu sendiri. Hubungan dari karakteristik perangkat lunak dan atribut kualitas dapat bermacam-macam tergantung pada proses pengembangan, teknologi dan tipe dari sistem perangkat lunak itu sendiri. Produk metrik dapat dibagi kedalam dua kelas yaitu:

1. *Dynamic metrics*

   Metrik yang didapatkan berdasarkan pengukuran yang dilakukan saat program di eksekusi. Metrik dinamis membantu untuk menilai efisiensi dan keandalan dari perangkat lunak. Metrik dinamis biasanya cukup erat kaitannya dengan atribut kualitas perangkat lunak. Cukup mudah untuk mengukur waktu eksekusi yang diperlukan untuk fungsi tertentu dan untuk menilai waktu yang dibutuhkan untuk memulai sistem. Ini berhubungan langsung dengan efisiensi sistem. Demikian pula, jumlah kegagalan sistem dan jenis kegagalan dapat diketahui dan terkait langsung dengan keandalan perangkat lunak.

2. *Static metrics*

   Matrik yang didapatkan oleh pengukuran yang dilakukan pada representasi

sistem seperti desain, program atau dokumentasi. Metrik statis membantu menilai *complexity, understandability* dan *maintainability* sistem perangkat lunak. Metrik statis, di sisi lain, memiliki hubungan langsung dengan atribut kualitas. Sejumlah besar metrik ini telah diusulkan, dan banyak percobaan sudah mencoba untuk menurunkan dan memvalidasi hubungan antara metrik ini dengan *complexity, understandability* dan *maintainability* [6].

II.3 Domain Perangkat Lunak

Domain perangkat lunak yang ada saat ini terdapat dalam 7 kategori yaitu :

1. *System software*

   Adalah kumpulan program yang berfungsi sebagai pendukung untuk melayani program lain contohnya adalah *compiler, editor, drivers, networking software,* dan lain-lain.

2. *Application software*

   Program yang berdiri sendiri dan berfungsi untuk memenuhi kebutuhan bisnis tertentu. Seperti aplikasi untuk memfasilitasi operasi bisnis dan mengontrol fungsi bisnis secara *real time*. Contohnya *point-of-sale transaction processing, real-time manufacturing process control*, dan lain-lain.

3. *Engineering/scientific software*

   Program yang penggunaannya lebih kepada kebutuhan dalam penelitian di bidang teknik dan ilmiah, dan saat ini bahkan mulai dapat digunakan secara *real time*. Contohnya c*omputer-aided design, system simulation,* aplikasi interaktif lainnya.

4. *Embedded software*

   Program yang berada dalam sautu produk atau sistem dan digunakan untuk melaksanakan dan mengendalikan fitur dan fungsi untuk pengguna dan sistem itu sendiri. Program dalam domain ini dapat melakukan fungsi yang bersifat esoterik dan terbatas. Contoh *key pad control for a microwave oven, fuel control, dashboard displays, and braking systems*.

5. *Product-line software*

   Program yang dirancang agar dapat memiliki kemampuan khusus untuk digunakan oleh banyak pengguna yang berbeda. Contohnya *word processing, spreadsheets, computer graphics, multimedia, entertainment, database management,* dan *personal andbusiness financial applications.*

6. *Web applications*

   Meliputi beragam aplikasi yang dapat menyajikan informasi menggunakan teks dan grafis.

7. *Artificial intelligence software*

   Program yang memanfaatkan algoritma non numerik untuk memecahkan masalah yang kompleks yang tidak dapat diselesaikan dengan perhitungan, melainkan dengan analisis secara langsung. Contohnya *robotics, expert systems, pattern recognition (image and voice), artificial neural networks, theorem proving,* dan *game playing.*

II.4 Perangkat Lunak Pengolah Vektor Dua Dimensi

Gambar vektor atau grafik vektor merupakan gambar digital yang berbasiskan persamaan matematis. Gambar vektor terdiri dari penggabungan koordinat-koordinat titik menjadi garis atau kurva untuk kemudian menjadi sebuah objek, sehingga gambar tidak menjadi pecah walaupun diperbesar atau diperkecil. Gambar vektor umumnya memiliki ukuran yang lebih kecil bila dibandingkan dengan gambar bitmap. Beberapa format gambar vektor di antaranya SVG, EPS, dan CDR.

Keunggulan grafis bertipe vektor apabila dilakukan perbesaran tidak menghasilkan grafis yang pecah karena vektor tidak terikat oleh tingkat resolusi piksel, begitu juga saat dilakukan proses pencetakan maka akan dihasilkan gambar yang tidak akan pecah, tidak seperti gambar dengan jenis bitmap yang kualitasnya sangat tergantung terhadap resolusi piksel dari gambar itu sendiri, semakin kecil resolusinya apabila diperbesar dan dilakukan pencetakan yang ukurannya besar maka akan menghasilkan gambar yang pecah. Maka dari itu vektor banyak

digunakan oleh para desainer untuk mendesain berbagai kebutuhan dalam multimedia seperti iklan dan kebutuhan *public design* lainnya.

Salah satu perangkat lunak yang digunakan untuk mendesain gambar dengan berbasis vektor dua dimensi adalah *Inkscape*, perangkat lunak pengolah vektor dua dimensi ini bersifat *open source*. Perangkat lunak bersifat *open source* selain berlisensi *free*, juga memiliki ruang lingkup pengguna yang luas, hal ini dikarenakan alasan untuk menggunakan perangkat lunak open source adalah penghematan biaya, waktu ke pasar perangkat lunak cepat dan perangkat lunak berkualitas tinggi [8].

II.5 Model Penilaian Kualitas Perangkat Lunak

Model penilaian kualitas perangkat lunak yang dapat digunakan salah satunya adalah model ISO 9126. Model ISO 9126 yang digunakan pada penelitian ini adalah hasil revisi dan peningkatan dari model sebelumnya yaitu ISO 9126 (1991). Terdapat beberapa perbedaan utama di dalam model ISO 9126 dan model sebelumnya, yaitu:

- Pengenalan dari sub-karakteristik yang normatif lebih banyak berdasarkan kepada sub-karakteristik yang informatif pada model sebelumnya.
- Spesifikasi pada model kualitas
- Pengenalan pada *quality in use*.
- Pemindahan proses evaluasi pada model sebelumnya ke standar ISO 14598 mengenai evaluasi produk perangkat lunak
- Kesesuaian konten dengan standar ISO 14598-1.

Dalam model ini memungkinkan kualitas produk perangkat lunak di spesifikasikan dan di evaluasi dari berbagai sudut pandang yang berbeda oleh faktor faktor terkait yang dapat mempengaruhi yaitu akuisisi, kebutuhan, pengembangan, penggunaan, evaluasi, dukungan, perawatan, jaminan kualitas dan audit software. Hal ini dapat digunakan oleh pengembang, *acquirer*, staf jaminan kualitas dan evaluator independen, terutama yang bertanggung jawab untuk menentukan dan mengevaluasi kualitas produk perangkat lunak [9]. Selain itu, terdapat beberapa standar internasional lainnya yang memiliki hubungan dengan

ISO 9126, hal ini dikarenakan terdapat beberapa keterkaitan antara model ISO 9126 dan beberapa standar lain dalam bidang perangkat lunak, seperti diantaranya adalah ISO 15504 yang berfokus pada standar mengenai penilaian proses perangkat lunak, ISO 12207 yang berfokus pada standar siklus hidup perangka lunak, dan ISO 9001 yang berfokus kepada jaminan kualitas proses.

Model ISO 9126 terdiri kedalam dua bagian utama yaitu pertama *internal quality, external quality* dan kedua adalah *quality in use.* Hubungan antar bagian dalam model ini yaitu memiliki hubungan yang bersifat *sequential* dimana kualitas produk memiliki kontribusi dalam meningkatkan *quality in use* pada perangkat lunak, demikian juga evaluasi atribut dalam *quality in use* dapat memberikan pengaruh untuk meningkatkan kualitas suatu produk perangkat lunak, sedangkan evaluasi sebuah produk perangkat lunak akan menghasilkan pengaruh bagi peningkatan kualitas proses perangat lunak. Pada bagian pertama model kualitas yaitu terdapat kualitas internal dan eksternal yang terbagi lagi kedalam enam karakteristik utama yang dispesifikasikan kedalam sub-karakteristik.

Kualitas eksternal adalah totalitas karakteristik produk perangkat lunak dari sudut pandang eksternal. Kualitas ketika perangkat lunak dieksekusi, biasanya diukur dan dievaluasi saat uji coba di lingkungan simulasi dengan data yang disimulasikan menggunakan metrik eksternal [9]. Sub-karakteristik pada kualitas eksternal dapat diwujudkan secara eksternal ketika perangkat lunak digunakan sebagai bagian dari sistem komputer, dan sebagai hasil dari atribut internal perangkat lunak. Diagram yang menggambarkan model ISO 9126 terdapat pada Gambar II-1.



Gambar II-1 Diagram Model Kualitas ISO 9126

Penjelasan mengenai karakteristik dan sub-karakteristik dari model ISO 9126 yaitu:

1. *Functionality*

   Kemampuan produk perangkat lunak untuk memberikan fungsi yang memenuhi kebutuhan pengguna ketika perangkat lunak digunakan dalam kondisi tertentu.

   a. Suitability

   Kemampuan produk perangkat lunak dalam memberikan rangkaian fungsionalitas untuk kebutuhan tertentu dan sesuai dengan tujuan pengguna.

   b. *Accuracy*

   Kemampuan produk perangkat lunak dalam menyajikan hasil yang benar, dan disetujui sesuai dengan kebutuhan dalam hal presisi.

   c. *Interoperability*

   Kemampuan produk perangkat lunak untuk berinteraksi dengan satu atau lebih sistem tertentu.

d. *Security*

Kemampuan produk perangkat lunak untuk melindungi informasi dan data, sehingga pihak pengguna atau sistem yang tidak memiliki hak tidak dapat membaca atau mengubah informasi dan data, dan hanya dapat di akses oleh pihak yang memiliki wewenang.

e. *Functionality compliance*

Tingkat dimana produk perangkat lunak memenuhi standar atau konvensi yang berkaitan dengan fungsionalitas.

2. *Reliability*

Sejauh mana produk perangkat lunak dapat mempertahankan tingkat kinerja tertentu bila digunakan dalam kondisi tertentu.

a. *Maturity*

Kemampuan produk perangkat lunak untuk menghindari kegagalan sebagai akibat dari kesalahan dalam perangkat lunak.

b. *Fault Tolerance*

Sejauh mana produk perangkat lunak dapat mempertahankan tingkat tertentu dalam kinerja ketika terjadi kesalahan perangkat lunak atau kesalahan antarmuka tertentu.

c. *Recoverability*

Tingkat dimana produk perangkat lunak dapat bekerja kembali pada tingkat tertentu memulihkan data secara langsung yang telah terkena dampak dalam kasus kegagalan.

d. *Reliability compliance*

Sejauh mana produk perangkat lunak mematuhi standar, konvensi atau regulasi dengan perangkat lunak lain dalam keandalan.

3. *Usability*

Kemampuan produk software untuk dipahami, dipelajari, digunakan dan menarik bagi pengguna, ketika digunakan dalam kondisi tertentu.

a. *Understandability*

Kemampuan produk perangkat lunak yang memungkinkan pengguna untuk memahami apakah perangkat lunak ini cocok, dan bagaimana perangkat lunak dapat digunakan dalam kebutuhan dan kondisi tertentu.

b. *Learnability*

Tingkat dimana produk perangkat lunak memungkinkan pengguna untuk mempelajari penerapannya.

c. *Operability*

Kemampuan produk perangkat lunak dapat memungkinkan pengguna untuk mengoperasikan dan mengendalikannya.

d. *Attractiveness*

Tingkat dimana produk software menarik bagi pengguna.

e. *Usability compliance*

Sejauh mana produk perangkat lunak mematuhi standar, konvensi dengan perangkat lunak lain dalam yang berkaitan dengan penggunaan.

4. *Efficiency*

Sejauh mana produk perangkat lunak menyediakan kinerja yang tepat, relatif dengan jumlah sumber daya yang digunakan, di bawah kondisi tertentu.

a. *Time behaviour*

Sejauh mana produk perangkat lunak memberikan respon yang tepat dan

Waktu pengolahan dan proses yang berhasil sesuai fungsinya dalam kondisi tertentu.

b. *Resource utilisation*

Sejauh mana produk perangkat lunak menggunakan jumlah dan jenis yang sesuai sumber daya ketika perangkat lunak melakukan fungsinya dalam kondisi tertentu.

c. *Efficiency compliance*

Tingkat dimana produk perangkat lunak memenuhi standar atau konvensi yang berkaitan dengan efisiensi.

5. *Maintainability*

   Kemampuan produk perangkat lunak untuk diubah. Perubahan dapat mencakup koreksi, perbaikan atau adaptasi dari perangkat lunak untuk perubahan komponen sekelilingnya, dengan kebutuhan dan spesifikasi fungsional.

   a. Analysability

   Kemampuan produk perangkat lunak untuk dapat didiagnosis kekurangan atau penyebab kegagalan dalam perangkat lunak, atau untuk bagian-bagian yang akan dimodifikasi untuk diidentifikasi.

   b. Changeability

   Kemampuan produk perangkat lunak untuk memeungkinkan perubahan tertentu untuk di implementasikan.

   a. *Stability*

   Kemampuan produk perangkat lunak untuk menghindari pengaruh yang tidah diharapkan dari perubahan yang dilakukan pada perangkat lunak.

   b. *Testability*

   Kemampuan produk perangkat lunak untuk memungkinkan perangkat lunak dapat di ubah dan di validasi.

   c. *Maintainabilityy compliance*

   Sejauh mana produk perangkat lunak mematuhi standar, konvensi, peraturan yang berkaitan dengan pemeliharaan.

6. *Portability*

   Kemampuan produk perangkat lunak untuk beralih dari suatu lingkungan sistem ke lingkungan lainnya.

   a. *Adaptability*

Kemampuan produk perangkat lunak untuk menyesuakan diri dengan lingkungan sistem yang berbeda.

b. *Installability*

Kemampuan produk perangkat lunak untuk dapat digunakan dalam lingkungan sistem tertentu.

c. *Co-existence*

Kemampuan produk perangkat lunak untuk dapat bekerja secara berdampingan dengan perangkat lunak lainnya dan berbagi sumber daya.

d. *Replaceability*

Kemampuan produk perangkat lunak untuk dapat digunakan dalam perangkat lunak lain untuk tujan yang sama dan dalam ligkungan sistem yang sama.

e. *Portability compliance*

Sejauh mana produk perangkat lunak mematuhi standar, konvensi atau peraturan yang berkaitan dengan portabilitas.

Dari keseluruhan karakteristik kualitas eksternal terdapat diantara setiap karakteristik yaitu *compliance*, pada penelitian ini *compliance* tidak dilibatkan karena berhubungan dengan pemenuhan standar, konvensi dan peraturan pada masing-masing karakteristik.

Pada bagian kedua model kualitas ISO 9126 yaitu *quality in use* adalah sudut pandang pengguna terhadap kualitas produk perangkat lunak bila digunakan dalam lingkungan dan konteks penggunaan tertentu. Lebih dominan dalam mengukur sejauh mana pengguna dapat mencapai tujuan mereka dalam menggunakan perangkat lunak pada kondisi tertentu daripada mengukur atribut dari perangkat lunak itu sendiri [9]. *Quality in use* merupakan kombinasi dari enam karakteristik kualitas di bagian pertama yang berpengaruh bagi pengguna. Terdapat empat karakteristik dalam *quality in use* yaitu *effectiveness, productivity, safety,* dan *satisfaction.* Bagan *quality in use* terdapat dalam Gambar II-2.



Gambar II-2 Diagram *quality in use* ISO 9126

Penjelasan mengenai karakteristik yang ada pada *quality in use* adalah sebagai berikut:

1. *Effectiveness*

   Kemampuan perangkat lunak untuk memungkinkan pengguna mencapai tujuan yang diinginkan dengan keakuratan dan kelengkapan dalam kondisi tertentu.

2. *Productivity*

   Kemampuan perangkat lunak untuk memungkinkan pengguna menggunakan jumlah yang tepat dari sumber daya yang berhubungan dengan efektivitas yang dapat diraih dalam kondisi tertentu.

3. *Safety*

   Kemungkinan perangkat lunak untuk mencapat tingkat resiko yang masih dapat diterima yang memiliki kemungkinan membahayakan bagi orang-orang, perangkat lunak, bisnis, benda kepemilikan dan lingkungan dalam kondisis tertentu.

4. *Satisfaction*

   Kemampuan perangkat lunak untuk memberikan kepuasan terhadap pengguna dalam kondisi tertentu.

II.6 Metode Pengukuran Perangkat Lunak

Metode pengukuran yang digunakan untuk mengukur nilai kualitas perangkat lunak adalah metrik pada kualitas eksternal dan *quality in use* yang ada pada model ISO 9126. Berikut ini tabel metrik kualitas eksternal beserta penjelasan tujuan pengukurannya yang digunakan dalam metrik ISO 9126:

Tabel II-1 Metrik Eksternal Model ISO 9126

Karakteristik Kualitas	Sub-karakteristik Kualitas	*Metric External*	Tujuan dari metrik	Persamaan metrik
Functionality	*Suitability*	*Functional adequacy*	Seberapa memadai fungsi yang di evaluasi	X=1-A/B
		Functional specification stability (volatility)	Seberapa stabil fungsional setelah dieksekusi	X = 1- A / B
		Functional implementation coverage	Seberapa tepat implementasi dari fungsional	X=1- A / B
		Functional implementation completeness	Seberapa lengkap implementasi menurut spesifikasi kebutuhan	X = 1 - A / B
	Accuracy	*Accuracy expectation*	Adalah perbedaan hasil antara kenyataan dan yang diharapkan yang masih bisa diterima	X=A / T
		Computational accuracy	Seberapa sering pengguna akhir mendapatkan hasil yang tidak akurat	X=A / T
		Precision	Seberapa sering pengguna akhir mendapatkan hasil yang tidak cukup presisi	X=A / T
	Interoperability	*Data exchangeability (Data format based)*	Seberapa tepat pertukaran fungsi antarmuka untuk data tertentu dalam proses pemindahan	X= A / B

		Data exchangeability (User's success attempt based)	Seberapa sering pengguna gagal saat melakukan pemindahan data antar perangkat lunak	X= 1 - A / B
	Security	*Access auditability*	Seberapa lengkap pemeriksaan pengguna mengenai sistem dan data	X= A / B
		Access controllability	Seberapa jauh akses yang dapat dilakukan terhadap sistem	X= A / B
		Data corruption prevention	Frekuensi kerusakan data	X= 1 – A / N Y= 1- B / N Z= A/T atau Z= B/T
Reliability	*Maturity*	*Estimated latent fault density*	Seberapa banyak masalah yang sering muncul yang dapat berpengaruh terhadap kegagalan di masa yang akan datang	X= {ABS(A1 - A2)} / B
		Failure density against test cases	Seberapa banyak kegagalan yang terdeteksi selama waktu yang telah ditentukan dalam kasus tertentu	X= A1 / A2
		Failure resolution	Seberapa banyak kondisi kegagalan yang terselesaikan	X= A1 / A2
		Fault density	Seberapa banyak kegagalan yang terdeteksi selama waktu yang telah	X= A / B

			ditentukan	
		Fault removal	Seberapa banyak kegagalan yang berhasi diperbaiki	X= A1 / A2 Y= A1 / A
		Mean time between failures (MTBF)	Frekuensi kegagalan dalam operasi perangkat lunak	X = T1 / A Y = T2 / A
		Test coverage (Specified operation scenario testing coverage)	Seberapa banyak kasus tes yang dibutuhkan yang telah dieksekusi selama pengetesan	X= A / B
		Test maturity	Apakah perangkat lunak teah teruji dengan baik	X= A / B
	Fault Tolerance	*Breakdown avoidance*	Seberapa sering perangkat lunak menyebabkan kerusakan dari keseluruhan ruang lingkup produksi	X= 1- A / B
		Failure avoidance	Seberapa banyak kegagalan yang bersifat kritis dan serius	X=A / B
		Incorrect operation avoidance	Seberapa banyak fungsi yang terimplementasi dengan kapabilitas penghindaran operasi yang salah	X=A / B
	Recoverability	*Availability*	Seberapa siapkah sistem untuk digunakan dalam waktu tertentu	X= { To / (To + Tr) } Y= A1 / A2

		Mean down time	Rata-rata waktu sistem tidak tersedia ketika terjadi kegagalan sebelum dimulai	X= T / N
		Mean recovery time	Rata-rata waktu sistem untuk menylesaikan perbaikan dari perbaikan awal	X= Sum(T) / B
		Restartability	Seberapa sering sistem dapat memulai kembali layanan yang disediakan untuk pengguna dalam waktu yang dibutuhkan	X = A / B
		Restorability	Seberapa sanggup produk dalam mengembalikan seperti semula setelah terdapat kejadian tidak normal	X= A / B
		Restore effectiveness	Seberapa efektif kesanggupan untuk mengembalikan seperti semula	X= A / B
Usability	*Understandability*	*Completeness of description*	Bagian fungsi yang dapat dimengerti setelah membaca deskripsi produk	X = A / B
		Demonstration accessibility	Bagian panduan yang dapat diakses pengguna	X = A / B
		Demonstration accessibility in use	Bagian panduan yang dapat diakses pengguna ketika pengguna sangat membutuhkan dalam penggunaan	X = A / B
		Demonstration	Bagian fungsi yang dapat pengguna	X = A / B

		effectiveness	gunakan dengan baik setelah pemanduan	
		Evident functions	Bagian fungsi yang dapat diidentifikasi oleh pengguna berdasarkan kondisi awal	X = A / B
		Function understandability	Bagian dari fungsi produk yang dapat dimengerti oleh pengguna dengan baik	X= A / B
		Understandable input and output	Dapatkah pengguna mengerti apa yang dibutuhkan sebagai data masukan dan hasi keluaran dari perangkat lunak	X= A / B
	Learnability	*Ease of function learning*	Berapa lama waktu yag pengguna butuhkan untuk dapat mempelajari satu fungsi	T= ?
		Ease of learning to perform a task in use	Berapa lama waktu yag pengguna butuhkan untuk dapat menjalankan fungsi tertentu	T= ?
		Effectiveness of the user documentation and/or help system	Bagian dari penggunaan yang dapat digunakan dengan baik setelah menggunakan dokumentasi pengguna	X= A / B
		Effectiveness of user documentation and/or help system	Bagian dari fungsi yang dapat digunakan dengan baik setelah menggunakan dokumentasi pengguna	X= A / B

		in use		
		Help accessibility	Bagian dari topik *help* yang dapat pengguna temukan	X= A / B
		Help frequency	Seberapa sering pengguna mengakses *help* untuk mempelajari penggunaan dalam menyelesaikan tugas mereka	X= A / B
	Operability	*Operational consistency in use*	Seberapa konsisten komponen dari antarmuka	X = 1 - A / B Y = N / UOT
		Error correction	Dapatkah pengguna dengan mudah memperbaiki kesalahan operasi	T= Tc - Ts
		Error correction in use	Dapatkah pengguna dengan mudah mengembalikan kedalam keadaansemula dalam kegagalan atau operasi untuk dicoba kembali	X= A / UOT X = A / B
		Default value availability in use	Daparkah pengguna dengan muda memilih nilai prameter yang tepat untuk penggunaan mereka	X = 1 - A / B
		Message understand-ability in use	Dapatkah pengguna dengan mudah mengerti pesan dari sistem perangkat lunak	X = A / UOT
		Self-explanatory	Dalam kesalahan tertentu apakah	X= A / B

		error messages	pengguna melakukan tindakan perbaikan yang tepat	
		Operational error recoverability in use	Dapatkah pengguna dengan mudah mengembalikan keadaan ke-semula dalam kondisi terburuk mereka	X = 1 - A / B
		Time between human error operations in use	Dapatkah pengguna mengoperasikan perangkat lunak dalam waktu yang lama tanpa adanya *human error*	X = T / N (at time t during [t-T, t])
		Undoability (User error correction)	Seberapa sering pengguna berhasil memperbaiki kesalahan pemasukan data	X= A / B Y= A / B
		Customisability	Dapatkah pengguna dengan mudah mengubah prosedur penggunaan sesuai dengan kenyamanan mereka	X= A / B
		Operation procedure reduction	Dapatkah pengguna dengan mudah mengurangi prosedur penggunaan untuk kenyamanan mereka	X = 1 - A / B
		Physical accessibility	Bagian dari fungsi yang dapat diakses pengguna dengan keterbatasan tertentu	X= A / B
	Attractiveness	*Attractive interaction*	Seberapa atraktif antarmuka bagi pengguna	Kuesioner untuk menilai daya tarik dari antarmuka

				untuk pengguna, setelah pengalaman pemakaian
		Interface appearance customisability	Bagian dari antarmuka yang dapat diubah dalam tampilan sesuai dengan kepuasan pengguna	X= A / B
Efficiency	*Time Behaviour*	*Response time*	Waktu yang dibutuhkan untuk menyelesaikan suatu operasi	T = ?
		Response time (Mean time to response)	Rata-rata waktu yang dibutuhkan pengguna untuk menunggu setelah menyelesaikan sebuah oeprasi	X = Tmean / TXmean
		Response time (Worst case response time ratio)	Batas absolute waktu untuk mennyelesaikan sebuah fungsi	X= Tmax / Rmax
		Throughput (Mean amount of throughput)	Rata rata jumlah operasi yang dapat sistem tangani dalam waktu tertentu	X = Xmean / Rmean
		Throughput (Worst case throughput ratio)	Batas absolut sistem dapat menangani operasi	X = Xmax / Rmax
		Turnaround time	Berapa operasi yang dapat berhasi diselesaikan dalam waktu tertentu	T = ?

		Turnaround time (Mean time for turnaround)	Rata-rata waktu tunggu pengguna untuk menyelesaikan kumpulan operasi	X = Tmean/TXmean
		Turnaround time (Worst case turnaround time ratio)	Waktu tunggu pengguna untuk menyelesaikan kumpulan operasi	X= Tmax / Rmax
		Waiting time	Bagian dari waktu yang digunakan pengguna untuk menunggu sistem merespon	X = Ta / Tb
		Throughput	jumlah operasi yang dapat sistem tangani dalam waktu tertentu	X = A / T
	Resource Utilization	*I/O devices utilisation*	Apakah perangkat I/O ter lalu tinggi menyebabkan tidak efisien	X = A / B
		I/O loading limits	Batas absolut penggunaan I/O untuk menyelesaikan fungsi	X = Amax / Rmax
		I/O related errors	Seberapa sering pengguna mengalami masalah dalam perangkat I/O	X = A / T
		Mean I/O fulfillment ratio	Rata rata jumlah I/O yang berhubungan dengan pesan kegagalan yang melebihi batas waktu tertentu	X = Amean / Rmean
		User waiting time of I/O devices	Dampak penggunaak I?O dalam waktu tunggu pengguna	T = ?

		utilisation		
		Maximum memory utilisation	Batas absolut memori dalam penggunaan untuk menyelesaikan fungsi	X = Amax / Rmax
		Mean occurrence of memory error	Rata rata jumlah memori yan gberhubungan dengan pesan kegagalan dalam waktu tertentu	X = Amean / Rmean
		Ratio of memory error/time	Seberapa banyak kegagalan dalam memori yang dialami dalam waktu tertentu	X = A / T
		Maximum transmission utilisation	Batas absolut transimisi yang dibutuhkan untuk dapat menyelesaikan fungsi	X = Amax / Rmax
		Media device utilisation balancing	Derajat sinkronisasi antara media dalam waktu tertentu	X = SyncTime/T
		Mean occurrence of transmission error	Rata rata jumlah transmisi yang berhubungan dengan pesan kesalahan dalam waktu tertentu	X = Amean / Rmean
		Mean of transmission error per time	Berapa banyak transimi yang berhubungan dengan pesan kegagalan yang dialami dalam waktu tertentu	X = A / T
		Transmission capacity utilisation	Apakah perngkat lunak memiliki kemampuan untuk menjalankan	X = A / B

			fungsi sesuai dengan kapasitas transmisi yang diharapkan	
Maintainability	*Analyzability*	*Audit trail capability*	Dapatkah pengguna mengidentifikasi operasi yang menyebabkan kegagalan	X= A / B
		Diagnostic function support	Seberapa mampu diagnosa fungsi dalam analisis pendukung	X= A / B
		Failure analysis capability	Dapatkah pengguna mengidentifikasi oeprasi yang menyebabkan kegagalan	X=1- A / B
		Failure analysis efficiency	Dapatkah pengguna secara efisien menganaisa penyebab kegagalan	X= Sum(T) / N
		Status monitoring capability	Dapatkah pengguna mengidentifikasi operasi tertentu yang menyebabkan kegagalan dengan memonitor data selama operasi	X= 1-A / B
	Changeability	*Change cycle efficiency*	Dapatkah pengguna menyelesaikan masalah untuk kepuasannya dengan waktu tertentu yang dapat diterima	Tav = Sum(Tu) / N Tu= Trc - Tsn
		Change implementation elapse time	Dapatkah pemelihara dengan mudah mengubah perangkat lunak untuk menyelesaikan	Tav = Sum(Tm) / N Tm=Tout - Tin

			kegagala	
		Modification complexity	Dapatkah pemelihara dengan mudah perangkat lunak untuk menyelesaikan masalah	T = Sum (A / B) / N
		Parameterised modifiability	Dapatkah pengguan atau pemelihara dengan muda mengubah parameter untuk mengubah perangkat lunak dalam menyelesaikan masalah	X=1- A / B
		Software change control capability	Dapatkah pengguna dengan mudah mengidentifikasi versi yang diubah	X= A / B
	Stability	*Change success ratio*	Dapatkah pengguna mengoperasikan perangkat lunak tanpa mengalami kegagalan setelah melakukan pemeliharaan	X= Na / Ta Y = { (Na / Ta) / (Nb / Tb) }
		Modification impact localisation (Emerging failure after change)	Dapatkah pemelihara dengan mudah mengurangi kegagalan yang disebabkan oleh pemeliharaan	X= A / N
	Testability	*Availability of built-in test function*	Dapatkah pengguna dan pemelihara dengan mudah melakukan operasi pengujian tanpa fasilitas pengujian tambahan	X= A / B
		Re-test	Dapatkah pengguna	X= Sum(T) / N

		efficiency	dan pemelihara melakukan operasi pengujian dan memutuskan kapan perangkat lunak siap digunakan atau tidak	
		Test restartability	Dapatkah pengguna dan pemelihara dengan mudah melakukan operasi pengujian dengan penandaan setelah pemeliharaan	X = A / B
Portability	*Adaptability*	*Adaptability of data structures*	Dapatkah pengguna memelihara dengan mudah menyesuaikan perangkat lunak kedalam set data di raung lingkup baru	X = A / B
		Hardware environmental adaptability	Dapatkah pengguna atau pemelihara dengan mudah menyesuaikan dengan perangkat lunak	X= 1 - A / B
		Organisational environment adaptability	Dapatkah pengguna atau pemelihara dengan mudah menyesuaikan perangkat lunak kedalam ruang lingkup	X= 1 - A / B
		Porting user friendliness	Dapatkah pengguna atau pemelihara degan mudah menyesuaikan perangkat lunak kedalam ruang lingkup	T=?
		System software environmental	Apakah perangkat lunak cukup mammpu untuk menyesuaikan dengan ruang lingkup	X= 1 - A / B

		adaptability	pengoperasiannya	
	Installability	*Ease of installation*	Dapatkah pengguna atau pemelihara dengan mudah memasang perangkat lunak dalam ruang lingkup operasi	X = A / B
		Ease of Setup Re-try	Dapatkah pengguna atau pemelihara dengan mudah mencoba ulang pemasangan perangkat lunak	X = 1 - A / B
	Co-existence	*Available co-existence*	Seberapa sering pengguna mendapatkan kegagalan yang tidak diduga ketika melakukan operasi secara bersamaan	X = A / T
	Replaceability	*Continued use of data*	Dapatkah pengguna atau pemelihara dengan mudah melanjutkan penggunaan dengan data yang sama setelah mengganti perangkat lunak dengan versi sebelumnya	X = A / B
		Function inclusiveness	Dapatkah pengguna atau pemelihara dengan mudah melanjutkan penggunaan pada fungsi yang sejenis setelah mengganti perangkat lunak dengan versi sebelumnya	X = A / B

		User support functional consistency	Seberapa konsisten komponen baru dengan antarmuka yang telah ada	X= 1 - A1 / A2

Sedangkan untuk metrik yang ada dalam *quality in use* pada model ISO 9126 adalah sebagai berikut:

Tabel II-2 Metrik Q*uality in use* Model ISO 9126

Karakteristik Kualitas	*Metric Quality in use*	Tujuan metrk	Persamaan Metrik
Effectiveness	*Task effectiveness*	Bagian dari pencapaian tujuan eksekusi tercapai dengan baik	$M1 = \|1-\Sigma A_i\|_1$
	Task completion	Bagian dari kelengkapan eksekusi	X = A/B
	Error frequency	Frekuensitingkat kegagalan	X = A/T
Productivity	*Task time*	Seberapa lama waktu yang dibutuhkan untuk menyelesaikan tugas	X = Ta
	Task efficiency	Seberapa efisienkah pengguna	X = M1 / T
	Economic productivity	Seberapa efektif biaya penggunanya	X = M1 / C
	Productive proportion	Bagian dalam waktu yan gpengguna gunakan untuk melakuka aksi produktif	X = Ta / Tb
	Relative user efficiency	Seberapa efisien pengguna dibandingkan dengan pengguna ahli	X = A / B
Safety	*User health and safety*	Kejadian dari masalah kesehatan diantara pengguna dari produk	X = 1-A / B
	Safety people affected by use of the system	Kejadian yang membahayakan bagi pengguna yang menggunakan	X = 1-A / B

		sistem	
	Economic damage	Kejadian yang membahayakan keuangan	X = 1-A/ B
	Software damage	Kejadian yang dapat menyebabkan kerusakan perangkat lunak	X = 1-A / B
Satisfaction	*Satisfaction scale*	Seberapa puaskah pengguna	X = A/B
	Satisfaction questionnaire	Seberapa puaskah pengguna dengan fitur perangkat lunak tertentu	X = Σ (Ai)/n
	Discretionary usage	Bagian yang membuat pengguna yang berpotensi untuk memilih untuk menggunakan sistem	X = A/B

Dalam penggunaannya, metrik-metrik pada tabel di atas merupakan kandidat metrik yang akan digunakan dalam pengukuran kualitas, hal ini dikarenakan penggunaan metrik disesuaikan dengan kriteria dan karakteristik yang berhubungan dengan perangkat lunak. Dengan begitu maka memungkinkan hanya beberapa saja metrik yang tercantum pada tabel yang akan dipilih dan digunakan dalam penelitian ini.

II.7 Metode Pengolahan Kuesioner

Kuesioner atau angket merupakan teknik pengumpulan data yang dilakukan dengan cara memberi seperangkat pertanyaan atau pernyataan tertulis kepada responden untuk dijawabnya [10]. Setiap pertanyaan akan diberikan nilai jawaban berbentuk skala dari nilai paling rendah hingga nilai tertinggi yang telah ditentukan, sehingga responden dapat memberikan jawaban dalam rentang nilai pada pilihan jawaban tersebut yang telah disesuaikan. Menurut Roscoe ukuran sampel yang layak digunakan dalam penelitian adalah antara 30 sampai dengan 500 [11]. Sehingga dalam penelitian ini jumlah data responden pengguna perangkat lunak pengolah vektor dua dimensi berjumlah minimal 30 hingga 500.

Dalam penelitian ini untuk mengolah data kuesioner dilakukan dengan menormalisasi terlebih dahulu data hasil kuesioner sesuai dengan skala yang dibutuhkan untuk data penelitian.