BAB II
LANDASAN TEORI

II.1 Definisi Perangkat Lunak

Perangkat lunak didefinisikan sebagai suatu intruksi yang terinci yang mengkontrol aktivasi dari sistem komputer. Tanpa adanya *software, hardware* komputer tidak dapat melakukan tugas yang terkait dengan komputer. Terdapat tiga fungsi utama dari *software*, yaitu: mengelola sumberdaya komputer yang dimiliki oleh suatu organisasi, sebagai alat bagi manusia dalam mengambil manfaat dari sumberdaya komputer yang dimiliki, bertindak sebagai perantara antara organisasi dengan sumberdaya data yang dimiliki. Sedangkan *software programs* merupakan serangkaian perintah atau instruksi ke komputer[7].

Menurut Kadir mengkelompokkan perangkat lunak secara umum menjadi dua, yaitu program aplikasi (*application program*) dan program sistem (*system program*)[8]. Sedangkan menurut *O'Brien* mengelompokkan *software* berdasarkan jenis dan fungsi utamanya menjadi dua, yaitu *software aplikasi* dan *software sistem* [9].

II.2 Pengukuran Perangkat Lunak

Menurut Sommerville, pengukuran terbagi atas dua, yaitu pengukuran (metrik) kontrol dan pengukuran (metrik) prediktor. Metrik kontrol biasanya dihubungkan dengan proses perangkat lunak, misalnya usaha dan waktu rata-rata yang dibutuhkan untuk memperbaiki cacat yang dilaporkan. Sedangkan metrik prediktor berhubungan dengan produk perangkat lunak, misalnya kompleksitas sikomatik modul, panjang rata-rata identifier pada program dan jumlah atribut dan operasi yang berhubungan dengan objek pada suatu rancangan.

Metrik juga dapat dipisahkan dalam dua kategori, yaitu metrik langsung dan metrik tidak langsung. Metrik langsung dalam proses rekayasa perangkat lunak

berhubungan dengan biaya dan sumber daya yang diperlukan, misalnya: pengukuran jumlah baris kode, kecepatan eksekusi, ukuran memori, dan kesalahan yang ditemui dalam suatu periode waktu. Metrik tidak langsung dari suatu produk berhubungan dengan fungsionalitas, kualitas, kompleksitas, efisiensi, reliabilitas, dan lain sebagainya. Pengukuran secara langsung lebih mudah dilakukan, karena hasil dapat diperoleh secara langsung, sedangkan pengukuran tidak langsung lebih sulit dilakukan, karena harus melalui proses yang lebih kompleks.[10].

II.3 Software Quality

Ada banyak yang berpendapat mengenai *software quality*, berikut beberapa pemaparan mengenai *software quality* menurut para ahli, diantaranya :

Menurut *The Institute of Electronic Engineer* (IEEE), kualitas perangkat lunak dapat diartikan sebagai tingkatan dimana sistem, komponen atau proses dapat memenuhi spesifikasi kebutuhan dan keinginan *user* atau klien. Selanjutnya, pressman mengungkapkan bahwa kualitas perangkat lunak adalah kesesuaian antara fungsionalitas dan kinerja sistem terhadap kebutuhan klien, standar dokumentasi pengembangan sistem yang telah ditentukan, dan karakterisktik implisit yang diharapkan pengembangan perangkat lunak[11].

Menurut definisi Steve McConnell, *code complette* membagi perangkat lunak ke dalam dua karakteristik, yaitu karakteristik kualitas internal dan eksternal. Karakteristik kualitas eksternal merupakan bagian-bagian dari sebuah produk yang berhubungan langsung dengan para penggunanya, sedangkan karakteristik kualitas internal tidak berhubungan secara langsung dengan pegguna.

Software quality didefinisikan sebagai kesesuaian yang diharapkan pada semua perangkat lunak yang dibangun berkaitan dengan fungsi perangkat lunak yang diutamakan dan unjuk kerja perangkat lunak, standar pembangunan perangkat lunak yang terdokumentasi, dan karakteristik yang ditunjukkan oleh perangkat lunak[12].

Menurut Agarwal dkk *software quality* merupakan sebagai kesesuaian terhadap kebutuhan performa dan fungsionalitas, standar pengembangan yang

terdokumentasi, serta karakteristik implisit dari sebuah perangkat lunak yang dikembangkan secara professional.

Secara umum, definisi *software quality* yang disebutkan oleh Pressman adalah sebagai proses perangkat lunak yang efektif diterapkan dengan cara yang menciptakan produk yang bermanfaat yang memberikan nilai terukur bagi mereka yang memproduksinya dan mereka yang menggunakannya. Proses dalam pembuatan sebuah barang dimana kita harus memastikan apakah barang tersebut sudah sesuai yang diharapkan atau belum, pengembangan perangkat lunak atau *software* juga menuntut hal yang sama. Metode yang dipakai dalam menganalisis kualitas perangkat lunak tersebut tentu saja berbeda dibandingkan dengan metode yang digunakan di pabrik-pabrik misalnya.

Pengujian adalah proses mengeksekusi program secara intensif untuk menemukan kesalahan-kesalahan. Pengujian tidak hanya untuk mendapatkan program yang benar, namun juga memastikan bahwa program tersebut bebas dari kesalahan-kesalahan untuk segala kondisi. Pengujian perangkat lunak adalah elemen kritis dari jaminan kualitas perangkat lunak dan mempresentasikan spesifikasi, desain dan pengkodean[13].

II.4 Domain Perangkat Lunak

Domain perangkat lunak merupakan kategori dari setiap jenis perangkat lunak yang ada. Terdapat tujuh kategori mengenai jenis domain perangkat lunak ini sendiri, diantaranya :

1. *System software*

   System software merupakan sebuah program yang dibuat untuk mendukung program lain untuk dapat digunakan. Perangkat lunak jenis ini misalnya *compilers, editor, file management, operating system, telecommunications processors,* dan lain-lain.

2. *Application software*

   Application software adalah sebuah program yang berdiri sendiri dan digunakan untuk mengatasi kebutuhan bisnis yang spesifik.

3. *Engineering/scientific software*

   Perangkat lunak pada domain ini biasanya ditekankan pada penggunaan algoritma. Penggunaan perangkat lunak ini terdapat pada kebutuhan seperti astronomi, vulkanologi, pabrik, biologi, dan lain sebagainya.

4. *Embedded software*

   Embedded software merupakan perangkat lunak yang ditanam pada suatu sistem. Perangkat lunak ini digunakan dalam mengatur fungsi untuk pengguna maupun untuk dirinya sendiri.

5. *Product-line software*

   Perangkat lunak pada domain *product-line software* dibuat untuk membantu kebutuhan pengguna yang bersifat spesifik yang dapat digunakan oleh pengguna yang berbeda. Contoh dari perangkat lunak pada domain *product-line software* diantaranya untuk keperluan *word processing, multimedia, computer graphic, database management, entertainment,* dan lain sebagainya.

6. *Web application*

   Web application atau biasa disebut *webapps* adalah perangkat lunak yang berbasis *website*. Pada perangkat lunak ini bukan hanya sekedar menampilkan informasi berbentuk teks namun dapat juga berupa gambar.

7. *Artificial intelligence software*

   Perangkat lunak pada domain ini ditekankan pada algoritma untuk dapat menyelesaikan suatu masalah yang kompleks, yang tidak bisa diselesaikan dengan perhitungan ataupun analisis langsung. Perangkat lunak ini seperti untuk pengenalan pola, jaringan syaraf tiruan, robotik, dan lain-lain.

II.5 Perangkat Lunak *Video Editor*

Video editing dengan menggunakan komputer atau disebut juga dengan *offline editing* atau *non linier editing* sudah semakin maju dan menarik. Hal ini ditandai dengan semakin tingginya kemampuan perangkat lunak untuk keperluan editing, video. Versi terakhir dari perangkat lunak *video editor* bahkan sudah mampu untuk menghasilkan *file video* dalam format MPEG-2 yang merupkan standar baku DVD. Video editing sangat berkaitan dengan keterampilan-keterampilan lain dengan sangat erat seperti grafis visual, audio editing, *special effect*, serta animasi 2D/3D.

Dibidang editing video dan film, terdapat banyak sekali pilihan perangkat lunak yang dapat digunakan, baik berbayar, *open source*, maupun *freeware*, disertai tambahan efek-efek memukau. Salah satu perangkat lunak yang digunakan oleh para editor untuk pengeditan video adalah openshot.

Openshot *video editor* merupakan salah satu perangkat lunak berbasis open source yang ada di *operating system* Linux, namun sekarang sudah tersedia untuk *operating system* windows. Openshot merupakan proyek perangkat lunak *open source* di bidang multimedia yang dikembangkan oleh Jonathan Thomas dan tim mulai tahun 2008. Openshot memfokuskan diri untuk menyediakan perangkat lunak *video editor* yang gratis, stabil, dan mudah digunakan[15]. Dengan kata lain siapapun bisa ikut serta untuk membantu mengembangkan perangkat lunak ini, karena pihak *developer* membebaskan pengguna untuk ikut serta mengubah, menghapus, ataupun menambanhkan kekurangan yang ada pada perangkat lunak tersebut agar kedepannya dapat menjadi lebih baik.

II.6 Model Kualitas Perangkat Lunak

Dalam rekayasa perangkat lunak terdapat beberapa model kualitas perangkat lunak. Pada setiap model ini terdapat beberapa faktor yang menjadi poin-poin utama dalam penilaian kualiras sebuah perangkat lunak. Model kualitas perangkat lunak yang akan digunakan dalam penilaian kualitas sebuah perangkat lunak adalah model ISO-9126. Model ini merupakan bagian dari *Software product Quality Requirements*

and Evaluation (SQuaRE), dimana model ini berkaitan dengan model kualitas perangkat lunak yang merupakan pengembangan dari model sebelumnya. Pada model ini terdapat beberapa sub-karakteristik tambahan dan beberapa sub-karakteristik yang dipindahkan ke karakteristik lain. Berikut ini merupakan karakteristik atau faktor kualitas internal dan eksternal yang terdapat pada model ISO-9126, model ISO-9126 terdiri dari dua bagian model kualitas untuk kualitas produk perangkat lunak, diantaranya:

1. Model Kualitas
2. Metrik Eksternal
3. Metrik Internal
4. Quality in use model

Bagian pertama dari kualitas model terebut menentukan karakteristik yang dibagi dalam beberapa sub karakteristik untuk kualitas internal dan kualitas eksternal yang dapat dilihat pada tabel II-1.

Tabel II-1 Faktor Kualitas Internal dan Eksternal

Internal and External Quality	
Faktor	**Sub-Faktor**
Functionality	*Suitability* *Accuracy* *Interoperability* *Security* *Functionality Compliance*
Reability	*Maturity* *Fault Tolerance* *Recoverability* *Reliability Compliance*
Usability	*Understandability* *Learnability* *Operability* *Attractiveness*

Internal and External Quality	
Faktor	**Sub-Faktor**
	Usability Compliance
Efficiency	*Time Behavior* *Resource Utilization* *Efficiency Compliance*
Maintainability	*Analyzability* *Changeability* *Stability* *Maintainability Compliance*
Portability	*Adaptability* *Instability* *Co-existence* *Replaceability* *Portability Complaince*

Berikut ini merupakan pengertian dari masing-masing faktor dan sub-faktor yang terdapat pada model *internal* dan *eksternal* ISO-9126, antara lain:

1. *Functionality*

   Faktor ini berhubungan dengan seberapa jauh sebuah perangkat lunak dapat memenuhi kebutuhan penggunanya.

   a. *Suitability*

      Sub-faktor *suitability* berhubungan dengan tingkat kemampuan dan kelayakan dari sebuah perangkat lunak untuk dapat menyediakan fungsionalitas untuk kebutuhan yang spesifik.

   b. *Accuracy*

      Accuracy berhubungan dengan kemampuan sebuah perangkat lunak untuk dapat memberikan hasil yang benar dan sesuai dengan fungsi yang dijalankan.

c. *Interoperability*

Sub-faktor *interoperability* menggambarkan tentang kemampuan sebuah perangkat lunak untuk dapat berinteraksi dengan sistem yang lain atau dengan sistem tertentu.

d. *Security*

Sub-faktor ini berhubungna dengan keamanan yang dimiliki oleh sebuah perangkat lunak. Keamanan yang dimaksud dapat berupa pemberian hak akses kepada penggunanya.

e. *Functionality Compliance*

Functionality compliance menggambarkan tentang kesesuaian antara fungsionalitas yang ada pada sebuah perangkat lunak dengan aturan standar yang berlaku pada perangkat lunak lainnya yang sejenis.

2. *Reliability*

Faktor *reliability* berhubungan dengan keandalan atau seberapa baik tingkat pelayanan sebuah perankat lunak apabila digunakan dalam kondisi tertentu.

a. *Maturity*

Maturity berhubungan dengan kelayakan sebuah perangkat lunak dalam menangani kegagalan atau kesalahan yang terdapat didalamnya.

b. *Fault Tolerance*

Fault tolerance merupakan sub-faktor *reliability* yang menggambarkan tentangseberapa jauh sebuah perangkat lunak dapat memberikan toleransi kegagalanatas kesalahan yang terjadi saat digunakan pada kondisi tertentu.

c. *Recoverability*

Sub-faktor ini berhubungan dengan kemampuan sebuah perangkat lunak untuk dapat membangun kembali kinerja dan memulihkan data baik secara langsung maupun secara tidak langsung.

d. *Reliability Compliance*

Reliability compliance berhubungan dengan kesesuaian antara tingkat pelayanan yang diberikan oleh perangkat lunak dengan aturan standar yang digunakan pada setiap perangkat lunak yang memiliki tipe yang sama.

3. *Usability*

Usability berhubungna dengan seberapa baik perangkat lunak dapat dipahami, dipelajari, dan digunakan.

a. *Understandability*

Sub-faktor ini berhubungan dengan seberapa jauh sebuah perangkat lunak dapat dipahami oleh pengguna baik dari secara konsep logis dan penerapan penggunaan perangkat lunak tersebut.

b. *Learnability*

Learnability menggambarkan tentang sebuah perangkat lunak dapat dipelajari dengan baik oleh penggunanya.

c. *Operability*

Operability berhubungan dengan seberapa jauh perangkat lunak dapat dioperasikan oleh penggunanya.

d. *Attractiveness*

Sub-faktor ini menggambarkan tentang bagaimana sebuah perangkat lunak dapat menarik perhatian bagi penggunanya.

e. *Usability Compliance*

Sub-faktor ini berhubungan dengan kesesuaian antara kegunaan perangkat lunak dengan standar yang digunakan oleh perangkat lunak sejenis lainnya.

4. *Efficiency*

Faktor *efficiency* berhubungan dengan efisiensi dari seberapa besar sumber daya yang digunakan oleh sebuah perangkat lunak.

a. *Time Behaviour*

Time behaviour berhubungan dengan waktu yang dibutuhkan oleh sebuah perangkat lunak untuk menjalankan suatu fungsi atau tujuan tertentu.

b. *Resource Utilization*

Sub-faktor ini menggambarkan tentang jumlah sumber daya yang digunakan oleh perangkat lunak untuk melakukan fungsi atau tujuan tertentu.

c. *Efficiency Compliance*

Efficiency compliance berhubungan dengan kesesuaian antara sumber daya yang digunakan oleh sebuah perangkat lunak dengan standar penggunaan sumber daya pada perangkat lunak lainnya yang sejenis.

5. *Maintainability*

Maintainability menggambarkan tentang pemeliharaan sebuah perangkat lunak, seberapa baik perangkat lunak tersebut dapat dipertahankan.

a. *Analyzability*

Analyzability berhubungan dengan seberapa jauh sebuah perangkat lunak dapat di analisis, hal ini diperlukan untuk analisis kekurangan atau penyebab kegagalan agar dapat diketahui bagian mana yang perlu dimodifikasi.

b. *Changeability*

Changeability berhubungan dengan seberapa baik perangkat lunak dapat diubah, upaya ini diperlukan untuk modifikasi, penghapusan kesalahan atau perubahan lingkungan.

c. *Stability*

Sub-faktor ini berhubungan dengan stabilitas dari sebuah perangkat lunak yang memungkinkan untuk menyimpulkan resiko efek tak terduga yang disebabkan oleh modifikasi.

d. *Testability*

Testability menggambarkan tentang bagaimana perangkat lunak dapat diuji, hal ini untuk menyimpulkan tentang upaya yang diperlukan untuk memvalidasi perangkat lunak dan cakupan pengujian.

e. *Maintainability Compliance*

Hal ini berhubungan dengan kesesuaian antara pemeliharaan yang dilakukan terhadap perangkat lunak dengan standarisasi yang terdapat pada pemeliharaan perangkat lunak lainnya yang sejenis.

6. *Portability*

Portability menggambarkan tentang kemampuan sebuah perangkat lunak untuk dapat berpindah dari sebuah lingkungan atau sistem ke sistem lainnya.

a. *Adaptability*

Sub-faktor ini berhubungan dengan seberapa jauh sebuah perangkat lunak dapat beradaptasi dengan perubahan lingkungan atau sistem yang berbeda.

b. *Installability*

Installability menggambarkan tentang seberapa baik perangkat lunak dapat digunakan dalam lingkungan atau sistem tertentu.

c. *Co-existence*

Co-existence berhubungan dengan bagaimana perangkat lunak dapat berdampingan dengan produk atau perangkat lunak lain pada suatu lingkungan atau sistem yang sama untuk mengetahui tentang dependensi, perilaku, atau efek samping yang ditimbulkan.

d. *Replaceability*

Replaceability berhubungan dengan bagaimana sebuah perangkat lunak dapat menggantikan perangkat lunak lain apakah ada kebergantungan kepada perangkat lunak lain saat perangkat lunak tersebut digunakan.

e. *Portability Compliance*

Portability compliance berhubungan dengan kesesuaian antara perubahan yang dapat dilakukan oleh sebuah perangkat lunak dengan standarisasi *portability* yang terdapat pada perangkat lunak lain yang sejenis.

Selain faktor internal dan eksternal yang terdapat pada model ISO-9126, terdapat empat faktor *quality in use* pada model ini yang dapat dilihat pada gambar II-1, antara lain:



Gambar II-1 *Quality in Use* pada ISO-9126

1. *Effectiveness*

   Effectiveness berhubungan dengan kemampuan untuk mencapai tujuan pengguna melalui akurasi dan kelengkapan perangkat lunak.
2. *Productivity*

   Productivity merupakan upaya perangkat lunak dalam menghindari kelebihan penggunaan sumber daya untuk mencapai tujuan pengguna.
3. *Safety*

   Safety merupakan kemampuan perangkat lunak untuk dapat mengurangi tingkat kegagalan pada pengguna lain.
4. *Satisfaction*

   Satisfaction merupakan tingkat kepuasan pengguna dalam menggunakan sebuah perangkat lunak.

II.6.1 Metriks ISO 9126-2 (*External*)

Pengukuran kualitas perangkat lunak dibutuhkan untuk membantu dalam menentukan atau membuat karakteristik khusus dari proses dan produk informasi. Pengukuran meliputi evaluasi kuantitatif sehingga suatu subjek dapat diketahui perbedaannya dalam jumlah atau derajat. variabel yang diukur menggunakan skala *interval* dan *ratio* umumnya merupakan variabel metrik serta ukuran dapat menggunakan secara langsung menentukan pencapaian tujuan kualitas numerik.

Pengukuran selalu menjadi kemajuan untuk setiap disiplin pengujian perangkat lunak. pengukuran perangkat lunak dan metrik menjadi peran mendasar dalam siklus hidup pengembangan perangkat lunak, dan metrik perangkat lunak telah digunakan dalam membuat keputusan kuantitatif/kualitatif maupun dalam penilaian risiko dan pengurangan dalam proyek perangkat lunak. Proses pengukuran merupakan cara Untuk melakukan pengukuran faktor kualitas dari sebuah perangkat lunak.

Salah satu tolak ukur kualitas perangkat lunak yang akan digunakan untuk menganalisis perangkat lunak *video editor* adalah metriks ISO 9126, yang dibuat oleh *International Organization for Standardization* (ISO) dan *International Electrotechnical Commission* (IEC). ISO 9126 mendefinisikan kualitas produk perangkat lunak, model, karakteristik mutu, dan metrik terkait yang digunakan untuk mengevaluasi dan menetapkan kualitas sebuah produk *software*. Metriks ISO 9126 itu sendiri meliputi *External Metric, Internal Metric dan Quality in Use Metric,* dan melakukan perhitungan berdasarkan karakteristik yang terdapat pada model ISO 9126.

Perhitungan metrik ISO 9126 itu sendiri meliputi karakteristik dari perangkat lunak yang akan di analisis dan juga 6 faktor dan beberapa sub-faktor yang terdapat pada model ISO 9126. Dan selain itu setiap faktor dan karakteristik perangkat lunak memiliki cara perhitungan tersendiri, karena telah ditentukan oleh metrik ISO 9126.

Tabel II-2 ISO 9126 *Metrics*

ISO 9126 *Metrics*	
Functionality Metric	*Suitability Metric* *Accuracy Metric* *Interoperability Metric* *Security Metric* *Functionality Compliance Metric*
Reability Metric	*Maturity Metric* *Fault Tolerance Metric* *Recoverability Metric* *Reliability Compliance Metric*

ISO 9126 *Metrics*	
Usability Metric	*Understandability Metric* *Learnability Metric* *Operability Metric* *Attractiveness Metric* *Usability Compliance Metric*
Efficiency Metric	*Time Behavior Metric* *Resource Utilization Metric* *Efficiency Compliance Metric*
Maintainability Metric	*Analyzability Metric* *Changeability Metric* *Stability Metric* *Maintainability Compliance Metric*
Portability Metric	*Adaptability Metric* *Instability Metric* *Co-existence Metric* *Replaceability Metric* *Portability Complaince Metric*

1. Functionality Metrics

Metrik fungsionalitas eksternal harus mampu mengukur atribut seperti yang terdapat pada fungsional dari setiap perangkat lunak. Setiap perangkat lunak dapat diamati dari perspektif sebagai berikut:

a. Suitability Metrics

Metrik kesesuaian eksternal harus mampu mengukur atribut seperti terjadinya fungsi yang tidak sesuai atau terjadinya operasi yang tidak sesuai selama pengujian dan user pengoperasian sistem. Dapat dilihat pada Tabel II-3.

Tabel II-3 *Suitability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Functional adequacy	X = 1-A/B A = jumlam fungsi yang terdapat masalah dalam evaluasi B = Jumlah fungsi yang di evaluasi	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Functional implementation completeness	X = 1-A/B A = jumlah fungsi yang hilang terdeteksi dalam evaluasi B = Jumlah fungsi yang dijelaskan dalam spesifikasi kebutuhan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Functional implementation coverage	X = 1-A/B A = jumlah yang dilakasanakan dengan benar atau fungsi yang hilang dalam evaluasi B = jumlah fungsi yang dijelaskan dalam spesifikasi kebutuhan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Functional specification stability (volatility)	X = 1-A/B A = Jumlah fungsi perubahan setelah memasuki operasi dimulai B = Jumlah fungsi yang dijelaskan dalam spesifikasi kebutuhan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

b. Accuracy Metrics

Metrik akurasi eksternal harus mampu mengukur atribut seperti frekuensi pengguna dalam menghadapi terjadinya hal-hal yang tidak akurat. Dapat dilihat pada Tabel II-4.

Tabel II-4 *Accuracy Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Accuracy to expectation	X = A/T A = jumlam kasus yang dihadapi oleh pengguna dengan perbedaan terhadap hasil yang diharapkan T = Waktu operasi	0 <= X Semakin dekat ke 0 semakin baik	X= Count/Time A=Count T= Time
Computational Accuracy	X = A/T A=Jumlah perhitungan akurasi yang dihadapi oleh pengguna T = Waktu operasi	0 <= X Semakin dekat ke 0 semakin baik	X= Count/Time A=Count T= Time
Precision	X = A/T A = Jumlah hasil yang dihadapi oleh pengguna dengan tingkat presisi yang berbeda dari yang dibutuhkan T = Waktu operasi	0 <= X Semakin dekat ke 0 semakin baik	X= Count/Time A=Count T= Time

c. Interoperability Metrics

Metrik interoperabilitas eksternal harus mampu mengukur atribut seperti jumlah fungsi atau kejadian yang kurang baik yang melibatkan data dan perintah, yang dengan mudah antara produk perangkat lunak dan sistem

lainya, produk perangkat lunak lain, atau peralatan yang berhubungan. Dapat dilihat pada Tabel II-5.

Tabel II-5 ***Interoperability Metrics***

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Data exchangeability (Data format based)	X = A/B A = jumlam format yang data yang disetujui untuk ditukar dengan perangkat lunak lain atau sistem pengujian bursa data B = Total jumlah format data yang akan diperlukan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Data exchangeability (User's success attempt based)	X = 1-A/B A = Jumlah kasus dimana pengguna gagal untuk pertukanaran data dengan perangkat lunak atau sistem lain B = Jumlah kasus di mana pengguna upaya untuk pertukaran data	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

d. Security Metrics

Metrik keamanan eksternal harus mampu mengukur atribut seperti jumlah fungsi berhubungan dengan masalah keamanan. Dapat dilihat pada Tabel II-6.

Tabel II-6 *Security Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Access auditability	X = A/B A= Jumlah "pengguna akses ke sistem data" tercatat dalam database sejarah akses B= Jumlah "pengguna akses ke sistem dan data" dilakukan selama evaluasi	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Access controllability	X = A/B A = Jumlah yang terdeteksi berbagai jenis operasi ilegal B = Jumlah jenis operasi ilegal seperti dalam spesifikasi	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Data corruption prevention	X = 1 – A/N A = Jumlah kali bahwa peristiwa korupsi data terbesar yang terjadi N = Jumlah uji coba menyebabkan acara korupsi data	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count N= Count

e. Functionality Compliance Metrics

Sebuah fungsi kepututusan metrik eksternal harus mampu mengukur atribut seperti jumlah fungsi dengan akurat, atau kejadian masalh keputusan, yang

merupakan produk *software* gagal untuk mengetahui standar, konvesi, kontrak atau persyaratan lainya. Dapat dilihat pada Tabel II-7.

Tabel II-7 *Functionality Compliance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Functional compliance	X = 1 - A/B A = Jumlah item kepatuhan fungsi tertentu yang belum dilakasankan selama pengujian B = Total jumlah item kepatuhan fungsi tertentu	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Interface standard compliance	X = A/B A = Jumlah interface diterapkan dengan benar sebagaimana ditentukan B = Total jumlah interface yang membutuhkan pemenuhan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

2. Reliability Metrics

Metrik kehandalan eksternal harus mampu mengukur atribut yang berhubungan dengan perilaku dari sistem dengan perangkat lunak merupakan bagian selama pengujian eksekusi untuk menunjukan sejauh mana keandalan perangkat lunak dalam selama operasi, sistem dan perangkat lunak yang tidak dibedakan dari satu sama lain untuk semua kebanyakan kasus.

a. Maturity Metrics

Matrik ini harusmampu mengukur atribut seperti kebebasan *software*, kegagalan disebabkan oleh kesalahan yang ada dalam perangkat lunak. Dapat dilihat pada Tabel II-8.

Tabel II-8 *Maturity Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Estimated latent fault density	X = {ABS(A1 – A2)}/B (X:perkiraan yang kepadatan kesalahan laten) ABS ()= Absolut A1 = Jumlah kesalahan laten memprediksi dalam produk perangkat lunak A2 = Jumlah sebenarnya terprediksi kesalahan. B = Produk Ukuran	0 <= X Hal ini tergantung pada tahap pengujian. Pada tahap selanjutnya lebih kecil lebih baik.	X= Count/Size A1=Count A2= Count B = Size
Failure density against test cases	X = A1/A2 A1 = Jumlah kegagalan terdeteksi A2 = Jumlah kasus uji yang dilakukan	0 <= X Hal ini tergantung pada tahap pengujian. Pada tahap selanjutnya lebih kecil lebih baik.	X, Y= Count/Size A1= Count A2=Count B= Size
Fault density	X = A/B A = Jumlah kesalahan yang terdeteksi B = Ukuran Produk	0 <= X Hal ini tergantung pada tahap pengujian. Pada tahap selanjutnya lebih kecil lebih baik.	X = Count/Size A = Count B = Size

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Fault removal	a).X = A1/A2 A1= Jumlah kesalahan dikoreksi A2= Jumlah Sebenarnya terdeteksi kesalahan. b).Y = A1/A3 A= Jumlah kesalahan laten memprediksi dalam produk perangkat lunak.	0 <= X <= 1 Semakin dekat 1.0 semakin lebih baik sebagai kesalahan yang lebih sedikit 0 <= Y Semakin dekat 1.0 semakin lebih baik sebagai kesalahan yang lebih sedikit	X= Count/Count Y= Count/Count A1=Count A2=Count A3=Count
Mean time between failures (MTBF)	a).X = T1/A b).Y = T2 / A T1=Waktu operasi T2=Jumlah interval waktu yang berturut-turuk mengalami kegagalan A=Jumlah sebenarnya terdeteksi kegagalan,(kegagalan terjadi selama waktu operasi).	0 <= X, Y Semakin lama semakin baik, seperti waktu yang lebih lama dapat diharapkan anatara kegagalan	X =Time / Count Y =Time /Count A = Count T1 = Time T2 = Time
Test coverage (Specified operation scenario testing coverage)	X = A / B A=Jumlah kasus uji bener-bener dilakukan mewakili skenario operasi selama pengujiaan. B=Junlah kasus uji yang akan dilakukan untuk menutupi kebutuhan	0 <= X <= 1 Semakin mendekati 1.0 semakin lebih baik	X= Count/Count A=Count B= Count
Test maturity	X = A / B A=Jumlah kasus uji lulus selama pengujian atau	0 <= X <= 1 Semakin mendekati 1.0 semakin lebih baik	X= Count/Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	operasi B=Jumlah kasus uji yang akan dilakukan untuk menutupi kebutuhan		A=Count B= Count

b. Fault Tolerance Metrics

Sebuah kesalahan toleransi metrik eksternal harus berhubungan dengan kamampuan *software*, mempertahankan tingkat kinerja yang ditentukan dalam kasus kesalahan operasi atau pelanggaran antarmuka yang ditentukan. Dapat dilihat pada Tabel II-9.

Tabel II-9 *Fault Tolerance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Breakdown avoidance	X = 1-A/B A = Jumlah kerusakan B = Jumlah kegagalan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Failure avoidance	X = A/B A = Jumlah kejadian kegagalan kritis dan serius terhadap kasus uji dari pola kesalahan B=Jumlah kasus uji dieksekusi dari pola kesalahan.	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Incorrect operation avoidance	X = A/B A = Jumlah kejadian kegagalan kritis dan serius terhadap kasus uji dari pola kesalahan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	B=Jumlah kasus uji dieksekusi dari pola kesalahan.		

c. Recoverability Metrics

Metrik pemulihan eksternal harus mampu mengukur atribut seperti perangkat lunak dengan sistem yang mampu membangun kembali tingkat yang memadai atas kinerja dan memulihkan data yang terkena dampak langsung dalam kasus kegagalan. Dapat dilihat pada Tabel II-10.

Tabel II-10 *Recoverability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Availability	a).X={To/(To+Tr)} To=Waktu untuk operasi Tr=Wakto untuk meperbaiki A1=Total kasus yang tersedia pengguna *software* pengguna berhasil ketika pengguna mencoba untuk menggunakan b).Y=A1/A2 A2=Jumlah kasus upaya pengguna untuk menggunakan perangkat lunak selama waktu pengamatan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik, karena pengguna dapat menggunakan perangkat lunak untuk lebih banyak waktu 0 <= Y <= 1 Semakin dekat ke 1.0 semakin baik	X= Time/Time To = Time Tr = Time Y= Count/Count A1= Count A2= Count
Mean down time	X=T/N T= Jumlah down time N=Jumlah kerusakan yang	0 <X Semakin dekat ke 1.0 semakin baik	X= Size/Count T=Time

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	diamati kasus terburuk atau distribusi down time harus diukur.		N=Count
Mean recovery time	X=Sum(T)/N T=Waktu untuk pemulihan sistem perangkat lunak pada setiap kesempatan B=Jumlah kasus yang diamati sistem *software* masuk recovery	0 <X Semakin dekat ke 1.0 semakin baik	X= Time/Count T=Time B= Count
Restartability	X=A/B A=Jumlah restart yang bertemu untuk waktu yang dibutuhkan selama pengujian atau operasi pengguna B=Total jumlah restart selama pengujian atau operasi pengguna	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik, karena pengguna dapat me restart dengn mudah.	X=Count/Count A=Count B=Count
Restorability	X=A/B A=Jumlah kasus restorasi berhasil dilakukan B=Jumlah kasus restorasi diuji sesuai kebutuhan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik, karena produk lebih mampu untuk mengembangkan dalam kasus tertentu.	X=Count/Count A=Count B=Count
Restore effectiveness	X=A/B A=Jumlah kasus berhasil dipublikasikan memenuhi terget mengembalikan waktu B=Jumlah kasus yang	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik, karena proses restorasi di produk lebih efektif.	X=Count/Count A=Count B=Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	dilakukan		

d. Reliability Compliance Metrics

Sebuah kehandalan kepatuhan metrik eksternal harus mampu mengukur atribut seperti jumlah fungsi dengan, atau kejadian masalah keputusan, di mana produk perangkat lunak gagal untuk mematuhi standar, konvensi atau peraturan yang berkaitan dengan kehandalan suatu perangkat lunak. Dapat dilihat pada Tabel II-11.

Tabel II-11 *Reliability Compliance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Reliability compliance	X = 1-A/B A = Jumlah item kepuasan keandalan ditentukan yang belum dilaksanakan selama pengujian B = Total jumlah item kepatuhan keandalan ditentukan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

3. Usability Metrics

Metrik ini mengukur sejauh mana perangkat lunak dapat dipahami, dipelajari,dioperasikan, menarik dan sesuai dengan peraturan kegunaan dan pedoman.

Banyak metrik *usability* eksternal diuji oleh pengguan mencoba untuk menggunakan fungsi yang terdapat pada perangkat lunak. Hasil akan dipengaruhi oleh kemampuan pengguna dan karakteristik perangkat lunak. Kareana *software* dievaluasi dijalankan dalam kondisi eksplisit ditentukan oleh sampel pengguna yang mewakili kolompok pengguna yang di diindentifikasikan. Untuk hasil yang bisa diandalkan sampel pengguna setidaknya harus memnguasai perangkat lunak tersebut,

meskipun informasi yang berguna dapat diperoleh dari kelompok-kelompok yang lebih kecil. Pengguna harus melakukan tes tanpa petunjuk atau bantuan dari luar.

a. Understandability Metrics

Pengguna harus dapat memilih produk perangkat lunak, yang cocok untuk digunakan. Sebuah metrik harus dapat menilai apakah penguna baru dapat memahami, apakah *software* tersebut cocok, dan bagaimana hal itu dapat digunakan untuk tugas-tugas tertentu. Dapat dilihat pada Tabel II-12.

Tabel II-12 *Understandability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Completeness of description	X = A/B A =Jumlah fungsi (jenis fungsi) dipahami B = Total jumlah (jenis fungsi)	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count
Demonstration accessibility	X = A/B A=Jumlah demonstrasi/tutorial yaang pengguan berhasil mengaksesnya B=Jumlah demonstrasi/tutorial yang tersedia	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count
Demonstration accessibility in use	X = A/B A =Jumlah kasus di mana pengguna berhasil melihat demonstrasi ketika pengguna mencoba untuk meliha demonstrasi B =Jumlah kasus di mana pengguna mencoba untuk melihat demonstrasi	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	selama periode pengamatan		
Demonstration effectiveness	X = A/B A =Jumlah fungsi yang di operasikan berhasil B=Jumlah demonstrasi/tutorial yang di akses	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count
Evident functions	X = A/B A =Jumlah fungsi (jenis fungsi) yang didentifikasi oleh pengguna B =Total jumlah fungsi yang sebenarnya	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count
Function understand-ability	X = A/B A =Jumlah fungsi antarmuka yang tujuanya dengan benar dijalankan oleh pengguna B =Jumlah fungsi yang tersedia dari antarmuka	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count
Understandable input and output	X = A/B A =Jumlah input dan output data item yang pengguna berhasil mengerti B =Jumlah item data input dan output yang tersedia dari antarmuka.	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.	X= Count/Count A=Count B= Count

b. Learnability Metrics

Sebuah *learnability* metrik eksternal harus dapat menilai seberapa lama pengguna mengambil untuk belajar sebagai menggunakan fungsi tertentu, dan efektivitas sistem bantuan dan dokumentasi. Dapat dilihat pada Tabel II-13.

Tabel II-13 *Learnability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Ease of function learning	T= Berarti waktu yang dibutuhkan untuk pembelajaran menggunakan fungsi dengan benar	0 < T Semakin pendek semakin baik	T = Time
Ease of learning to perform a task in use	T=Jumlah waktu operasi pengguna sampau waktu dicapai untuk melakukan tugas tertentu dalam waktu singkat	0 < T Semakin pendek semakin a=baik	T = Time
Effectiveness of the user documentation and/or help system	X = A/B A = Jumlah tugas yang berhasil diselesaikan setelah mengakses bantuan online atau dokumentasi B =Jumlah tugas diuji	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

c. Operability Metrics

Sebuah *operability* metrik eksternal harus dapat menilai apakah pengguna dapat mengoprasikan dan mengendalikan perangkat lunak. Metrik pengoperasian dapat dikategorikan oleh prinsip-prinsip dialog dalam ISO-92126-10. Dapat dilihat pada Tabel II-14.

Tabel II-14 *Operability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Operational consistency in use	a).X = 1-A/B A = Jumlah pesan atau fngsi uang menemukan data yang tidak konsisten dengan akspektasi pengguna B = Jumlah pesan atau fungsi b).Y=N/UOT N= Jumlah pesan atau fngsi uang menemukan data yang tidak konsisten dengan akspektasi pengguna yang UOT=Pengguna waktu operasi selama sistem berjalan.	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik 0 <= Y Semakin dekat ke 0.0 semakin baik	X= Count/Count A=Count B= Count UOT= Count/Count N=Count Y= Count

d. Attractiveness Metrics

Eksternal metrik *attractiveness* metrik harus bisa menilai penamplan perangkat lunak. Dan akan dipengaruhi oleh fakto-faktor seperti desain layar dan warna. Hal ini sangat penting untuk produk konsumen. Dapat dilihat pada Tabel II-15.

Tabel II-15 *Attractivanes Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Attractive interaction	Kuesioner untuk menilai daya tarik dari antarmuka pengguan, setelah pengguna mengunakan.	Tergantung pada metode kuesioner scoringnya	Count
Interface appearance customisability	X = A/B A = Jumlah elemen antarmuka disesuaikan dalam penampilan untuk kepuasan pengguna B = Jumlah antarmuka pengguna yang ingin menyesuaikan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

e. Usability Compliance Metrics

Sebuash *Usability compliance* metrik eksternal harus mampu menilai kepatuhan terhadap standar, konvensi, panduan gaya atau peraturan yang berkaitan dengan kegunaan. Dapat dilihat pada Tabel II-16.

Tabel II-16 *Usability Compliance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Usability compliance	X = 1-A/B A = Jumlah item kepuasan kegunan terentu yang belum dilaksanakan selama pengujian B = Total jumlah item kepuasan kegunaan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	tertentu		

4. Efficiency Metrics

Efficiency metrik eksternal harus mampu mengukur atribut seperti konsumsi waktu dan sumber daya perilaku pemanfaatan sistem komputer termasuk perangkat lunak selama pengujian operasi.

Disarankan bahwa maksimal dan distribusi waktu yang diuji untuk banyak kasus pengujian atau operasi, karena ukuran yang dipengaruhi kuat dan tergantung pada kondisi penggunaan, seperti beban pengolahan data, frekuensi penggunaan, jumlah yang mengandung situs dan sebagainya. Oleh karena itu, metrik efisiensi dapat mencakup rasio nilai aktual yang di ukur dengan fluktual kesalahan dengan nilai yang dirancang dengan memungkinkan berbagai kesalahan, yang dubutuhkan oleh spesifikasi.

a Time Behaviour Metrics

Time behaviour metrik eksternal harus mampu mengukur atribut seperti perilaku saat sistem komputer termasuk perangkat lunak selama pengujian atau pengoperasian. Dapat dilihat pada Tabel II-17.

Tabel II-17 *Time Behaviour Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Response time	T = (Waktu untuk mendapatkan hasil)-(saat masuk perintah selesai)	0 < T Semakin cepat semakin baik.	T = Time
Response time (Mean time to response)	Tmean = (Ti) ∑/ N, (untuk i = 1 sampai N) TXmean = diperlukan waktu respon rata-rata	0 <= X Semakin dekat ke 1.0 dan kurang dari 1.0 lebih baik	Tmean= Time TXmean= Time Ti= Time N= Count X= Time/ Time

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	Ti = waktu respon untuk evaluasi ke-i (shot) N = jumlah evaluasi (tembakan sampel)		
Response time (Worst case response time ratio)	X = Tmax / Rmax Tmax = MAX (Ti) (untuk i = 1 sampai N) Rmax = diperlukan waktu respons maksimum MAX (Ti) = waktu respons maksimum antara evaluasi N = jumlah evaluasi (tembakan sampel) Ti = waktu respon untuk evaluasi ke-i (shot)	0 <= X Semakin ke 1 dan kurang dari 1 akan lebih baik	Tmax= Time Rmax=Time Ti= Time N= Count X= Time/ Time

b. Resource Utilisation Metrics

Resource Utilisation metrik eksternal harus mampu mengukur atribut seperti sumber daya sistem perangkat lunak selama pengujian atau pengoperasian. Dapat dilihat pada Tabel II-18.

Tabel II-18 *Resource Utilisation Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
I/O devices utilisation	X = A/B A = Waktu I/O perangkat diduduki B = Ditentukan waktu yang dirancang untuk menempati	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	A= Time B= Time X= Time/Time
I/O loading limits	X = Amax/Rmax Amax = MAX (Ai), (untuk i = 1 sampai N)	0 <= X Semakin kecil semakin baik	Amax = Count Rmax =

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	Rmax = maksimum yang diperlukan I / O pesan MAX (Ai) = Jumlah maksimum pesan I / O dari 1 untuk evaluasi ke-i. N = jumlah evaluasi.		Count Ai = Count N= Count X = Count/ Count
I/O related errors	X = A/T A= Jumlah pesan peringatan atau kegagalan sistem T= Penggunaan waktu operasi selam pengamatan pengguna	0 <= X Semakin kecil semakin baik	A = Count T = Time X = Count/ Time
Mean I/O fulfillment ratio	X = Amean/Rmean Amean = ∑ (Ai) / N Rmean = diperlukan rata jumlah pesan I /O Ai = jumlah pesan kesalahan I / O untuk i th evaluasi N = jumlah evaluasi	0 <= X Semakin kecil semakin baik	Amean = Count Rmean = Count Ai = Count N= Count X = Count/ Count
User waiting time of I/O devices utilisation	T= Waktu yang dihabiskan untu menugguakhir dari perangkat I/O operasi	0 < T Semakin kecil semakin baik	T = Time

c. Efficiency Compliance Metrics

Efficiency Compliance metrik eksternal harus mampu mengukur atribut seperti jumlah fungsi, atau kejadian masalah kepatuhan, yang merupakan produk *software* gagal untuk mematuhi standar, konvensi atau peraturan yang berkaitan dengan efisiensi. Dapat dilihat pada Tabel II-19.

Tabel II-19 *Efficiency Compliance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Efficiency Compliance	X = 1-A/B (X:Rasio item kepatuhan berkaitan denga efisiensi) A = Jumlah efisiensi item kepatuahan ditentukan yang belum dilaksanakan selama pengujian B = Total jumlah item kepatuhan efisiensi dintentukan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

5. Maintainability Metrics

Metrik *maintainability* eksternal harus mampu mengukur atribut seperti pengolahan atau perawatan perangkat lunak, pengguna atau sistem termasuk perangkat lunak, ketika perangkat lunak dipertahankan atau diubah selama pengujian atau pemeliharaan.

a. Analysability Metrics

Sebuah *analisability* metrik eksternal harus mampu mengukur atribut seperti pemeliharaan atau usaha atau menghabisan sumber daya ketika mencoba untuk mengetahui kekurangan atau penyebab kegagalan atau bagian untuk pengguna. Dapat dilihat pada Tabel II-20.

Tabel II-20 *Analysability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Audit trail capability	X = A/B A = Jumlah data yang sebenarnya direkam selama operasi B = Jumlah data direncanakan akan cukup direkam untuk memantau status perangkat lunak selama operasi	0 <= X Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Diagnostic function support	X = 1-A/B A = Jumlah kegagalan yang pemelihara dapat diagnos e (menggunakan fungsi diagnostik) untuk memahami penyebab - kapal hubungan efek B = Total jumlah kegagalan terdaftar	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

b. Changeability Metrics

Metrik *changeability* aksternal harus mampu mengukur atribut seperti upaya pemelihara atau pengguna dengan mengukur perilaku pengelola, pengguna atau sistem termasuk perangkat lunak ketika mencoba untuk menerapkan modifikasi tertentu. Dapat dilihat pada Tabel II-21.

Tabel II-21 *Changeability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Change cycle efficiency	Average Time : Tav = Sum(Tu) / N	0<Tav Semakin pendek lebih	Tu= Time Trc,

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	Tu= Trc – Tsn Tsn = Waktu di mana pengguna selesai untuk mengirim permintaan untuk pemeliharaan untuk pemasok dengan laporan masalah Trc = Waktu di mana pengguna menerima rilis versi revisi (atau laporan status) N = Jumlah versi revisi	baik kecuali dari jumlah versi revisi itu besar	Tsn = Time N= Count Tav= Time
Change implementation elapse time	Average Time : Tav = Sum(Tm) / N Tm=Tout – Tin Tout = Waktu di mana penyebab kegagalan dihapus dengan mengubah perangkat lunak (atau status dilaporkan kembali ke pengguna) Tin = Waktu di mana penyebab kegagalan yang menemukan N = Jumlah kegagalan terdaftar dan dihapus	0<Tav Semakin pendek lebih baik kecuali dari jumlah versi revisi itu besar	Tm=Time Tin, Tout = Time Tav= Time
Modification complexity	T = Sum (A / B) / N A = Kerja dihabiskan untuk mengubah B= Ukuran perubahan *software*	0 < T Semakin dekat ke 1.0 semakin baik	A= Time B=Size N= Count T= Time

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	N = Jumlah perubahan		
Parameterised modifiability	X=1- A / B A= Jumlah kasus yang pemelihara gagal untuk mengubah perangkat lunak dengan menggunakan parameter B = Jumlah kasus yang maintainer mencoba untuk mengubah perangkat lunak dengan menggunakan parameter	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	A=Count B= Count X= Count/ Count
Software change control capability	X= A / B A = Jumlah data log perubahan benar-benar dicatat B=Jumlah data perubahan log rencananya akan direkam cukup untuk melacak perubahan perangkat lunak	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	A=Count B= Count X= Count/ Count

c. Stability Metrics

Sebuah *Stability Metrics* eksternal harus mampu mengukur atribut yang berhubungan dengan perilaku yang tak terdiga dari sistem perangkat lunak ketika perangkat lunak diuji atau dioperasikan. Dapat dilihat pada Tabel II-22.

Tabel II-22 *Stability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Change success ratio	X = Na/Ta Y={(Na/Ta)/(Nb/Nb)} Na = Jumlah kasus yang pengguna pertemuan kegagalan selama operasi setelah *software* berubah Nb = Jumlah kasus yang pengguna pertemuan kegagalan selama operasi sebelum perangkat lunak berubah Waktu ta = Operasi selama periode observasi ditentukan setelah perangkat lunak berubah Tb waktu = Operasi selama periode pengamatan ditentukan sebelum perangkat lunak berubah	0 <= X,Y Semakin dekat ke 1.0 semakin baik	Na, Nb= Count Ta,Tb=Time X= Count/Time Y=[(Count/Time) / (Count/Time)]
Modification impact localisation (Emerging failure after change)	X=A/N A = Jumlah kegagalan muncul setelah kegagalan diselesaikan oleh perubahan selama periode tertentu N = Jumlah kegagalan diselesaikan	0<=X Semakin dekat ke 0 semakin baik	X= Count/Court A=Count N= Count

d. Testability Metrics

Sebuah *Testability Metrics* eksternal harus mampu mengukur atribut seperti upaya pemelihara atau pengguna dengan mengukur perilaku pengelola, pengguna atau sistem termasuk perangkat lunak ketika mencoba untuk diuji. Dapat dilihat pada Tabel II-23.

Tabel II-23 *Testability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Availability of built-in test function	X = A/B A=Jumlah kasus dimana pengelola dapat mengunakan sesuia buit-in fungsi tes B=Jumlah kasus peluang uji	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Re-test efficiency	X = Sum(T)/N T=Waktu yang dihabiskan untuk menguji untuk memastikan apakah melaporkan kegagalan itu diselesaikan atau tidak N = Jumlah kegagalan diselesaikan	0 <= X Semakin dekat ke 1.0 semakin baik	X= Time/Count T=Time N= Count
Test restartability	X = A/B A = Jumlah kasus di mana pengelola dapat menghentikan sebentar dan restart melaksanakan uji coba pada titik-titik yang diinginkan untuk memeriksa langkah demi langkah	0 <= X Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	B = Jumlah kasus jeda melaksanakan uji coba		

e. Maintainability Compliance Metrics

Sebuah *Maintainability Compliance Metrics* eksternal harus mampu mengukur atribut seperti fungsi atau kejadian masalah kepuasan, dimana produk peranglat lunak gagal untuk memenuhi standar yang diperlukan, konvensi atau peraturan yang berkaitan dengan pemeliharaan. Dapat dilihat pada Tabel II-24.

Tabel II-24 *Maintainability Compliance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Maintainability compliance	X = 1-A/B A=Jumlah item kepatuhan rawatan ditentukan yang belum dilaksanakan selama pengujian B = Total jumlah item kepatuhan rawatan ditentukan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

6. Portability Metrics

Portability Metrics ekternal harus mampu mengukru atribut seperti operasi atau sistem selama pengoperasian berlangsung.

a. Adaptability Metrics

Adaptability Metrics harus mampu mengukur atribut seperti perilaku sistem atau pengguna yang mencoba untuk beradaptasi pada perangkat lunak untuk

lingkungan tertentu yang berbeda, ketika pengguna harus menrapkan prosedur dari sebelumnya yang disediakan oleh perangkat lunak. Dapat dilihat pada Tabel II-25.

Tabel II-25 ***Adaptability Metrics***

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Adaptability of data structures	X = A/B A = Jumlah data yang beroperasi dan tetapi tidak diamati karena operasi tidak lengkap disebabkan oleh keterbatasan adaptasi B = Jumlah data yang diharapkan dapat beroperasi dalam lingkungan yang perangkat lunak disesuaikan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Hardware environmental adaptability	X = 1-A/B A=Jumlah fungsi ional operas yang tugas tidak selesai atau tidak cukup menghasilkan untuk memenuhi memadai tingkat s selama co mbined pengujian operasi dengan hardware lingkungan B=Total jumlah fungsi yang diuji	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

b. Installability Metrics

Installability Metrics eksternal harus mampu mengukur atribut seperti perilaku sistem atau pengguna yang mencoba untuk mengintal perangkat lunak dalam lingkungann tertentu. Dapat dilihat pada Tabel II-26.

Tabel II-26 *Installability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Ease of installation	X = A/B A= Jumlah kasus yang pengguna berhasil di chang ing menginstal operasi untuk / kenyamanan nya B= Total jumlah kasus yang pengguna mencoba untuk mengubah menginstal operasi untuk / kenyamanan nya	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
Ease of Setup Re-try	X = 1-A/B A = Jumlah kasus di mana pengguna gagal di coba selama set - up operasi B = Total jumlah kasus di mana pengguna upaya untuk kembali mencoba setup selama set - up operasi	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

c. Co-existence Metrics

Co-existence Metrics eksternal harus mampu mengukur atribut seperti perilaku sistem atau pengguna yang mencoba untuk menggunakan perangkat lunak dengan perangkat lunak independen lain dalam lingkungan yang sama berbagi sumber daya umum. Dapat dilihat pada Tabel II-27.

Tabel II-27 *Co-existence Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Available co-existence	X = A/T A=Jumlah hambatan atau kegagalan tak terduga yang pengguna pertemuan selama operasi bersamaan dengan *software* lain T=Waktu durasi bersamaan operasi perangkat lunak lain	0 <= X Semakin dekat ke 1.0 semakin baik	X=Count/Time A=Count T= Time

d. Replaceability Metrics

Replaceability Metrics eksternal harus mampu mengukur atribut seperti perilaku sistem atau pengguna yang mencoba untuk menggunakan perangkat lunak di tempat *software* ditentukan lain dalam lingkungan perangkat lunak. Dapat dilihat pada Tabel II-28.

Tabel II-28 *Replaceability Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Continued use of data	X= A/B A= jumlah data yang digunakan dalam perangkat lunak lain yang akan diganti dan menegaskan bahwa mereka mampu terus menerus digunakan B= jumlah data yang digunakan dalam perangkat lunak lain untuk diganti dan rencananya akan terus digunakan kembali	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Function inclusiveness	X = A/B A= jumlah fungsi yang menghasilkan hasil yang sama seperti sebelumnya diproduksi dan di mana perubahan tidak akan en diperlukan B= jumlah fungsi diuji yang mirip dengan fungsi yang disediakan oleh perangkat lunak lain yang akan diganti	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count B= Count
User support functional consistency	X = 1-A1/A2 A=Jumlah fungsi baru yang ditemukan pengguna unacceptabl y konsisten dengan ekspektasi pengguna B=Jumlah fungsi baru	0 <= X Semakin dekat ke 1.0 semakin baik	X= Count/Count A1=Count A2= Count

e. Portability Compliance Metrics

Portability Compliance Metrics eksternal harus mampu mengukur atribut seperti jumlah fungsi dengan, atau kejadian masalah kepatuhan, di mana produk perangkat lunak gagal untuk mematuhi standar yang diperlukan, konvensi atau peraturan yang berkaitan dengan portabilitas. Dapat dilihat pada Tabel II-29.

Tabel II-29 *Portability Compliance Metrics*

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
Portability compliance	X = 1-A/B A=Jumlah portabilitas item kepatuhan ditentukan yang belum dilaksanakan selama pengujian	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik	X= Count/Count A=Count

Nama Metrik	Pengukuran	Interaksi Nilai Yang Diukur	Jenis Ukuran
	B=Total jumlah portabilitas item kepatuhan ditentukan		B= Count

II.6.2 Metrik ISO 9126-4 (*Quality in Use*)

Ukuran kualitas yang digunakan harus didasarkan pada data yang mencerminkan hasil pengguna berinteraksi dengan suatu perangkat lunak. Hal ini memungkinkan untuk mengumpulkan data tujuan tertentu. Selain itu terdapat pula model *quality in use* pada model ISO-9126 yang akan digunakan dalam penelitian ini yang merupakan salah satu model kualitas berupa pandangan pengguna terhadap sebuah perangkat lunak. Berikut beberapa faktor yang terdapat dalam model quality in use diantaranya sebagai berikut :

a. Effectiveness

Menilai apakah tugas yang dilakukan oleh pengguna mencapai tujuan tertentu dengan akurasi dan kelengkapan dalam konteks tertentu yang digunakan.

Tabel II-30 *Effectiveness*

Nama Metrik	Pengukuran	Interpretasi nilai ukur
Task Effectiveness	$M1 = \| 1-\Sigma Ai \|_1$ Ai = nilai value proporsional masing-masing komponen yang hilang atau tidak benar dalam output tugas	0 <= M1 <= 1 Semakin dekat ke 1 lebih baik.
Task Completion	X = A / B A = jumlah tugas diselesaikan B = jumlah tugas berusaha	0 <= X <= 1 Semakin dekat ke 1 lebih baik.
Error Frequency	X = A / T A = jumlah kesalahan yang dibuat oleh pengguna T = waktu atau jumlah tugas	0 <= X Semakin dekat ke 0 yang lebih baik.

b. Productivity

Menilai sumber daya yang pengguna gunakan dalam konteks tertentu yang digunakan. Sumber daya yang paling umum adalah waktu untuk

menyelesaikan tugas, meskipun sumber daya lainyang relevan dapat mencakup upaya pengguna.

Tabel II-30 *Productivity*

Nama Metrik	Pengukuran	Interpretasi nilai ukur
Task Time	X=Ta Ta = tugas waktu	0 <= X Semakin kecil semakin baik.
Task Efficiency	X = M1 / T M1 = efektivitas tugas T = tugas waktu	0 <= X Semakin besar lebih baik.
Economic Productivity	X = M1 / C M1 = efektivitas tugas C = total biaya tugas	0 <= X Semakin besar lebih baik.
Productive Proportion	X = Ta / Tb Ta = waktu produktif = waktu tugas - membantu waktu - waktu error - waktu pencarian Tb = waktu tugas	0 <= X <= 1 Semakin dekat ke 1,0 lebih baik.
Relative User Efficiency	Efisiensi pengguna relatif X = A / B A= Efisiensi tugas atau Seorang pengguna biasa B = Efisiensi tugas atau ahli pengguna	0 <= X <= 1 Semakin dekat ke 1,0 lebih baik.

c. Safety

Menilai tingkat resiko terhadap pengguna, bisnis, *software* atau lingkungan dalam konteks tertentu yang digunakan. Ini termasuk kesehatan dan keselamatan baik pengguna dan mereka yang terkena dampak penggunaan serta konsekuensi fisik atau ekonomi yang tidak di inginkan.

Tabel II-32 *Safety*

Nama Metrik	Pengukuran	Interpretasi nilai ukur
User health and safety	X = 1-A / B A = jumlah pengguna melaporkan RSI (cedera yang diakibatkan dari kegiatan yang dilakukan berulang-ulang) B = Jumlah pengguna	0 <= X <= 1 Semakin dekat ke 1 yang lebih baik.
Safety of people affected by use of the	X = 1-A / B	0 <= X <= 1 Semakin dekat ke 1

Nama Metrik	Pengukuran	Interpretasi nilai ukur
system	A = jumlah orang menaruh di hazard B = Jumlah orang yang berpotensi terkena dampak oleh sistem	yang lebih baik.
Economic damage	X = 1-A / B A = jumlah kejadian kerusakan ekonomi B = Jumlah situasi penggunaan	0 <= X <= 1 Semakin dekat ke 1 yang lebih baik.
Software Damage	X = 1-A / B A = jumlah kejadian data hilang dalam perangkat lunak B = Jumlah situasi penggunaan	0 <= X <= 1 Semakin dekat ke 1 yang lebih baik.

d. Satisfaction

Menilai sikap pengguna terhadap penggunaan produk dalam konteks yang digunakan. Sebagai catatan kepuasan dipengaruhi oleh persepsi pengguna sifat dari produk perangkat lunak dan oleh persepsi pengguna dari efisiensi, produktivitas dan dalam penggunaan.

Tabel II-33 *Satisfaction*

Nama Metrik	Pengukuran	Interpretasi nilai ukur
Satisfaction Scale	X = A / B A = memproduksi kuesioner skala psikometrik B = rata populasi	0 < X yang lebih besar lebih baik
Satisfaction Questionnaire	X = Σ (Ai) / n Ai) = menanggapi pertanyaan n = jumlah tanggapan	Bandingkan dengan nilai sebelumnya, atau dengan rata-rata populasi
Discretionary Usage	X = A / B A = berapa kali fungsi perangkat lunak tertentu / aplikasi / sistem yang digunakan B = jumlah kali mereka dimaksudkan untuk digunakan	0 <= X <= 1 dekat ke 1 yang lebih baik.

II.7 Metode Pengolahan Kuesioner

Kuesioner adalah teknik pengumpulan data yang dilakukan dengan cara memberikan seperangkat pertanyaan atau pernyataan kepada orang lain yang dijadikan responden untuk dijawabnya. Pengumpulan data dilakukan untuk memperoleh informasi yang dibutuhkan dalam rangka mencapai tujuan penelitian.

Tujuan yang diungkapkan dalam bentuk hipotesis merupakan jawaban sementara terhadap petanyaan penelitian. Jawaban itu masih perlu diuji secara empiris, dan untuk maksud inilah dibutuhkan pengumpulan data. Data yang dikumpulkan ditentukan oleh variabel-variabel yang ada dalam hipotesis. Data itu dikumpulkan oleh sampel yang telah ditentukan sebelumnya. Sampel tersebut terdiri atas sekumpulan unit analisis sebagai sasaran penelitian.

Oleh karena itu diperlukan metode pengolahan kuesioner. Metode yang digunakan adalah metode dari ISO 9126. Dimana akan melibatkan beberapa metriks ISO 9126.

II.8 Rata-rata Hitung

Rata-rata hitung adalah statistik lokasi yang paling umum dipakai dalam mencari ukuran pemusatan, karena memiliki bias yang kecil terhadap harga sebenarnya. Rata-rata hitung diperoleh dengan menjumlahkan semua pengamatan dalam penelitian dan membaginya dengan banyak pengamatan tersebut.

Bila kita menggunakan lambang statistika, maka rata-rata hitung dapat dirumuskan sebagai berikut:

$$x = \frac{\sum_{i=1}^{n} x_i}{\mathrm{n}} \dots\dots\dots (1.1)$$

Keterangan :

$\sum x_1$ = Nilai

n = Jumlah Nilai

Rumus ini memberi tahu kita untuk menjumlahkan semua pengamatan (n) dan bagilag jumlah tersebut dengan n[16].

Dilambangkan dengan μ untuk populasi dan x untuk contoh. Didefinisikan sebagai berikut:

$$x = \frac{x_1 + x_2 + \cdots + x_n}{n} \text{ atau } x = \frac{\sum_{i=1}^{n} x_i}{n} \ldots\ldots\ldots (1.2)$$

II.9 Metode Pembobotan

Rank Order Centroid (ROC) didasarkan pada tingkat kepentingan atau prioritas dari kriteria. Menurut Jeffreys dan Cockfield dalam Afiefah Rahma (2013), teknik ROC memberikan bobot pada setiap kriteria sesuai dengan ranking yang dinilai berdasarkan tingkat prioritas. Biasanya dibentuk dengan pernyataan "Kriteria 1 lebih penting dari kriteria 2, yang lebih penting dari kriteria 3" dan seterusnya hingga kriteria n, ditulis. Untuk menentukan bobotnya, diberikan aturan yang sama yaitu dimana merupakan bobot untuk kriteria [17] Atau dapat dijelaskan sebagai berikut:

Jika

$$C_{r1} \geq C_{r2} \geq C_{r3} \geq \ldots \geq C_{rn}$$

Maka

$$W_1 \geq W_2 \geq W_3 \geq \ldots \geq W_n$$

Selanjutnya, jika k merupakan banyaknya kriteria, maka

$$W1 = \frac{1 + \frac{1}{2} + \frac{1}{3} + \cdots + 1/k}{k}$$

$$W2 = \frac{0 + \frac{1}{2} + \frac{1}{3} + \cdots + 1/k}{k}$$

$$W3 = \frac{0 + 0 + \frac{1}{3} + \cdots + 1/k}{k}$$

$$Wk = \frac{0 + \cdots + 0 + 1/k}{k} \qquad (2.1)$$

Secara umum pembobotan ROC dapat dirumuskan sebagai berikut:

$$\mathrm{Wk} = \frac{1}{k}\sum_{i=1}^{k}\left(\frac{1}{\mathrm{i}}\right) \ldots\ldots\ldots (2.2)$$

Dimana : k = Banyaknya komponen

i = Bilangan bulat 0 ke -k

Wk = Bobot untuk komponen ke-i

II.10 Metode Perhitungan Kualitas Perangkat Lunak

Weighted Summation (WSum) adalah metode kompensasi, yang berarti dapat menghitung skor kriteria yang "buruk", sehingga dapat dikompensasikan dengan kriteria skor yang "baik". Seperti yang dikemukakan oleh Keeney dan Raiffa (1976), metode ini merupakan bentuk khusus dari *Multi Attribut Value Theory* (MAVT) dan juga disebut model aditif linier. Metode *weighted summation* hanya dapat dilakukan jika informasi mengenai prioritas kriteria sudah diketahui. Perhitungan untuk menilai kualitas perangkat lunak yang berasal dari metrik akan dilakukan menggunakan metode ini, metode ini digunakan pada setiap level perhitungan dalam perhitungan nilai kualitas. Perhitungan dilakukan dengan mengubah atribut yang tak bisa dihitung menjadi dapat dihitung setelah menetapkan bobot tertentu pada attribut tersebut, lalu dilakukan perkalian antara bobot dan nilai kualitas[18] Berikut adalah rumus yang digunakan dalam metode *Weighted Summation,* sebagai berikut:

Jika

$$\mathrm{score}\,(\mathrm{c_i}) = \sum_{i=1}^{N} W_i . V_i\ (c_i) \ldots\ldots\ldots (3.1)$$

Maka

$$\mathrm{score}\,(\mathrm{c_i}) = \sum_{i=1}^{N} W_i . X_i\ (c_i)$$

$$\mathrm{score}\,(\mathrm{c_i}) = W_i\ .X_i + W_2\ .X_2 + \cdots + W_n\ .X_n \ldots\ldots\ldots (3.2)$$

Dimana :

$\mathrm{score}\,(\mathrm{c_i})$ = Nilai kualitas kriteria $\mathrm{c_i}$ $(0 \leq \mathrm{score}\,(\mathrm{c_i}) \leq 1\ )$

W_i = Bobot untuk kriteria c_i

X_i = Nilai perhitungan metrik kriteria ke-*i*

N = Banyaknya Kriteria