BAB II
TINJAUAN PUSTAKA

2.1 Sistem Informasi

2.1.1 Sistem

Sistem didefinisikan sebagai sekumpulan komponen yang saling terkait, dengan batas yang didefinisikan dengan jelas, bekerja bersama untuk mencapai serangkaian tujuan bersama dengan menerima masukan dan menghasilkan *output* dalam proses transformasi yang terorganisir. Banyak contoh sistem dapat ditemukan dalam ilmu fisik dan biologi, dalam teknologi modern, dan dalam masyarakat manusia. Dengan demikian, dapat berbicara tentang sistem fisik matahari dan planet-planetnya, sistem biologis tubuh manusia, sistem teknologi dari kilang minyak, dan sistem sosioekonomi dari organisasi bisnis [2].

Sistem memiliki tiga fungsi dasar, yaitu [2]:

1. *Input*: Melibatkan elemen menangkap dan merakit yang memasuki sistem untuk diproses. Misalnya, bahan mentah, energi, data, dan usaha manusia harus diamankan dan diorganisasikan untuk diproses.
2. *Process*: Melibatkan proses transformasi yang mengubah *input* menjadi *output*. Contohnya adalah proses manufaktur, proses pernafasan manusia, atau perhitungan matematis.
3. *Output*: Melibatkan mentransfer elemen yang telah dihasilkan oleh proses transformasi ke tujuan akhir mereka. Misalnya, produk jadi, layanan manusia, dan informasi manajemen harus dikirim ke pengguna.

Selain 3 (tiga) fungsi dasar yang berinteraksi tersebut, sistem juga memiliki 2 (dua) komponen tambahan, yaitu [2]:

1. Umpan balik adalah data mengenai kinerja sistem.
2. Pengendalian melibatkan pengawasan dan pengevaluasian umpan balik untuk menetapkan apakah sistem bergerak menuju pencapaian tujuan atau tidak.

2.1.2 Informasi

Informasi adalah kumpulan fakta yang terorganisir dan diproses sehingga memiliki nilai tambah di luar nilai fakta individu. Jenis informasi yang dibuat tergantung pada hubungan yang didefinisikan di antara data yang ada [3].

Informasi secara langsung terkait dengan bagaimana hal itu membantu pengambil keputusan mencapai tujuan organisasi. Informasi berharga dapat membantu orang dan organisasi melakukan tugas secara lebih efisien dan efektif. Informasi dapat menimbang perkiraan pasar yang memprediksi tingginya permintaan akan produk baru [3].

Informasi akan berguna bagi manajer dan pembuat keputusan apabila informasi tersebut memiliki karakteristik. Karakteristik informasi adalah sebagai berikut [2].

Tabel 2.1 Karakteristik Informasi

Karakteristik	Pengertian
Dapat Diakses	Informasi harus mudah diakses oleh pengguna yang berwenang sehingga mereka dapat memperolehnya dalam format yang tepat dan pada waktu yang tepat untuk memenuhi kebutuhan mereka.
Akurat	Informasi akurat tidak mengandung kesalahan. Dalam beberapa kasus, informasi yang tidak akurat dihasilkan karena data yang tidak akurat dimasukkan ke dalam proses transformasi (ini biasa disebut sampah masuk, sampah keluar (GIGO)).
Lengkap	Informasi lengkap mengandung semua fakta penting. Misalnya, laporan investasi yang tidak mencakup semua biaya penting tidak lengkap.
Ekonomis	Informasi juga harus relatif ekonomis untuk diproduksi. Pembuat keputusan harus selalu menyeimbangkan nilai informasi dengan biaya memproduksinya.
Fleksibel	Informasi yang fleksibel dapat digunakan untuk berbagai keperluan. Misalnya, informasi tentang berapa banyak persediaan yang tersedia untuk bagian tertentu dapat digunakan oleh perwakilan penjualan dalam menutup penjualan, oleh manajer produksi untuk menentukan apakah lebih banyak persediaan diperlukan, dan oleh seorang eksekutif keuangan untuk menentukan nilai total perusahaan telah berinvestasi dalam persediaan.
Relevan	Informasi yang relevan adalah penting bagi pembuat keputusan. Informasi yang menunjukkan bahwa harga kayu mungkin turun mungkin tidak relevan dengan produsen *chip* komputer.

Tabel 2.1 Karakteristik Informasi (Lanjutan)

Karakteristik	Pengertian
Dipercaya	Informasi yang dapat dipercaya oleh pengguna. Dalam banyak kasus, keandalan informasi tergantung pada keandalan metode pengumpulan data. Dalam kasus lain, keandalan tergantung pada sumber informasi. Sebuah rumor dari sumber yang tidak diketahui bahwa harga minyak mungkin naik mungkin tidak dapat diandalkan.
Aman	Informasi harus aman dari akses oleh pengguna yang tidak sah.
Sederhana	Informasi harus sederhana, tidak terlalu rumit. Informasi yang canggih dan terperinci mungkin tidak diperlukan. Pada kenyataannya, terlalu banyak informasi dapat menyebabkan informasi yang berlebihan, dimana pembuat keputusan memiliki terlalu banyak informasi dan tidak dapat menentukan apa yang benar-benar penting. Aman dari akses oleh pengguna yang tidak sah.
Tepat Waktu	Informasi tepat waktu disampaikan ketika dibutuhkan. Mengetahui kondisi cuaca minggu lalu tidak akan membantu ketika mencoba memutuskan mantel apa yang akan dikenakan hari ini.
Verifikasi	Informasi harus dapat diverifikasi. Ini berarti dapat memeriksanya untuk memastikan itu benar, mungkin dengan memeriksa banyak sumber untuk informasi yang sama.

2.1.3 Sistem Informasi

Sistem informasi merupakan kombinasi teratur apapun dari orang-orang, perangkat keras, perangkat lunak, jaringan komunikasi, dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi. Orang bergantung pada sistem informasi untuk berkomunikasi antara satu sama lain dengan menggunakan berbagai jenis alat fisik (perangkat keras), perintah dan prosedur pemrosesan informasi (perangkat lunak), saluran komunikasi (jaringan), dan data yang disimpan (sumber daya data) sejak permulaan peradaban [2].

Mengembangkan solusi sistem informasi yang berhasil baik mengatasi masalah bisnis adalah tantangan utama untuk para manajer bisnis dan praktisi bisnis saat ini. Seorang praktisi bisnis bertanggung jawab untuk mengajukan atau mengembangkan teknologi informasi baru atau meningkatkan bagi organisasi. Sebagian besar sistem informasi berbasis komputer disusun, didesain, dan diimplementasikan dengan menggunakan beberapa bentuk proses pengembangan yang sistematis. Di dalam proses pengembangan ini, para pemakai akhir dan spesialis

informasi mendesain aplikasi sistem informasi berdasarkan pada analisis kebutuhan bisnis suatu organisasi [2].

Sistem informasi bergantung pada sumber daya manusia (pemakai akhir dan pakar SI), *hardware* (mesin dan media), *software* (program dan prosedur), data (dasar data dan pengetahuan), serta jaringan (media komunikasi dan dukungan jaringan) untuk melakukan *input*, pemroresan, *output*, penyimpanan, dan aktivitas pengendalian yang mengubah sumber data menjadi produk informasi. Model sistem informasi ini memperlihatkan hubungan antarkomponen dan aktivitas sistem informasi. Model tersebut memberikan kerangka kerja yang menekankan pada empat konsep utama yang dapat diaplikasikan ke semua jenis sistem informasi, yaitu [2]:

1. Manusia, perangkat keras, perangkat lunak, data, dan jaringan adalah lima sumber daya dasar sistem informasi.
2. Sumber daya manusia meliputi pemakai akhir dan pakar SI, sumber daya perangkat keras terdiri dari mesin dan media, sumber daya perangkat lunak meliputi baik program maupun prosedur, sumber daya data dapat meliputi dasar data dan pengetahuan, serta sumber daya jaringan yang meliputi media komunikasi dan jaringan.
3. Sumber daya data diubah melalui aktivitas pemrosesan sistem informasi menjadi berbagai produk informasi bagi pemakai akhir.
4. Pemrosesan informasi terdiri dari aktivitas *input* dalam sistem, pemrosesan, *output*, penyimpanan, dan pengendalian.

2.1.4 Pengelompokan Sistem Informasi

Pada dasarnya istilah sistem informasi mengacu pada istilah *Computer Based Information System* (CBIS), dengan kata lain sistem informasi yang menggunakan teknologi komputer. Terdapat 4 (empat) jenis CBIS, yaitu [4]:

1. *Transaction Processing System* (TPS)

   Sistem informasi terkomputerisasi yang digunakan untuk memroses sejumlah besar transaksi bisnis. Hal yang bisa dilakukan sistem ini meliputi:

   a. Mengotomatisasi penanganan data aktifitas bisnis dan transaksi.
   b. Menangkap data dari setiap transaksi.

c. Memverifikasi transaksi untuk diterima atau ditolak.
d. Menghasilkan laporan rangkuman transaksi.

2. *Management Information System* (MIS)

   Sistem informasi manajemen adalah sebuah sistem informasi pada level manajemen yang berfungsi untuk membantu perencanaan, pengendalian, dan pengambilan keputusan. Sistem ini mengambil data yang ada di TPS yang akan diubah menjadi data yang lebih berarti yang dibutuhkan oleh manajer dalam menjalankan tugas dan tanggung jawabnya.

3. *Decision Support System* (DSS)

   Merupakan sistem informasi pada level manajemen dari suatu organisasi yang mengkombinasikan data dan model analisis canggih atau peralatan data analisis untuk mendukung pengambilan yang semiterstruktur dan tidak terstruktur yang dirancang untuk membantu pengambilan keputusan organisasional. DSS biasanya tersusun dari:

   a. *Database* (bisa diekstraksi dari TPS/MIS).
   b. Model grafis atau matematis untuk proses bisnis.
   c. Antarmuka pengguna, yang digunakan oleh pengguna untuk berkomunikasi dengan DSS.

4. *Expert System and Artificial Intelligence* (ES & AI)

   Expert System (ES) merupakan representasi pengetahuan yang menggambarkan cara seorang ahli dalam mendekati suatu masalah. ES lebih berpusat pada bagaimana mengkodekan dan memanipulasi pengetahuan dari informasi (misalnya aturan *If...Then*). Adapun cara kerjanya sebagai berikut:

   a. Pengguna berkomunikasi dengan sistem menggunakan dialog interaktif.
   b. ES menanyakan pertanyaan (yang akan ditanyakan seorang pakar) dan pengguna memberikan jawaban.
   c. Jawaban digunakan untuk menentukan aturan mana yang dipakai dan ES sistem menyediakan rekomendasi berdasarkan aturan yang telah disimpan.
   d. Seorang *Knowledge Engineer* bertanggunag jawab pada bagaimana melakukan akuisisi pengetahuan, sama seperti seorang analisis, tetapi dilatih untuk menggunakan teknik yang berbeda.

2.2 Rekayasa Perangkat Lunak

Rekayasa Perangkat Lunak (*Software Engineering*) merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunakan mesin. Perangkat lunak banyak dibuat dan pada akhirnya sering tidak dipergunakan karena tidak memenuhi kebutuhan pelanggan atau karena masalah non-teknis, seperti keengganan pemakai perangkat lunak (*user*) untuk mengubah cara kerja dari manual ke otomatis, atau ketidakmampuan *user* menggunakan komputer. Rekayasa perangkat lunak fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat kepada pelanggan (*customer*). Adapun kriterianya yaitu [5]:

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi (*maintainability*).
2. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*dependability* dan *roburst*).
3. Efisiensi dari segi sumber daya dan penggunaan.
4. Kemampuan untuk dipakai sesuai dengan kebutuhan (*usability*).

Dari kriteria di atas, maka perangkat lunak yang baik adalah perangkat lunak yang dapat memenuhi kebutuhan pengguna (*customer*) atau pemakai, bukan berorientasi pada pengembangan perangkat lunak [5].

The Software Engineering Body of Knowledge (SWEBOK) membagi rekayasa perangkat lunak ke dalam 10 (sepuluh) area pengetahuan, yaitu [5]:

1. Kebutuhan perangkat lunak

   Tahap-tahap analisis kebutuhan *software*.
2. Perancangan perangkat lunak

   Bagaimana menentukan arsitektur, komponen, antarmuka, dan karakterisik dari sistem yang akan dibuat.
3. Konstruksi perangkat lunak

   Kegiatan yang dilakukan untuk mengevaluasi kualitas *software* dan untuk meningkatkannya dengan mengidentifikasi kerusakan-kerusakan dan berbagai masalah yang dihadapi. *Software testing* terdiri dari verifikasi yang dinamis terhadap perilaku program pada satu set terbatas kasus uji, sesuai yang dipilih dari domain eksekusi, biasanya terbatas terhadap perilaku yang diharapkan.

4. Pengujian perangkat lunak

   Kegiatan yang dilakukan untuk mengevaluasi kualitas *software* dan untuk meningkatkannya dengan mengidentifikasi kerusakan-kerusakan dan berbagai masalah yang dihadapi. *Software testing* terdiri dari verifikasi yang dinamis terhadap perilaku program pada satu set terbatas kasus uji, sesuai yang dipilih dari domain eksekusi, biasanya terbatas terhadap perilaku yang diharapkan.

5. Pemeliharaan perangkat lunak

   Memelihara produk perangkat lunak dari awal pembuatan sampai pemakaian perangkat lunak. Setiap perubahan yang tejadi pada produk perangkat lunak akan dicatat dalam bentuk *log*. Perubahan perangkat lunak menyebabkan perubahan kode program (*coding*), memerlukan *test*, sampai menghasilkan produk dengan versi yang baru.

6. Manajemen konfigurasi perangkat lunak

   Merupakan *tools* manajemen konfigurasi yang dipakai untuk menyimpan versi komponen sistem, membangun sistem dari komponen, dan memantau rilis versi sistem ke pelanggan beserta hasil laporannya. *Tools* yang digunakan untuk mengatur sekumpulan aktivitas yang dirancang untuk mengontrol perubahan adalah sebagai berikut:

   a. Mengidentifikasi jenis produk yang akan diubah
   b. Audit dan *report* perubahan yang dibuat
   c. Mengontrol perubahan
   d. Menentukan mekanisme untuk me-*manage* versi yang berbeda dari jenis produk
   e. Mengelompokkan produk

7. Manajemen perangkat lunak

   Membahas aplikasi dari aktivitas-aktivitas manajemen, seperti *planning*, *coordinating*, *measuring*, *monitoring*, *controlling*, dan *reporting* untuk memastikan bahwa pembangunan dari suatu perangkat lunak dilakukan secara sistematik, disiplin, dan sesuai perkiraan. Oleh sebab itu, *software engineering management knowledge area* memetakan manajemen dari pembangunan perangkat lunak dengan perkiraan dan pemodelan dari pembangunan perangkat lunak tersebut.

8. Proses perangkat lunak

   Untuk membangun suatu perangkat lunak tentu membutuhkan proses-proses tertentu yang disebut dengan model proses. Ada beberapa macam model proses di dalam *Sofware Engineering*, seperti *Linear Sequential Model* (yang sering dikenal dengan Model *Waterfall*), kemudian *prototyping*, dan lain sebagainya.

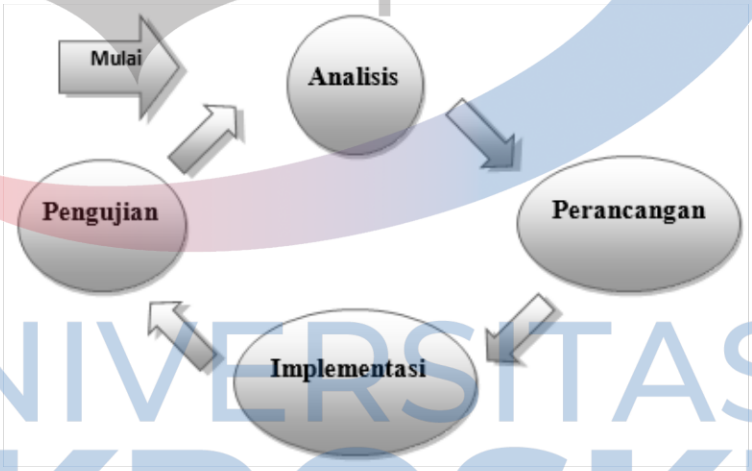
9. Metode dan *tool* perangkat lunak

   Membahas kajian teoritis tentang alat bantu dan metode rekayasa perangkat lunak.

10. Kualitas perangkat lunak

    Software Quality menitikberatkan pada kualitas dan daur hidup perangkat lunak.

Gambar berikut ini menunjukkan tahapan-tahapan umum di dalam rekayasa perangkat lunak [5].



Gambar 2.1 Tahapan Umum Rekayasa Perangkat Lunak

2.3 Pengujian Perangkat Lunak

Berdasarkan standar IEEE, pengujian perangkat lunak memiliki pengertian aktivitas yang dilakukan untuk mengevaluasi kualitas produk dan untuk mengembangkannya dengan mengidentifikasi kelemahan dan permasalahan yang terjadi. Definisi secara umum adalah pengujian perangkat lunak terdiri dari verifikasi dinamis perilaku program pada sekumpulan kasus-kasus pengujian yang terbatas. Pada umumnya dipilih dengan tepat dari domain eksekusi yang tak terbatas dan berlawanan dengan perilaku yang diharapkan [5].

Sebuah perangkat lunak perlu dijaga kualitasnya, bahwa kualitas bergantung kepada kepuasan pelanggan. Kualitas perangkat lunak perlu dijaga untuk keperluan sebagai berikut [5]:

1. Agar dapat bertahan hidup di bisnis perangkat lunak
2. Dapat bersaing dengan perangkat lunak lainnya
3. Penting untuk pemasaran global
4. Mengefektifkan biaya agar tidak banyak membuang perangkat lunak karena kegagalan pemasaran atau kegagalan produksi
5. Mempertahankan pelanggan (*customer*) dan meningkatkan keuntungan

Sering perangkat lunak mengandung kesalahan (*error*) pada proses-proses tertentu pada saat perangkat sudah berada di tangan *user*. Kesalahan-kesalahan ini disebut *bug*. Untuk menghindari banyak *bug* diperlukan adanya pengujian selama diberikan ke pelanggan atau dalam masa pengembangan. Kelakuan perangkat lunak yang tidak sesuai dengan spesifikasi yang dibutuhkan bisa dianggap sebagai *bug*. Pengujian perangkat juga bisa meminimalisir adanya kesalahan non-teknis seperti adanya pesan kesalahan sehingga *user* tidak bingung dengan adanya pesan kesalahan yang muncul [5].

Pengujian perangkat lunak juga merupakan sebuah elemen topik yang memiliki cakupan luas dan sering dikaitkan dengan verifikasi (*verification*) dan validasi (*validation*). Verifikasi mengacu pada sekumpulan aktivitas yang menjamin bahwa perangkat lunak mengimplementasikan dengan benar sebuah fungsi yang spesifik. Validasi mengacu pada sekumpulan aktivitas yang berbeda yang menjamin bahwa perangkat lunak yang dibangun dapat ditelusuri sesuai dengan kebutuhan pelanggan. Dengan kata lain sebagai berikut [5]:

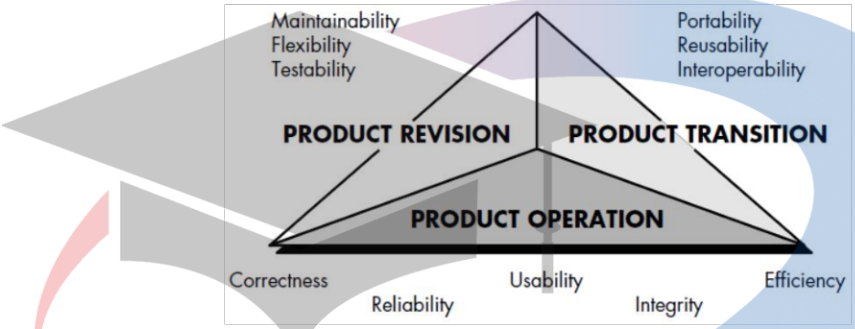
1. Verifikasi: “Apakah produk dibangun dengan benar?” (lebih ke arah apakah proses produk sudah benar dan telah berhasil mengimplementasikan fungsi dengan benar).
2. Validasi: “Apakah sudah membangun produk yang benar?” (lebih ke arah hasil produk sudah sesuai dengan yang diinginkan).

2.4 Metode McCall

McCall, Richards, dan Walters mengusulkan suatu penggolongan yang bermanfaat untuk faktor-faktor yang menentukan kualitas suatu perangkat lunak

yang sedang akan dikembangkan seperti yang terlihat pada gambar berikut ini. Tiga aspek yang penting dari suatu produk perangkat lunak meliputi karakteristik-karakteristik operasionalnya (*product operation*), kemampuannya untuk segera berubah (*product transition*), dan kemampuannya untuk beradaptasi terhadap lingkungan yang baru (*product revision*) [6].

Gambar berikut ini menunjukkan tiga aspek penting faktor-faktor kualitas perangkat lunak menurut McCall [1].



Gambar 2.2 Teori Kualitas McCall

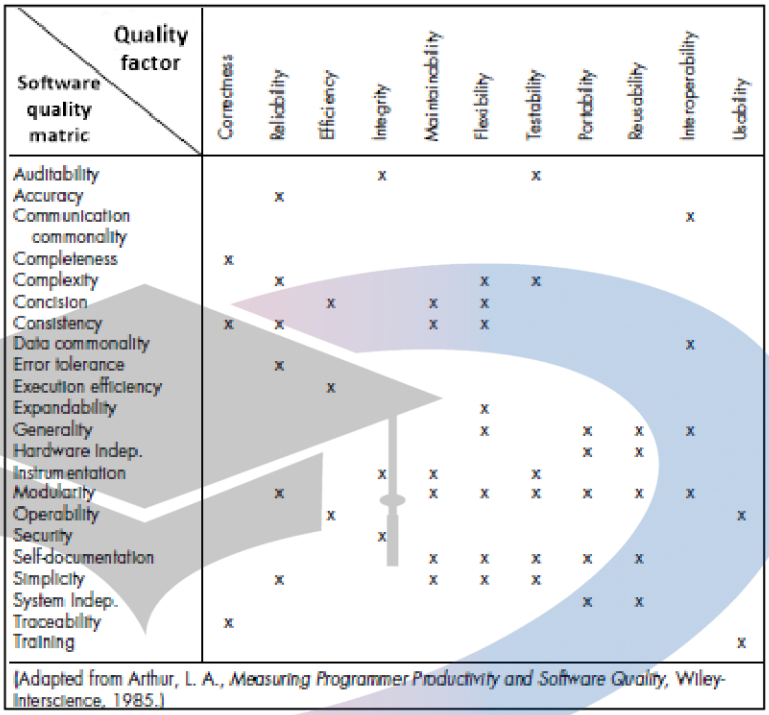
Product Revision meliputi beberapa faktor yaitu *maintainability*, *flexibility*, dan *testability.* Kemudahan perawatan (*maintainability*) adalah usaha yang dibutuhkan untuk menemukan dan memperbaiki *error* yang terjadi dalam suatu perangkat lunak. Fleksibilitas (*flexibility*) adalah usaha yang dibutuhkan/kemudahan untuk memodifikasi suatu perangkat lunak. Testabilitas (*testability*) adalah usaha yang dibutuhkan untuk menguji suatu perangkat lunak dengan tujuan memastikan perangkat lunak berjalan sesuai dengan fungsi yang diharapkan [6].

Product Transition meliputi beberapa faktor yaitu *portability*, *reusability*, dan *interoperability*. Portabilitas (*portability*) adalah usaha yang dibutuhkan jika perangkat lunak dipindahkan/ ditransfer ke perangkat lunak ataupun perangkat keras yang lain. Reusabilitas (*reusability*) adalah kemampuan suatu perangkat lunak ataupun bagian perangkat lunak dapat digunakan kembali untuk dikembangkan di perangkat lunak lainnya – berhubungan dengan pengelompokan/pembuatan paket sistem (*packaging*) dan ruang lingkup dari fungsi yang dijalankan oleh perangkat lunak. Interoperabilitas (*interoperability*) adalah kemampuan perangkat lunak untuk dapat berinteraksi dengan perangkat lunak lainnya [6].

Product Operation meliputi beberapa faktor yaitu *correctness*, *reliability*, *usability*, *integrity*, dan *efficiency*. Kebenaran (*correctness*) adalah bagaimana program akan memberikan hasil sesuai dengan spesifikasi yang sudah ditetapkan sebelumnya dan memenuhi sasaran-sasaran pelanggan. Kehandalan (*reliability*) adalah bagaimana suatu program diharapkan dapat melakukan fungsi-fungsi tertentu sesuai dengan tingkat ketelitian yang diinginkan. Efisiensi (*efficiency*) adalah jumlah sumber daya komputasi dan kode yang diperlukan program untuk mampu melaksanakan fungsinya secara baik dan benar. Integritas (*integrity*) adalah bagaimana akses ke data dan perangkat lunak oleh orang-orang yang tidak berkepentingan dapat dikendalikan. Penggunaan (*usability*) adalah besarnya usaha yang diperlukan untuk mempelajari, mengoperasikan, menyediakan masukan (*input*), dan menafsirkan keluaran (*output*) suatu program [6].

Hubungan antara faktor kualitas dan metrik kualitas perangkat lunak ditunjukkan pada gambar 2.3 [1]. Ada beberapa metrik kualitas perangkat lunak yang juga digunakan sebagai metrik kualitas dari faktor kualitas yang lain, di antaranya metrik kualitas *audability* merupakan metrik untuk faktor kualitas *integrity* dan *testability*. Metrik kualitas *complexity* merupakan metrik untuk faktor kualitas *reliability*, *flexibility*, dan juga *testability*. Metrik kualitas *concision* merupakan metrik untuk faktor kualitas *efficiency*, *maintainability*, dan *flexibility*. Metrik kualitas *consistency* merupakan metrik untuk faktor kualitas *correctness*, *reliability*, *maintainability*, dan *flexibility*. Metrik kualitas *generality* merupakan metrik untuk faktor kualitas *flexibility*, *portability*, *reusability*, dan juga *interoperability*. Metrik kualitas *hardware independence* merupakan metrik untuk faktor kualitas *portability* dan *reusability*. Metrik kualitas *instrumentation* merupakan metrik untuk faktor kualitas *integrity*, *maintainability*, dan *testability*. Metrik kualitas *modularity* merupakan metrik untuk faktor kualitas *reliability*, *maintainability*, *flexibility*, *testability*, *portability*, *reusability*, dan juga *interoperability*. Metrik kualitas *operability* merupakan metrik untuk faktor kualitas *integrity* dan *usability*. Metrik kualitas *self documentation* merupakan metrik untuk faktor kualitas *maintainability*, *flexibility*, *testability*, *portability*, dan juga *reusability*. Metrik kualitas *simplicity* merupakan metrik untuk faktor kualitas *reliability*, *maintainability*, *flexibility*, dan

testability. Metrik kualitas *system independence* merupakan metrik untuk faktor kualitas *portability* dan juga *reusability*.



Software quality matric \ Quality factor	Correctness	Reliability	Efficiency	Integrity	Maintainability	Flexibility	Testability	Portability	Reusability	Interoperability	Usability
Auditability				x			x				
Accuracy		x									
Communication commonality										x	
Completeness	x										
Complexity		x				x	x				
Concision			x		x	x					
Consistency	x	x			x	x					
Data commonality										x	
Error tolerance		x									
Execution efficiency			x								
Expandability						x					
Generality						x		x	x	x	
Hardware indep.								x	x		
Instrumentation				x	x		x				
Modularity		x			x	x	x	x	x	x	
Operability			x								x
Security				x							
Self-documentation					x	x	x	x	x		
Simplicity		x			x	x	x				
System indep.								x	x		
Traceability	x										
Training											x

(Adapted from Arthur, L. A., *Measuring Programmer Productivity and Software Quality*, Wiley-Interscience, 1985.)

Gambar 2.3 Hubungan Antara Faktor Kualitas dan Metrik Kualitas Perangkat Lunak

Adapun metrik yang dipakai untuk skema pengukuran sesuai faktor-faktor kualitas perangkat lunak di atas adalah sebagai berikut [1]:

1. *Auditability* adalah kemudahan untuk memeriksa apakah *software* memenuhi standar atau tidak.
2. *Accuracy* adalah ketelitian dari komputasi dan kontrol.
3. *Communication Commonality* adalah sejauh mana *interface*, protokol, dan *bandwidth* digunakan.
4. *Completeness* adalah sejauh mana implementasi penuh dari fungsi-fungsi yang diperlukan telah tercapai.
5. *Conciseness* adalah keringkasan program dalam ukuran *line of commands* (LOC).
6. *Consistency* adalah derajat penggunaan teknik-teknik desain dan dokumentasi yang seragam pada seluruh proyek pengembangan *software*.
7. *Data Commonality* adalah derajat penggunaan tipe dan struktur data baku pada seluruh program.

8. *Error Tolerance* adalah kerusakan yang terjadi apabila program mengalami *error*.
9. *Execution Efficiency* adalah kinerja *run-time* dari program.
10. *Expandability* adalah sejauh mana desain prosedur, data, atau arsitektur dapat diperluas.
11. *Generality* adalah luasnya kemungkinan aplikasi dari komponen-komponen program.
12. *Hardware Independence* adalah sejauh mana *software* tidak bergantung pada kekhususan dari *hardware* tempat *software* itu beroperasi.
13. *Instrumentation* adalah sejauh mana program memonitor operasi dirinya sendiri dan mengidentifikasi *error* yang terjadi.
14. *Modularity* adalah *functional independence* dari komponen-komponen program.
15. *Operability* adalah kemudahan mengoperasikan program.
16. *Security* adalah ketersediaan mekanisme untuk mengontrol dan melindungi program dan data terhadap akses dari pihak yang tidak berhak.
17. *Self-Documentation* adalah sejauh mana *source-code* memberikan dokumentasi yang berarti.
18. *Simplicity* adalah kemudahan suatu program untuk dimengerti.
19. *Traceability* adalah kemudahan merujuk balik implementasi atau komponen program ke kebutuhan pengguna *software*.

Untuk menghitung kualitas dengan menggunakan metode McCall dapat menggunakan persamaan berikut ini [8].

$$Fq = c1 * m1 + c2 * m2 + c3 * m3 + \ldots + cn * mn \quad (2.1)$$

Dimana:

Fq = Faktor *software quality*

c = Bobot yang bergantung pada produk dan kepentingan

m = Metrik yang mempengaruhi faktor *software quality*

Tahapan yang ditempuh dalam melakukan pengukuran adalah sebagai berikut [8]:

1. Tentukan kriteria yang digunakan untuk mengukur suatu faktor
2. Tentukan bobot dari setiap kriteria

3. Tentukan skala dari nilai kriteria
4. Berikan nilai pada setiap kriteria
5. Hitung nilai total dengan rumus pada persamaan (2.1)
6. Ubah nilai faktor kualitas dalam bentuk persentase (%) dengan menggunakan persamaan berikut.

$$Persentase = \frac{Nilai\ yang\ didapat}{Nilai\ maksimum}\ x\ 100\% \quad (2.2)$$

Berikut ini adalah contoh perhitungan kualitas perangkat lunak dengan menggunakan metode McCall [8].

Tabel 2.2 Contoh Perhitungan Kualitas Perangkat Lunak

No.	Faktor	Bobot	Kriteria	Bobot	Nilai
1	Ketepatan (*Correctness*)	0,3	Kelengkapan informasi yang disajikan sistem	0,3	8,3
			Kesesuaian informasi yang disajikan sistem dengan kebutuhan informasi di dalam sistem pakar deteksi kerusakan mesin sepeda motor *nonmatic*	0,3	8
			Kemampuan sistem dalam menelusuri kesalahan informasi ataupun kesalahan *input* data	0,2	8
			Kesesuaian informasi keadaan fasilitas yang disajikan dalam sistem dengan keadaan sebenarnya dalam sistem pakar deteksi kerusakan mesin sepeda motor *nonmatic*	0,2	8
2	Keandalan (*Reliability*)	0,2	Kemampuan sistem dalam mencegah terjadinya kesalahan *input* data	0,4	7,8
			Konsistensi sistem dalam proses penyimpanan data	0,3	8,2
			Konsistensi sistem dalam menyimpan data	0,3	8,4

Tabel 2.2 Contoh Perhitungan Kualitas Perangkat Lunak (Lanjutan)

No.	Faktor	Bobot	Kriteria	Bobot	Nilai

3	Efisiensi (*Efficiency*)	0,2	Efisiensi waktu yang dibutuhkan sistem dalam memroses data dan menyajikan informasi	0,3	8,3
			Kecepatan sistem dalam memroses penyimpanan data	0,2	8,5
			Bahasa dan informasi dalam sistem dapat dipahami dengan cepat	0,2	8,3
			Sistem tidak membutuhkan spesifikasi *hardware* yang tinggi	0,3	8,2
4	Kegunaan (*Usability*)	0,2	Bahasa dan informasi dalam sistem mudah dimengerti oleh *user* (*user friendly*)	0,4	8,5
			User dapat dengan mudah mengoperasikan sistem	0,4	8,2
			Tidak membutuhkan waktu yang lama untuk dapat mempelajari dan mengoperasikan sistem	0,2	8,2
5	Pemeliharaan (*Maintainability*)	0,1	Kelengkapan penyajian modul program atau pembagian menu	0,3	7,9
			Ketersediaan petunjuk penggunaan dan pengoperasian sistem di dalam sistem	0,3	8
			Ketersediaan dokumentasi sistem	0,2	7,9
			Ketersediaan pesan kesalahan dan petunjuk dalam mengatasi masalah sistem	0,2	8,3

Correctness = w1n1 + w2n2 + w3n3 + w4n4

= (0,3*8,3) + (0,3*8,0) + (0,2*8,0) + (0,2*8,0)

= 2,49 + 2,40 + 1,60 + 1,60

= 8,09

Reliability = w1n1 + w2n2 + w3n3

= (0,4*7,8) + (0,3*8,2) + (0,3*8,4)

= 3,12 + 2,46 + 2,52

= 8,10

Efficiency = w1n1 + w2n2 + w3n3 + w4n4
= (0,3*8,3) + (0,2*8,5) + (0,2*8,3) + (0,3*8,2)
= 2,49 + 1,70 + 1,66 + 2,46
= 8,31

Usability = w1n1 + w2n2 + w3n3
= (0,4*8,5) + (0,4*8,2) + (0,2*8,2)
= 3.40 + 3,28 + 1,64
= 8,32

Maintainability = w1n1 + w2n2 + w3n3 + w4n4
= (0,3*7,9) + (0,3*8,0) + (0,2*7,9) + (0,2*8,3)
= 2,37 + 2,40 + 1,58 + 1,66
= 8,01

Sehingga total kualitas (∑) yang diperoleh adalah sebagai berikut:

∑ = (3*8,09) + (2*8,10) + (2*8,31) + (2*8,32) + (1*8,01)
= 24,27 + 16,2 + 16,62 + 16,64 + 8,01
= 81,74 / 100 * 100%
= 81,74%