BAB II
LANDASAN TEORI

II.1 Definisi Perangkat Lunak

Menurut definisi IEEE *Institute of Electrical and Electronics Engineers*, perangkat lunak adalah program komputer, prosedur, dan dokumentasi yang menyertainya serta data yang digunakan untuk mengoperasikan sistem komputer. Definisi IEEE tersebut hampir identik dengan definisi ISO [**1**] .perangkat lunak terdiri dari 4 komponen, yaitu :

1. Program komputer
2. Prosedur
3. Dokumentasi
4. Data yang diperlukan agar perangkat lunak dapat beroperasi.

II.2 Definisi Kualitas Perangkat Lunak

Menurut definisi dalam Steve McConnell's Code Complete membagi perangkat lunak ke dalam dua hal yaitu *internal* dan *external quality characteristics*. Karakteristik kualitas eksternal merupakan bagian-bagian dari suatu produk yang berhubungan dengan para pemakainya, sedangkan karakteristik kualitas internal tidak secara langsung berhubungan dengan pemakai. *Software Quality* didefinisikan sebagai kesesuaian yang diharapkan pada semua *software* yang dibangun dalam hal fungsi *software* yang diutamakan dan unjuk kerja *software*, standar pembangunan *software* yang terdokumentasi dan karakteristik yang ditunjukkan oleh *software* [**1**]. Definisi ini menekankan pada 3 hal yaitu:

1. kebutuhan *software* adalah fondasi ukuran kualitas *software*, jika *software* Tidak sesuai dengan kebutuhan yang ditentukan maka kualitas pun kurang.
2. jika menggunakan suatu standar untuk pembangunan *software* maka jika *software* tidak memenuhi standar tersebut maka dianggap kurang berkualitas.

3. seringkali ada kualitas yang secara langsung diutarakan (tersirat) seperti kemudahan penggunaan dan pemeliharaan yang baik. Kualitas *software* dipertanyakan jika tidak memenuhi kebutuhan ini.

Hal lain yang perlu diperhatikan dalam kualitas perangkat lunak adalah *quality assurarance* (QA) yang merupakan *acctivity of providing revidence*. Selain itu harus adanya *software quality management* (SQM). Tujuan dari SQM adalah untuk mengembangkan suatu pemahaman kuantitatif dari kualitas proyek produk perangkat lunak dan mencapai tujuan spesifik kualitasnya.

Menurut IEEE, kualitas perangkat lunak didefinisikan sebagai derajat dari sebuah sistem, komponen atau proses yang memenuhi kebutuhan-kebutuhan yang dispesifikasikan dan kebutuhan pengguna atau harapan pengguna. Menurut Pressman, kualitas perangkat lunak adalah kesesuaian antara kebutuhan fungsional dan kebutuhan performansi yang terdefinisi, standar pengembangan yang terdokumentasi secara ekplisit dan karakteristik-karakteristik yang tersirat yang diharapkan dari perangkat lunak yang dikembangkan secara profesional [**5**].

II.3 Definisi Video

Video adalah teknologi untuk menangkap, merekam, memproses, mentransmisikan dan menata ulang gambar bergerak. Pada dasarnya terdapat dua jenis video dalam *layer* computer, yaitu *analog* dan *digital* video [**2**].

1. Video *Analog* merupakan produk dari industri pertelevisian dan oleh sebab itu dijadikan sebagai standar televisi. Meskipun banyak video yang diproduksi hanya untuk *platform display digital* (untuk Web, CD-ROM, atau sebagai presentasi HDTV DVD), video *analog* (kebanyakan masih digunakan untuk penyiaran televisi) masih merupakan *platform* yang paling banyak diinstal untuk mengirim dan melihat video. Standar Penyiaran Video *Analog* : Tiga standar penyiaran video *analog* yang paling banyak digunakan di dunia adalah NTSC, PAL, dan SECAM.
2. Video *Digital* adalah produk dari industri komputer dan oleh sebab itu dijadikan standar data *digital.* Integrasi Penuh dari video *digital* dalam

kamera dan komputer mengurangi televisi *analog* dari video dari produksi multimedia dan *platform* pengiriman, jika kamera video anda menggerakkan sinyal *output digital*, Anda dapat merekam video Anda langsung ke disk, yang siap untuk diedit. Jika sebuah video klip disimpan sebagai data pada *hard disk*, CD-ROM atau perangkat penyimpanan masal lain, Dunia video kini telah mengalami perubahan dari *analog* ke *digital*. Pada konsumen rumahan dan perkantoran kita dapat menikmati kualitas video digital yang prima lewat hadirnya teknologi VCD dan DVD (*Digital Versatile Disc*), sedangkan dunia *broadcasting* kini juga lambat laun mengalihkan teknologinya kearah DTV (*Digital Television*). Arsitektur Video *Digital* tersusun atas sebuah format untuk mengenkode dan memainkan kembali file video dengan komputer dan menyertakan sebuah *player* yang dapat mengenali dan membuka file yang dibuat untuk format tersebut. Arsitektur video digital yang utama adalah AppleQuicktime, *Microsoft Windows Media Format, dan Real Network RealMedia*. Format file video yang terkait adalah *QuickTime movie* (.mov), *Audio Video Interleaved*(.AVI), *Windows Media Video* (.wmv) , dan RealMedia (.rm). Beberapa player mengenali dan memainkan lebih dari satu format file video. Seperti halnya audio juga mengalami proses yang serupa yaitu biasanya direkam dan dimainkan sebagai sinyal *analog*. Untuk itulah harus dikonversi menjadi digital terlebih dahulu agar dapat diproses menjadi sebuah *multimedia title*.

II.4 Definisi VLC *Media Player*

VLC *Media Player* adalah program aplikasi multimedia yang berguna untuk memutar lagu dan video tetapi lebih diperuntukan untuk memutar video. Perangkat lunak VLC *Media Player* ini dibuat pada tahun 1998 dan dikembangkan oleh tim VideoLAN. Perangkat lunak komputer ini memiliki kelebihan pada programnya yang ringan, cepat, dan beragamnya format file *audio* dan *video* yang bisa dimainkannya. VLC *Media Player* mampu menutupi

kelemahan beberapa multimedia player populer yang tidak bisa memainkan format file tertentu [**2**].

VLC juga dikenal sebagai multimedia player *multi platform*, karena bisa diinstall di sembarang sistem operasi komputer. Player VLC juga bisa diinstall secara *portable* di media simpan eksternal, misalnya di USB *Flash Disk*. Sistem operasi untuk *smartphone* seperti Android pun sudah didukung.Program VLC *Media Player* banyak diaplikasikan dan digunakan oleh berbagai kalangan pengguna komputer. VLC *Media Player* merupakan *software* gratis (*open source*). VLC *Media Player* dapat dijalankan di sistem operasi *Windows* XP SP2, *Windows Vista*, *Windows* 7. Selain itu VLC juga menyediakan paket *software* untuk sistem operasi Mac OS X, GNU/Linux, bahkan Android. dipersyaratkan. Spesifikasi yang dibutuhkan pun cenderung sangat ringan dengan minimal memiliki kapasitas hardisk sebesar 70mb [**2**].

II.4 Model Metriks Kualitas Perangkat Lunak

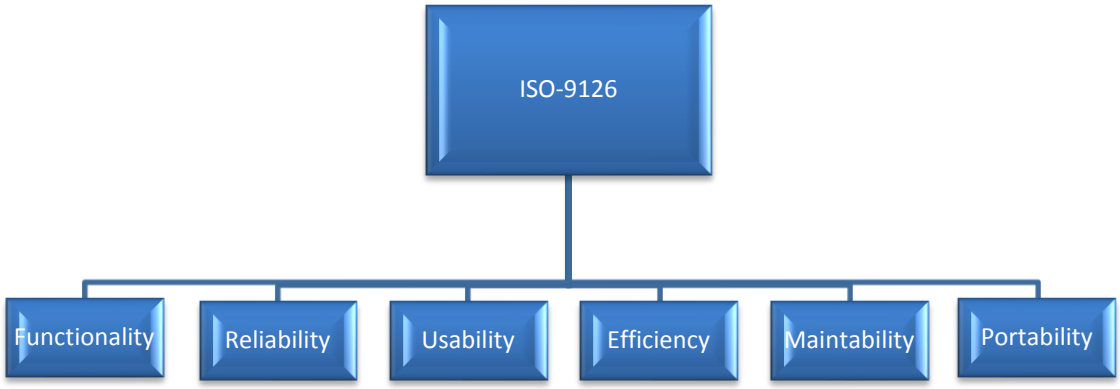
Dalam keilmuan rekayasa perangkat lunak terdapat beberapa model kualitas perangkat lunak. Pada setiap model ini terdapat beberapa faktor yang menjadi poin-poin utama dalam penilaian kualitas sebuah perangkat lunak. Berikut adalah penjelasan tentang model kualitas perangkat lunak yang akan diterapkan.

Model ISO-9126 dikenalkan pertama kali pada tahun 1991 sebagai standarisasi kualitas produk perangkat lunak. Standarisasi ini dibuat karena banyaknya model kualitas yang ditawarkan sebagai faktor kualitas perangkat lunak [**6**]. Dalam dokumen pertama model ISO-9126 terdiri dari beberapa bagian model kualitas untuk kualitas sebuah produk perangkat lunak, di antaranya:

1. *Internal quality.*
2. *external quality model.*
3. *Quality in use model.*

II.4.1 Model Kualitas Eksternal

Bagian pertama dari kualitas model tersebut menentukan 6 karakteristik yang mereka bagi kedalam 21 sub karakteristik untuk kualitas *internal* dan kualitas eksternal Berikut ini merupakan pengertian dari masing-masing faktor kualitas dan sub pada ISO-9126 [**4**].



Gambar II-1 Faktor ISO 9126

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.

Tabel II-1 *Suitability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Functional adequacy	X = 1-A/B A = jumlah fungsi yang terdapat masalah dalam evaluasi B = Jumlah fungsi yang di evaluasi	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik
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
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
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

b. *Accuracy*

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

Tabel II-2 *Accuracy* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Accuracy to expectation	X = A / T A = jumlah kasus yang dihadapi oleh pengguna dengan perbedaan terhadap hasil yang diharapkan T = Waktu operasi	0 <= X Semakin dekat ke 0 semakin baik
Computational Accuracy	X = A/T A = Jumlah perhitungan akurasi yang dihadapi oleh pengguna T = Waktu operasi	0 <= X Semakin dekat ke 0 semakin baik
Precision	X = A/T A = Jumlah hasil yang dihadapi oleh pengguna dengan tingkat presisi yang berbeda dari yang dibutuhkan waktu operasi	0 <= X Semakin dekat ke 0 semakin baik

c. *Interoperability*

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

Tabel II-3 *Interoperability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Data exchangeability (Data format based)	X = A/B A = jumlah 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
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	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

Nama Metriks	Pengukuran	Intrepretasi nilai
	B = Jumlah kasus di mana pengguna upaya untuk pertukaran data	

d. *Security*

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

Tabel II-4 *Security* Metriks

Nama Metrik	Pengukuran	Intrepretasi nilai
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
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
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

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.

Tabel II-5 *Functionality* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
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
Interface standard compliance	X = A/B A = Jumlah *interface* diterapkan dengan benar sebagaimana ditentukan B = jumlah *interface* yang membutuhkan pemenuhan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

2. Reliability

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

a. *Maturity*

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

Tabel II-6 *Maturity* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Estimated latent fault density	X = {ABS(A1 – A2)}/B (X:perkiraan yang kepadatan kesalahan laten) ABS ()= Absolut A1 = Jumlah kesalahan laten memprediksi dalam produk	0 <= X Hal ini tergantung pada tahap pengujian. Pada tahap selanjutnya lebih kecil lebih baik.

Nama Metriks	Pengukuran	Intrepretasi nilai
	perangkat lunak A2 = Jumlah sebenarnya terprediksi kesalahan. B = Produk Ukuran	
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.
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.
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
Mean time between failures (MTBF)	a).X = T1/A b).Y = T2 / A T1=Waktu operasi T2=Jumlah interval waktu yang berturut-turut 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
Test coverage	X = A / B	0 <= X <= 1

Nama Metriks	Pengukuran	Intrepretasi nilai
(Specified operation scenario testing coverage)	A=Jumlah kasus uji bener-bener dilakukan mewakili skenario operasi selama pengujiaan. B=Junlah kasus uji yang akan dilakukan untuk menutupi kebutuhan	Semakin mendekati 1.0 semakin lebih baik
Test maturity	X = A / B A=Jumlah kasus uji lulus selama pengujian atau operasi B=Jumlah kasus uji yang akan dilakukan untuk menutupi kebutuhan	0 <= X <= 1 Semakin mendekati 1.0 semakin lebih baik

b. *Fault Tolerance*

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

Tabel II-7 *Fault Tolerence* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Breakdown avoidance	X = 1-A/B A = Jumlah kerusakan B = Jumlah kegagalan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik
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
Incorrect operation avoidance	X = A/B A = Jumlah kejadian kegagalan kritis dan serius	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

Nama Metriks	Pengukuran	Intrepretasi nilai
	terhadap kasus uji dari pola kesalahan B=Jumlah kasus uji dieksekusi dari pola kesalahan.	

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.

Tabel II-8 *Recoverability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Availability	a).X={To/(To+Tr)} To=Waktu untuk operasi Tr=Waktu untuk memperbaiki 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
Mean down time	X=T/N T= Jumlah *down time* N=Jumlah kerusakan yang diamati kasus terburuk atau distribusi down time harus diukur.	0 <X Semakin dekat ke 1.0 semakin baik
Mean recovery time	X=Sum(T)/N T=Waktu untuk pemulihan sistem perangkat lunak pada setiap kesempatan B=Jumlah kasus yang diamati	0 <X Semakin dekat ke 1.0 semakin baik

Nama Metriks	Pengukuran	Intrepretasi nilai
	sistem *software* masuk *recovery*	
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 *restart* dengan mudah.
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.
Restore effectiveness	X=A/B A=Jumlah kasus berhasil dipublikasikan memenuhi terget mengembalikan waktu B=Jumlah kasus yang dilakukan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik, karena proses restorasi di produk lebih efektif.

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.

Tabel II-9 *Reliability Compliance* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Reliability compliance	X = 1 - A / B A = Jumlah item kepuasan keandalan ditentukan yang belum dilaksanakan selama	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

Nama Metriks	Pengukuran	Intrepretasi nilai
	pengujian B = Total jumlah item kepatuhan keandalan ditentukan	

3. Usability

Usability berhubungan 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.

Tabel II-10 *Understandbility* Metriks

Nama Metrik	Pengukuran	Intrepretasi nilai
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.
Demonstration accessibility	X = A/B A = Jumlah demonstrasi / tutorial yang pengguna berhasil mengaksesnya B =Jumlah demonstrasi /tutorial yang tersedia	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.
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 selama pengamatan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik.

Nama Metrik	Pengukuran	Intrepretasi nilai
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.
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.
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.
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.

b. *Learnability*

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

Tabel II-11 *Learnability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Ease of function learning	T= Berarti waktu yang dibutuhkan untuk pembelajaran menggunakan fungsi dengan benar	0 < T Semakin pendek semakin baik
Ease of learning	T = Jumlah waktu operasi	0 < T

Nama Metriks	Pengukuran	Intrepretasi nilai
to perform a task in use	pengguna sampai waktu dicapai untuk melakukan tugas tertentu dalam waktu singkat	Semakin pendek semakin a = baik
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

c. *Operability*

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

Tabel II-12 *Operability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Operational consistency in use	a).X = 1-A/B A = Jumlah pesan atau fungsi uang menemukan data yang tidak konsisten dengan ekspektasi pengguna B = Jumlah pesan atau fungsi b).Y=N/UOT N= Jumlah pesan atau fungsi yang menemukan data yang tidak konsisten dengan ekspektasi pengguna 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

d. *Attractiveness*

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

Tabel II-13 *Attractiveness* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Attractive interaction	Kuesioner untuk menilai daya tarik dari antarmuka pengguan, setelah pengguna mengunakan.	Tergantung pada metode kuesioner scoringnya
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

e. *Usability Compliance*

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

Tabel II-14 *Usability Compliance* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Usability compliance	X = 1-A/B A = Jumlah item kepuasan kegunaan tertentu yang belum dilaksanakan selama pengujian B = Total jumlah item kepuasan kegunaan tertentu	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

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.

Tabel II-15 *Time Behaviour* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Response time	T = (Waktu untuk mendapatkan hasil)-(saat masuk perintah selesai)	0 < T Semakin cepat semakin baik.
Response time (Mean time to response)	Tmean = (Ti) ∑/ N, (untuk i = 1 sampai N) TXmean = diperlukan waktu respon rata-rata Ti = waktu respon untuk evaluasi ke-i (shot) N = jumlah evaluasi (tembakan sampel)	0 <= X Semakin dekat ke 1.0 dan kurang dari 1.0 lebih baik
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

b. *Resource Utilization*

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

Tabel II-16 *Resource* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
I/O devices utilisation	X = A/B A = Waktu I/O perangkat diduduki	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

Nama Metriks	Pengukuran	Intrepretasi nilai
	B = Ditentukan waktu yang dirancang untuk menempati	
I/O loading limits	X = Amax/Rmax Amax = MAX (Ai), (untuk i = 1 sampai N) Rmax = maksimum yang diperlukan I / O pesan MAX (Ai) = Jumlah maksimum pesan I / O dari 1 untuk evaluasi ke-i. N = jumlah evaluasi.	0 <= X Semakin kecil semakin baik
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
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
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

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.

Tabel II-17 *Efficiency Compliance* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
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

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.

Tabel II-18 *Analyzability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
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
Diagnostic function support	X = 1-A/B A = Jumlah kegagalan yang	0 <= X <= 1 Semakin dekat ke 1.0

Nama Metriks	Pengukuran	Intrepretasi nilai
	pemelihara dapat diagnos e (menggunakan fungsi diagnostik) untuk memahami penyebab - kapal hubungan efek B = Total jumlah kegagalan terdaftar	semakin baik

b. *Changeability*

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

Tabel II-19 Changeability Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Change cycle efficiency	*Average Time* : Tav = Sum(Tu)/N 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	0<Tav Semakin pendek lebih baik kecuali dari jumlah versi revisi itu besar

Nama Metriks	Pengukuran	Intrepretasi nilai
Change implementation elapse time	Average Time : Tav = Sum(Tm) / N Tm=T *out* – T *in* T *out* = 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
Modification complexity	T = Sum (A / B) / N A = Kerja dihabiskan untuk mengubah B = Ukuran perubahan *software* N = Jumlah perubahan	0 < T Semakin dekat ke 1.0 semakin baik
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
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

c. *Stability*

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

Tabel II-20 *Stability* Metriks

Nama Metriks	Pengukuran	Interpretasi nilai
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
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

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.

Tabel II-21 *Testability* Metriks

Nama Metrik	Pengukuran	Intrepretasi nilai
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
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
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 B = Jumlah kasus jeda melaksanakan uji coba	0 <= X Semakin dekat ke 1.0 semakin baik

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.

Tabel II-22 *Maintability Compliance* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
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

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.

Tabel II-23 *Adaptibility* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
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
Hardware environmental adaptability	X = 1-A/B A = Jumlah fungsional operasi tugas tidak selesai atau tidak cukup menghasilkan untuk memenuhi selama *combined* pengujian operasi dengan hardware lingkungan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

Nama Metriks	Pengukuran	Intrepretasi nilai
	B = Total jumlah fungsi yang diuji	

b. *Installability*

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

Tabel II-24 *Instability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Ease of installation	X = A/B A = Jumlah kasus yang pengguna berhasil mengubah 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
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

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.

Tabel II-25 *Co-existance* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Available co-existence	X = A/T A = Jumlah hambatan (kegagalan) tak terduga yang pengguna temukan selama operasi dengan *software* lain T = Waktu durasi bersamaan operasi perangkat lunak lain	0 <= X Semakin dekat ke 1.0 semakin baik

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.

Tabel II-26 *Replaceability* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
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
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
User support functional	X = 1-A1/A2 A = Jumlah fungsi baru yang	0 <= X Semakin dekat ke 1.0

consistency	ditemukan pengguna , konsisten dengan ekspektasi pengguna B = Jumlah fungsi baru	semakin baik

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.

Tabel II-27 *Portability Compliance* Metriks

Nama Metriks	Pengukuran	Intrepretasi nilai
Portability compliance	X = 1-A/B A = Jumlah portabilitas item kepatuhan ditentukan yang belum dilaksanakan selama pengujian B = Total jumlah portabilitas item kepatuhan ditentukan	0 <= X <= 1 Semakin dekat ke 1.0 semakin baik

II.4.2 Model *Quality In Use*

Model Quality In Use mempunyai 4 Faktor yaitu *Effectiveness, Productivity, Safety dan Satisfaction*. Masing masing faktor juga mempunyai beberapa metriks. Berikut penjelasannya :

1. *Effectiveness*

   Effectiveness (Efektivitas) berhubungan dengan seberapa efektif suatu perangkat lunak terhadap penggunanya.

Tabel II-28 *Effectiveness*

Karakteristik	Metriks	Rumus
Effectiveness	*Task completion*	X = A/B A = jumlah tugas diselesaikan B = jumlah tugas yang dicoba
	Error frequency	X = A/T A = jumlah kesalahan yang dibuat oleh

Karakteristik	Metriks	Rumus
		pengguna T = jumlah tugas dicoba

2. *Productivity*

Productivity (Produktivitas) berhubungan dengan seberapa produktif kah perangkat lunak terhadap penggunanya.

Tabel II-29 *Productivity*

Karakteristik	Metriks	Rumus
Productivity	*Task time*	X = Ta Ta = Waktu mencoba
	Task efficiency	X = M1 / T M1 = efektivitas tugas T = Waktu
	Economic productivity	X = M1 / C M1 = efektivitas tugas C = total biaya tugas
	Productive proportion	X = Ta / Tb Ta = waktu produktif = waktu tugas – waktu bantu - waktu error - waktu pencarian Tb = waktu tugas
	Relative user efficiency	*Relative user efficiency* X = A / B A = efisiensi tugas pengguna biasa ini B = efisiensi tugas ahli pengguna

3. *Safety*

Safety (Keamanan) berhubungan dengan seberapa aman perangkat lunak yang dianalisis terhadap penggunanya.

Tabel II-30 *Safety*

Karakteristik	Metriks	Rumus
Safety	*Safety of people affected by use of the system*	X = 1-A / B A = jumlah orang menaruh di hazard B = Jumlah orang yang berpotensi terkena dampak oleh sistem
	Economic damage	X = 1-A / B A = jumlah kejadian kerusakan ekonomi B = Jumlah situasi penggunaan
	Software damage	X = 1-A / B A = jumlah kejadian perangkat lunak corrupt B = Jumlah situasi penggunaan

4. *Satisfaction*

   Satisfaction (Kepuasan) berhubungan dengan kepuasan pelanggan terhadap perangkat lunak yang digunakan.

Tabel II-31 *Satisfaction*

Karakteristik	Metriks	Rumus
Satisfaction	*Discretionary usage*	X = A / B A = berapa kali fungsi perangkat lunak tertentu / aplikasi / sistem yang digunakan B = jumlah pengguna menggunakan
	Satisfaction questionnaire	X = A / n Ai) menanggapi pertanyaan n = jumlah tanggapan

II.5 Metode Pengukuran Kualitas Perangkat Lunak

Pengukuran kualitas perangkat lunak merupakan hal yang terpenting dari praktek rekayasa perangkat lunak yang baik. Serta pengukuran aktivitas manusia menambah nilai dan terlibat aktif dalam proses informasi. Pengukuran ini membantu untuk membuat karakteristik khusus dari proses dan produk informasi. Pengukuran meliputi evaluasi kuantitatif yang biasanya menggunakan metriks serta ukuran dapat menggunakan secara langsung menentukan pencapaian tujuan

kualitas numerik. Pengukuran selalu menjadi fundamental bagi kemajuan untuk setiap disiplin rekayasa dan pengujian perangkat lunak. pengukuran perangkat lunak dan metrik menjadi peran mendasar dalam siklus hidup pengembangan perangkat lunak, dan Metriks 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.

Pengukuran kualitas perangkat lunak pemutar video akan menggunakan Metriks ISO 9126 yang merupakan turunan dari Metode ISO 9126. Metriks 9216 ini melakukan perhitungan *Internal* dan *External* berdasarkan karakteristik dari 6 faktor yang ada pada model ISO 9126 [**4**].

Metriks *Internal* Dapat digunakan untuk membantu memprediksi apakah perangkat lunak akan memenuhi kebutuhan keandalan organisasi selama pengembangan sistem perangkat lunak. *Maturity* Metrik menunjukkan satu set atribut untuk menilai kematangan suatu *software*. Deteksi kesalahan adalah Sebuah ukuran seberapa banyak *defects* terdeteksi dalam produk perangkat lunak. Metrik ini dikumpulkan dengan menghitung jumlah terdeteksi *bug* yang ditemukan dalam *review* dan membandingkan angka itu dengan jumlah yang diperkirakan. dapat ditemukan dalam tahap pengujian statis. Metriks yang dihitung oleh rumus X = A / B , dimana A adalah jumlah absolut *bug* terdeteksi (dari laporan *review*) dan B adalah perkiraan jumlah *expected*.2 Nilai tinggi untuk metriks ini menyiratkan baik tingkat kualitas produk.

Penghapusan kesalahan adalah Sebuah ukuran seberapa banyak cacat yang ditemukan selama *review* dihapus (dikoreksi) selama desain dan implementasi tahap. Metrik dapat dihitung dengan rumus X = A / B. dimana A adalah jumlah *bug* tetap selama desain dan *coding* dan B adalah Jumlah yang ditemukan selama *review*. Semakin dekat nilai adalah 1, yang lebih baik. A Nilai penghapusan kesalahan orang akan berarti bahwa setiap cacat terdeteksi telah dihapus.

Metriks Eksternal ISO 9126 mempunyai beberapa metrik eksternal. Metriks eksternal didefinisikan dalam ISO 9126-2 dan diukur selama pengujian dinamis. Kategori metrik ini sesuai dengan yang internal *yaitu maturity, fault*

tolerance, recoverability, and compliance. Metriks Kematangan mengukur kebebasan sistem perangkat lunak dari kegagalan disebabkan oleh *bug* di sistem itu sendiri. Faktor eksternal ini lebih menilai dari segi program kepada pengguna sedangkan internal lebih menilai dari segi program tersebut [**6**].

Perhitungan Metriks Pada metriks 9126 terdapat 6 faktor dan beberapa subfaktor yang akan digunakan sebagai perhitungan yang tentunya akan dilihat dari karakteristik perangkat lunak tersebut. Setiap faktor karakteristik mempunyai hitungan tersendiri yang sudah ditentukan oleh metriks 9126 [**6**].

Tabel II-32 Subfaktor Metriks

Funcionality Metrics	*Suitability metrics* *Accuracy metrics* *Interoperability metrics* *Security metrics* *Functionality compliance metrics*
Reliability metrics	*Maturity metrics* *Fault tolerance metrics* *Recoverability metrics* *Reliability compliance metrics*
Usability Metrics	*Understandability metrics* *Learnability metrics* *Operability metrics* *Attractiveness metrics* *Usability compliance metrics*
Efficiency metrics	*Time behaviour metrics* *Resource utilisation metrics* *Efficiency compliance metrics*
Maintainability metrics	*Analysability metrics* *Changeability metrics* *Stability metrics* *Testability metrics*

	Maintainability compliance metrics
Portability metrics	*Adaptability metrics* *Installability metrics* *Co-existence metrics* *Replaceability metrics* *Portability compliance metrics*

II.6 Pengukuran

Pengukuran perangkat lunak adalah faktor utama untuk mengetahui seberapa baik perangkat lunak yang akan dinilai berdasarkan metode yang digunakan. Pengukuran perangkat lunak dilakukan dengan cara mencari nilai kriteria perangkat lunak tersebut dan juga mencari bobot kepentingan berdasarkan metode yang akan digunakan. Penilaian keseluruhan dapat diperoleh dari hasil nilai kriteria dan juga nilai bobot. Perhitungan bobot dan nilai kriteria bisa bervariasi tergantung dengan metode apa yang digunakan [**10**] [**11**].

II.7 Metode Pembobotan

Rank Order Centroid (ROC) didasarkan pada tingkat kepentingan atau prioritas dari kriteria. Menurut Jeffreys dan Cockfield dalam Afiefah Rahma, 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 ke n, ditulis . Untuk menentukan bobotnya, diberikan aturan yang sama yaitu dimana merupakan bobot untuk kriteria[**6**] [**7**]. Atau dapat dijelaskan sebagai berikut:

Jika
$Cr1 \geq Cr2 \geq Cr3 \geq ... \geq Crn$

Maka
$W1 \geq W2 \geq W3 \geq ... \geq Wn$

Selanjutnya, jika k merupakan banyaknya kriteria, maka

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

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

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

$$Wk = \frac{0 + 0 + 0 + \cdots \ldots . + 1/k}{k}$$

Secara umum pembobotan ROC dapat dirumuskan sebagai berikut :

$$Wk = 1/k \sum_{t=1}^{k} (1/\mathrm{i})$$

II.8 Metode Pengolahan Kuesioner

Kuesioner merupakan sekumpulan pertanyaan yang diajukan kepada responden dengan tujuan dan maksud tertentu untuk memperoleh data, sehingga data tersebut dapat digunakan atau diolah kembali. Untuk mengolah data dari kuesioner tersebut dapat menggunakan berbagai macam metode pengolahan data sesuai dengan rumus metriks yang digunakan pada ISO 9126

Kuisioner ini diajukan untuk memenuhi nilai A, B maupun T yang dibutuhkan untuk menghitung nilai metriks dai setiap kriteria yang sudah ada. Dari hasil jawaban yang diberikan oleh responden akan dicari nilai rata rata, hasil jawaban responden akan dijumlahkan dan totalnya akan dibagi dengan jumlah responden. Maka akan didapat nilai A, B ataupun T.

II.9 Metode Penilaian

Perhitungan nilai sub faktor , faktor maupun nilai keseluruhan menjadi faktor penting untuk mengetahui hasil akhir sebuah nilai. menentukan nilai kriteria per faktor nya dan nilai akhir dengan rumus perhitungan *Weight Summation*. Nilai w adalah nilai metriks yang diperoleh dari hasil pengolahan kuisioner dan m adalah bobot dari sub faktor atau faktor yang sudah diperoleh dari hasil perhitungan dengan menggunakan metode *Rank Order Centroid* (ROC) [**8**].

1 Nilai kriteria

$$\text{Kriteria (a)} = \sum_{j=1}^{M} w_{j.}\, m_{ij} \ldots\ldots\ldots$$

$$= w_1 m_1 + w_2 m_2 + \ldots\ldots \text{ wnmn}$$