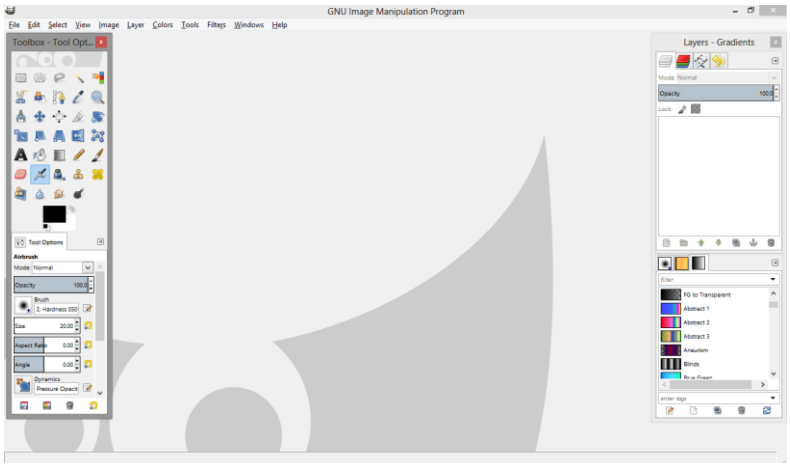
BAB 2
LANDASAN TEORI

2.1. Perangkat Lunak Pengolahan Citra (Digital)

Perangkat lunak pengolahan citra, dalam hal ini yang dimaksud dengan citra (*image*) adalah sebuah gambar yang terdapat pada bidang dua dimensi, jadi sebuah perangkat lunak pengolahan citra itu memiliki arti sebagai sebuah perangkat lunak yang digunakan untuk melakukan pengolahan gambar (dua dimensi) khususnya dengan menggunakan sebuah aplikasi komputer, yang bertujuan untuk memperbaiki kualitasnya agar menjadi lebih baik, dengan cara dimanipulasi. Beberapa contoh perangkat lunak yang bisa digunakan untuk melakukan pengolahan citra sebagai berikut:

2.1.1. GIMP 2

GIMP 2 merupakan sebuah perangkat lunak yang menjadi salah satu proyek GNU. GIMP pertama kali d kembangkan pada tahun 1995 oleh dua orang mahasiswa yaitu, Spencer Kimball dan Peter Mattis. GIMP sendiri merupakan sebuah perangkat alternatif pengganti Photoshop tertua dan memiliki fitur-fitur dan karakteristik yang hampir mendekati Photoshop. GIMP bisa digunakan untuk melakukan *retouching gambar* (*edit)*, menggambar, mengubah ukuran ataupun format dari sebuah gambar, *cropping* dan tugas-tugas khusus lainnya (Gambar 2.1).

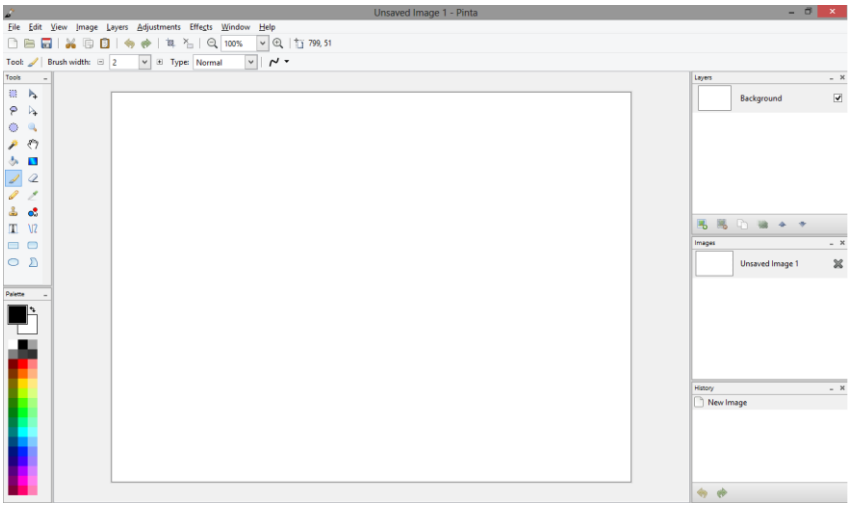


Gambar 2.1 Antarmuka Perangkat Lunak GIMP2

Tools yang digunakan untuk melakukan editing gambar dapat diakses melalui toolbox, melalui menu dan dialog windows. Fitur tersebut termasuk filter dan kuas, serta alat-alat transformasi, seleksi, lapisan dan masking. GIMP mendukung mengimpor dan mengekspor dengan jumlah besar dengan format file yang berbeda, seperti BMP, JPEG, PNG, GIF, TIFF, PSD, dll, sedangkan untuk format asli dari GIMP disebut dengan XCF, yang dirancang untuk menyimpan semua informasi dari GIMP, XCF adalah sebuah akronim dari *eXperimental Computing Facility*, yang merupakan sebuah fasilitas tempat GIMP dikembangkan.

2.1.2. Pinta

Pinta merupakan sebuah perangkat lunak *open-source* yang dibuat berdasarkan inspirasi Paint.NET, sebuah *program* yang hampir mirip dengan sebuah program *editing* gambar Paint dari Microsoft Windows. Pinta sendiri memiliki keterbatasan hanya bisa membuka *file* gambar seperti, BMP, JPEG, PNG, GIF, TIFF berbeda dengan GIMP yang memiliki fitur yang bisa membuka file berformat PSD. Keunggulan dari perangkat lunak ini adalah memiliki sebuah fitur *undo* yang tidak terbatas. Pinta mulai dikembangkan oleh Jonathan Pobst pada tahun 2010. Pinta memiliki beberapa fitur yang dimiliki oleh GIMP, meskipun dengan fungsional yang sedikit berbeda, namun dari segi pemakaian *resource*/sumber daya, pinta jauh lebih ringan dan lebih mudah digunakan dibandingkan dengan GIMP (Gambar 2.2).



Gambar 2.2 Antarmuka Perangkat Lunak Pinta

2.2. Software Quality

Definisi pertama dari *software quality* yang disarankan oleh IEEE (IEEE, 1991) menyebutkan bahwa *software quality* adalah [2] :

1. Tingkatan dari sebuah sistem, komponen atau proses yang ditentukan untuk memenuhi persyaratan yang telah ditentukan.
2. Tingkatan dari sebuah sistem, komponen atau proses sesuai dengan konsumen ataupun kebutuhan maupun harapan dari pengguna.

Dari dua penjelasan zzztersebut menjabarkan dua alternatif dari definisi *software quality* yang sebelumnya dikemukankan oleh penggagas *modern quality assurance*, Philip B. Crosby and Joseph M. Juran. Masing-masing dari definisi tersebut merefleksikan konsepsi yang berbeda dari *software quality* yaitu [2]:

1. "Kualitas berarti kesesuaian dari kebutuhan" (Crosby, 1979).
2. "a. Kualitas terdiri dari fitur-fitur yang memenuhi kebutuhan pengguna, dengan demikian dapat memberikan kepuasan produk.
   b. Kualitas itu terbebas dari kekurangan/kesalahan" (Juran, 1988).

Definisi software quality dari Crosby mengacu pada tingkatan dimana perangkat lunak memenuhui spesifikasi yang disiapkan oleh pengguna dan tim professionalnya, ini berarti bahwa error yang terdapat pada spesifikasi perangkat lunak tidak dianggap dan tidak mengurangi kualitas dari perangkat lunak.

Definisi dari Juran ditujukan untuk mencapai kepuasan dari pengguna, melihat dari pemenuhan kebutuhan yang benar-benar dibutuhkan sebagai tujuan sesungguhnya dari *software quality*. Mereferensi dari definisi yang kedua, para pengembang dituntut untuk melakukan penelitian dan pengkoreksian semaksimal mungkin, dan jika perlu para pengembang perangkat lunak melakukan penelitian dan pengkoreksian dari kebutuhan yang diperlukan oleh pengguna. Namun terdapat kekurangan dari definisi tersebut, yaitu membebaskan pengguna dari tanggung jawab dalam memenuhi kebutuhan dari spesifikasi perangkat lunak secara tepat, dan juga memperbolehkan pengguna melakukan penambahan tau pengurangan fungsionalitas yang mungkin berbeda dari spesifikasi yang telah ditentukan, pada tahap pertengahan ataupun pada tahap akhir dari proyek tersebut. Akibatnya timbul

masalah pada saat pembangunan proyek, terutama pada saat pengujian seberapa baik program tersebut bisa memenuhi kebutuhan pengguna.

Software quality menurut Pressman didefinisikan sebagai kesesuaian tuntutan fungsional dan performa yang dinyatakan secara eksplisit disebutkan pada standar dokumentasi pengembangan dan ciri implisit yang diharapkan dari pengembangan perangkat lunak secara profesional. Pressman menyarankan tiga hal dalam pengukuran kualitas yang harus dipenuhi oleh para pengembang, yaitu:

1. Secara spesifik menjelaskan kebutuhan fungsional, yang mengacu pada fungsi utama dari sistem perangkat lunak.
2. Menentukan standar kualitas dari perangkat lunak di dalam kontrak.
3. *Good Software Engineering Practices* (GSEP), merefleksikan seni dari tahapan – tahapan pembangunan perangkat lunak, yang harus dipenuhi oleh pengembang meskipun tidak disebutkan secara eksplisit di dalam kontak.

Dari tiga hal tersebut dapat ditentukan syarat-syarat yang menjadi acuan tepenuhinya operasi pengujian terhadap perangkat lunak [2].

2.3. Software Quality Model

Software quality model merupakan sebuah model acuan yang digunakan dalam melakukan penilaian kualitas dari sebuah perangkat lunak, yang bertujuan untuk melakukan peningkatan kualitas dari sebuah perangkat lunak dan membuatnya dapat diukur.

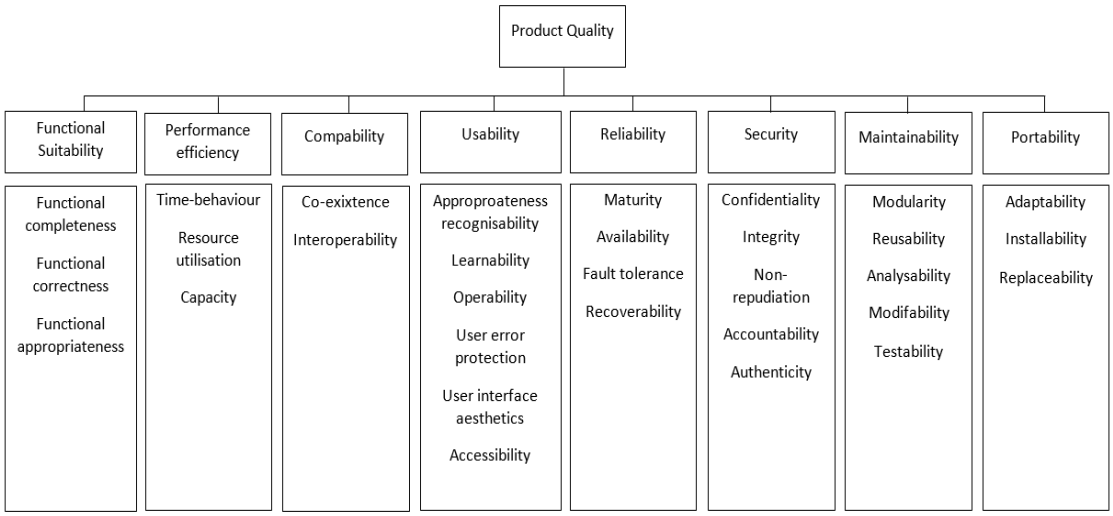
2.3.1. ISO/IEC 25010

Model ISO/IEC 25010 merupakan hasil pengembagan dari ISO/IEC 9126 yang pertama kali di publikasikan pada tahun 1991. ISO/IEC 9126 ini kemudian direvisi kembali pada tahun 2001 menjadi ISO/IEC 9126-1, yang kemudian mengalami perubahan kembali menjadi ISO/IEC 25000 pada tahun 2005 yang disebut juga sebagai SquaRE (*System and software Quality Requirement and Evaluation*) [3]. Dari ISO/IEC 25000 kemudian dikembangkan kembali menjadi ISO/IEC 25010. Model ISO/IEC 25010 ini dapat digunakan untuk melakukan

evaluasi kualitas perangkat lunak berdasarkan dua dimensi umum yaitu *product quality model* dan *quality in use model* [3].

1. *Product Quality Model*

Pada dimensi *product quality model*, proses yang dilakukan mengacu pada karakteristik intrinsik dari sebuah perangkat lunak, yang tersusun dari delapan buah karakteristik, yaitu *functional suitability, reliability, operability, performance efficiency, security, compatibility, maintainability* dan *transferability*. Masing-masing karakteristik terdiri dari sekumpulan sub-karakteristik dan penerapan model ini hanya digunakan untuk produk perangkat lunak ataupun sistem (Gambar 2.3).



Gambar 2.3 Product Quality Model [3]

Dari masing-masing karakteristik memiliki beberapa sub-karakteristik, penjelasan dari karakteristik dan sub-karakteristik dari dimensi faktor kualitas *pruduct quality model* adalah sebagai berikut:

1. *Functional suitability*

   Functional suitability adalah sejauh mana suatu produk atau sistem menyediakan fungsi yang memenuhi kebutuhan yang secara jelas disebutkan ataupun secara tersirat ketika digunakan dalam kondisi tertentu. *Functional suitability* ini terbagi menjadi tiga sub-karakteristik sebegai berikut:

a. *Functional completeness*

Sub-karakteristik ini membahas mengenai sejauh mana set fungsi bisa menangani tugas yang telah ditentukan dan memenuhi tujuan pengguna.

b. *Functional correctness*

Functional correctness adalah sejauh mana produk atau sistem dapat memberikan hasil yang benar secara akurat/presisi.

c. *Functional appropriateness*

Functional appropriatenes adalah sejauh mana fungsi yang ada dalam perangkat lunak bisa memfasilitasi pemenuhan tugas dan tujuan yang ingin dicapai.

2. *Performance efficiency*

Performance efficiency menjelaskan mengenai kinerja yang berhubungan dengan jumlah pemakaian sumber daya dalam kondisi tertentu. *Performance efficiency* ini dibagi kedalam tiga sub-karakteristik berikut:

a. *Time behaviour*

Time behaviour adalah kemampuan perangkat lunak dalam memberikan respon dan waktu pengolahan yang sesuai saat melakukan fungsinya.

b. *Resource utilization*

Resource utilization adalah jumlah dan jenis sumber daya (*resources)* yang digunakan oleh sebuah produk atau sistem. Ketika melakukan fungsinya dengan tujuan untuk memenuhi persyaratan.

c. *Capacity*

Capacity adalah batas maksimum sebuah parameter produk atau sistem ketika memenuhi persyaratan/kebutuhannya.

3. *Compatibility*

Compability adalah sejauh mana produk, sistem atau komponen dapat bertukar informasi dengan produk lain, sistem atau komponen, dan

melakukan fungsi yang diperlukan, ketika berbagi perangkat keras atau perangkat lunak yang sama. *Compability* ini dibagi menjadi dua sub-karakteristik yaitu:

a. *Co-existence*

   Co-existence yaitu kemampuan sebuah produk dapat melakukan fungsinya secara efisien ketika melakukan *sharing* data atau sumber daya dengan produk yang lainnya, tanpa menimbulkan dampak yang merugikan pada produk lain.

b. *Interoperability*

   Interoperability adalah sejauh mana dua atau lebih sistem, produk, atau komponen dapat melakukan pertukaran data/informasi dan menggunakan data/informasi dari hasil pertukaran data/informasi.

4. *Usability*

   Usability adalah sejauh mana suatu produk atau sistem dapat digunakan oleh pengguna untuk mencapai tujuan tertentu dengan efektif, efisien dan mencapai kepuasan penggunaan dalam konteks tertentu. *Usability* terbagi kedalam enam sub-karakteristik yaitu:

   a. *Appropriateness recognizability*

      Sub-karakteristik ini menjelaskan mengenai sejauh mana pengguna dapat mengenali, apakah suatu produk atau sistem yang sesuai untuk kebutuhan mereka.

   b. *Learnability*

      Learnability adalah sejauh mana produk atau sistem dapat digunakan pengguna untuk mencapai tujuan dari belajar menggunakan produk atau sistem secara efisien, efektif, terbebas dari resiko dan mencapai kepuasan penggunaan dalam konteks tertentu.

   c. *Operability*

      Operability membahas mengenai kemampuan suatu produk atau sistem memiliki atribut yang membuatnya mudah dioperasikan dan kontrol

d. *User error protection*

User error protection membahas mengenai sistem dari perangkat lunak yang melindungi pengguna membuat kesalahan/*error*.

e. *User interface aesthetics*

User interface aesthetics membahas mengenai *user interface* yang memungkinkan sebuah interaksi yang menyenangkan dan memberikan kepuasan bagi pengguna.

f. *Accessibility*

Accessibility adalah sejauh mana suatu produk atau sistem dapat digunakan oleh semua kalangan, *accessibility* juga merupakan sebuah kemampuan dari perangkat lunak untuk mencapai tujuan tertentu dalam penggunaannya.

5. *Reliability*

Reliability membahas mengenai seberapa lama perangkat lunak dapat digunakan dalam menjalankan fungsi-fungsinya pada jangka waktu yang telah ditentukan. Faktor yang akan menjadi tolak ukur dalam melakukan penilaian pada domain perangkat lunak pengolahan citra adalah sejauh mana perangkat lunak dapat diandalkan pada saat pengguna menggunakan berbagai fungsionalitas yang terdapat pada perangkat lunak, pada saat melakukan pekerjaannya. *Reliability* dibagi kedalam empat sub-karakteristik berikut:

a. *Maturity*

Maturity adalah keandalan sebuah produk atau komponen ketika memenuhi sebuah kebutuhan pada operasi normal.

b. *Availability*

Availability adalah ketersediaan produk atau komponen operasional dan dapat diakses ketika akan digunakan.

c. *Fault tolerance*

Fault tolerance adalah kemampuan dari produk atau komponen dalam mengatasi kesalahan yang terjadi, baik itu terjadi pada perangkat keras maupun perangkat lunak.

d. *Recoverability*

Recoverability membahas mengenai kemampuan dari produk atau sistem ketika terjadi kegagalan proses dan dapat secara langsung memulihkan data yang rusak ke kondisi awal sebelum terjadi kerusakan sistem.

6. *Security*

Security adalah sejauh mana produk atau sistem dapat melindungi informasi atau data dari pihak-pihak tertentu sesuai dengan otorisasi/ hak akses yang dimiliki dalam melakukan modifikasi data. *Security* ini terbagi menjadi lima sub-karakteristik yaitu:

a. *Confidentiality*

Confidentiality adalah kewenangan suatu produk atau sistem dalam memastikan pengaksesan data hanya bisa dilakukan oleh mereka yang berwenang dan memiliki akses.

b. *Integrity*

Integrity adalah kemampuan dari produk atau sistem dalam mencegah pengaksesan pihak yang tidak berwenang, yang ingin melakukan modifikasi data ataupun program komputer.

c. *Non-repudiation*

Non-repudiation adalah kemampuan dari perangkat lunak untuk melakukan pembuktian dari sebuah aksi, sehingga tidak akan terjadi penyangkalan dari aksi yang telah dilakukan.

d. *Accountability*

Accountability membahas mengenai sejauh mana tindakan dari satu entitas dapat ditelusuri.

e. *Authenticity*

Authenticity membahas mengenai sejauh mana perangkat lunak dapat mengidentifikasi subjek atau sumber daya bisa dibuktikan dimiliki oleh seseorang yang berhak.

7. *Maintability*

   Maintability adalah tingkat efektivitas dan efisiensi dari sebuah produk atau sistem dapat dimodifikasi atau diperbaharui oleh *maintainers.* *Maintability* dibagi kedalam lima sub-karakteristik berikut:

   a. *Modularity*

      Modularity adalah kemampuan sistem atau program komputer yang terdiri dari komponen diskrit, bahwa perubahan ke satu komponen memiliki dampak minimal pada komponen lainnya.

   b. *Reusability*

      Reusability adalah kemampuan dari sebuah aset dari perangkat lunak dapat digunakan kembali pada sistem yang lain ataupun ketika melakukan pembangunan aset yang lainnya.

   c. *Analysability*

      Analysability adalah tingkat efektifitas dan efisiensi untuk menilai dampak pada produk atau sistem dari perubahan yang dimaksudkan pada satu bagian atau lebih, atau untuk mendiagnosis produk dari kekurangan atau penyebab kegagalan, atau untuk mengidentifikasi bagian akan diubah.

   d. *Modifiability*

      Modifiablility adalah produk atau sistem dapat secara efektif dan efisien diubah tanpa memperkenalkan cacat atau merendahkan kualitas produk yang sudah ada.

   e. *Testability*

      Testability menjelaskan mengenai tingkat efektifitas dan efisiensi dalam pengujian kriteria ketika membangun sebuah sistem, produk atau komponen dan tes dapat dilakukan untuk menentukan apakah kriteria tersebut telah terpenuhi.

8. *Portability*

   Portability adalah tingkat efektifitas dan efisiensi dari sebuah sistem, produk, atau komponen yang dapat ditransfer dari satu perangkat keras, perangkat lunak atau cara kerja sebuah sistem kedalam sistem yang

lainnya. Pada bagian ini yang akan menjadi faktor penilaian adalah seberapa banyak perangkat lunak bisa dioperasikan pada lebih dari satu sistem operasi. *Portability* terbagi kedalam tiga sub-karakteristik yaitu:

a. *Adaptability*

   Adaptability adalah produk atau sistem dapat secara efektif dan efisien disesuaikan untuk perangkat keras, perangkat lunak atau lingkungan operasi yang berbeda.

b. *Installability*

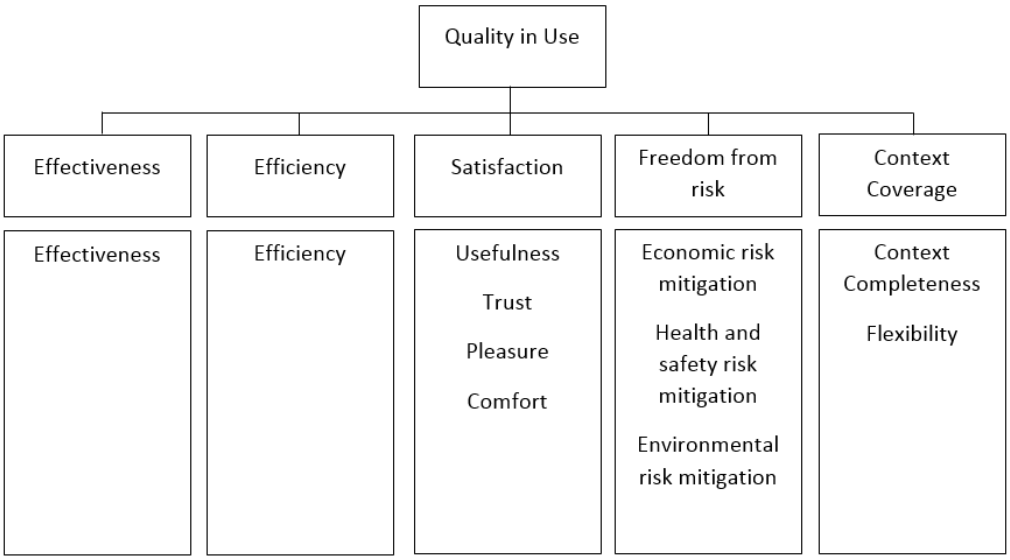
   Installability adalah tingkat efektifitas dan efisiensi dari produk atau sistem yang berhasil diinstal atau dihapus dalam lingkungan tertentu.

c. *Replaceability*

   Replaceability artinya produk atau sistem dapat menggantikan perangkat lunak lainnya dengan fungsi dan tujuan yang sama.

2. *Quality In Use Model*

Dimensi *quality in use model* merupakan tingkatan di mana sebuah produk atau sistem yang digunakan oleh pengguna tertentu dalam menjalankan kebutuhannya untuk mencapai taget dengan mengacu pada karakteristik *effectiveness*, *efficiency*, *freedom from risk*, *context coverage* serta *satisfication.* Tiap-tiap karakteristik dapat ditempatkan pada masing-masing aktivitas yang berbeda sesuaikan dengan para *stakeholder* (Gambar 2.4).



Gambar 2.4 Quality in Use Model [3]

Dari masing-masing karakteristik memiliki beberapa sub-karakteristik, penjelasan dari karakteristik dan sub-karakteristik dari dimensi faktor kualitas *quality in use model* adalah sebagai berikut [3]:

1. *Effectiveness*

   Faktor ini melihat dari akurasi dan kelengkapan agar pengguna bisa mencapai tujuan yang diharapkan. Pada faktor ini, pengukuran akan dilakukan berdasarkan fungsionalitas dari domain perangkat lunak pengolahan citra

2. *Efficiency*

   Efficiency berkaitan dengan sumber daya yang digunakan berhubungan dengan akurasi dan kelengkapan agar pengguna bisa mencapai tujuan. Pemakaian jumlah memori/RAM akan menjadi faktor penilaian bagi domain perangkat lunak pengolahan citra yang digunakan pada penelitian ini.

3. *Satisfaction*

   Faktor ini melihat sejauh mana pengguna puas dengan produk atau sistem yang digunakan dalam penggunaan konteks tertentu. Faktor ini memiliki empat sub-karakteristik sebagai berikut:

a. *Usefulness*

Sub-karakteristik ini menbahas mengenai kepuasan yang dirasakan oleh pengguna ketika menggunakan perangkat lunak untuk mencapai tujuan pragmatis, termasuk hasil penggunaan dan dampak yang meraka rasakan. Kemudahan dan kepuasan dari penggunaan perangkat lunak, akan menjadi faktor penilaian terhadap domain perangkat lunak pengolahan citra.

b. *Trust*

Trust adalah kepercayaan dari pengguna yang memiliki keyakinan bahwa suatu produk atau sistem akan berfungsi sebagaimana mestinya.

c. *Pleasure*

Pleasure adalah sejauh mana pengguna memperoleh kesenangan dari pemenuhan tujuan pribadi mereka dalam menggunakan perangkat lunak.

d. *Comfort*

Comfort membahas mengenai kepuasan pengguna yang berhubungan dengan kenyamanan secara fisik ketika menggunakan perangkat lunak.

4. *Freedom from risk*

Freedom from risk adalah sejauh mana suatu produk atau sistem meringankan risiko potensial terhadap status ekonomi, kehidupan manusia, kesehatan, atau lingkungan. Faktor ini memiliki tiga sub-karakteristik sebagai berikut:

a. *Economic risk mitigation*

Sub-karakteristik ini membahas mengenai sejauh mana suatu produk atau sistem meringankan risiko potensial terhadap status keuangan, operasi yang efisien, properti komersial, reputasi atau sumber daya lainnya dalam konteks penggunaan yang dimaksudkan.

b. *Health and safety risk mitigation*

Health and safety risk mitigation adalah sejauh mana suatu produk atau sistem meringankan risiko potensial terhadap pengguna dalam konteks penggunaan yang dimaksudkan.

c. *Environmental risk mitigation*

Environmental risk mitigation adalah tingkatan dari suatu produk atau sistem meringankan risiko potensial terhadap properti atau lingkungan dalam konteks penggunaan yang dimaksudkan.

5. *Context Coverage*

Context Coverage adalah sejauh mana suatu produk atau sistem dapat digunakan dengan efektif, efisien, terbebas dari resiko dan kepuasan baik penggunaan dalam konteks tertenu dan dalam konteks di luar yang awalnya diidentifikasi secara eksplisit. *Context coverage* memiliki dua sub-kategori berikut:

a. *Context completeness*

Context completeness membahas mengenai suatu produk atau sistem dapat digunakan dengan efektif, efisien, terbebas dari resiko dan kepuasan dalam penggunaan pada konteks tertentu.

b. *Flexibility*

sejauh mana suatu produk atau sistem dapat digunakan dengan efektif, efisien, terbebas dari resiko dan kepuasan dalam konteks yang melampaui orang-orang, yang awalnya ditentukan dalam persyaratan.

2.4. Metode Pengukuran Perangkat Lunak (Software Metrics)

Software metrics didefinisikan sebagai satuan untuk melakukan pengukuran suatu atribut perangkat lunak. Definisi formal dari *metrics* itu sendiri diartikan sebagai teori himpunan dan matematika. Proses perhitungan *metrics* ini dilakukan secara kuantitatif dengan tujuan untuk melakukan pengembangan perangkat lunak. *Software metrics* bisa digunakan untuk mengetahui nilai/bobot dari suatu perangkat lunak, sehingga dapat dilakukan perbandingan antar perangkat lunak yang sama

maupun berbeda. Untuk melakukan penilaian perangkat lunak pada penelitian ini, akan digunakan *Goal Question Metrics. Goal Question Metrics* merupakan sebuah pendekatan *sytemantic* yang digunakan untuk merangkai dan mengintegrasikan tujuan dari model proses perangkat lunak, produk ataupun sebuah organisasi. Metode ini mengasumsikan bahwa pengukuran harus dilakukan secara terarah dengan menentukan tujuan yang jelas, secara operasional dan akhirnya akan membentuk sebuah kerangkar kerja dalam hal menafsirkan data dengan tujuan yang telah ditetapkan [4]. Hasil dari penggunaan pendekatan GQM adalah untuk menargetkan masalah tertentu dan seperangkat aturan yang akan digunakan untuk interpretasi pengukuran data. Model pengukuran yang dihasilkan GQM terbagi kedalam tiga tingkatan yaitu:

1. *Conceptual level (goal)*

   Conceptual level didefinisikan untuk sebuah objek, alasan yang bervariatif, yang berhubungan dengan berbagai model kualitas, dari berbagai sudut pandang, dan relatif terhadap lingkungan tertentu. Adapun yang menjadi taget dari pengukuran yaitu:

   a. *Products* seperti spesifikasi, desain, program.
   b. *Processes* seperti *specifying,* perancangan, pengujia, *interviewing.*
   c. *Resources* seperti personel, perangkat lunak, perangkat keras.

2. *Operational level (question)*

   Seperangkat pertanyaan yang digunakan untuk mendefinisikan tujuan dari pengukuran perangkat lunak, dispesifikasikan berdasarkan dengan karakteristik dari model penilaian perangkat lunak yang sudah tersedia. Pertanyaan dibuat untuk mengkategorikan objek dari penilaian (produk, proses, sumber daya) yang mengacu pada model kualitas perangkat lunak dan ditentukan berdasarkan sudut padang yang dipilih.

3. *Quantitative level (metrics)*

   Satu set data yang terkait dengan setiap pertanyaan untuk mendapatkan jawaban secara kuantitatif. Data tersebut dapat berupa:

a. Objektif jika data hasil penilaian tidak didapatkan berdasarkan sudut pandang tertentu, seperti ukuran dari program, dokumentasi terkait versi program.
b. Subjektif jika data hasil dari penilaian didapatkan berdasarkan dari sudut pandang tertentu, seperti tingkatan kepuasan dari pengguna.

Sehingga untuk melakukan pengukuran bobot faktor kualitas perangkat lunak dari masing-masing faktor kualitas yang dominan pada domain perangkat lunak pengolahan citra, digunakan rumus sebagai berikut [5] [6]:

$$A^*_{WSM} = Max_i \sum_{j=1}^{N} a_{ij}\, w_j \quad \textbf{(2.1)}$$

Dimana :

A_{WSM}* merupakan nilai total dari setiap faktor penilaian.

N merupakan jumlah dari kriteria.

a_{ij} merupakan nilai dari setiap kriteria.

w_j merupakan bobot untuk kriteria penilaian.

Kemudian tahapan yang harus dilakukan dalam melakukan proses pengukuran adalah sebagai berikut:

1. Nilai total dari setiap faktor akan dihitung berdasarkan kriteria masing-masing (a_{ij}).
2. Bobot relatif dari setiap kriteria (w_j) ditentukan berdasarkan bobot relatif dari jumlah kriteria yang ditentukan, dengan rumus sebagai berikut [7]:

$$\sum_{j=1}^{n} w_j = 1 \quad \textbf{(2.2)}$$

Dimana :

w_j merupakan bobot dari kriteria

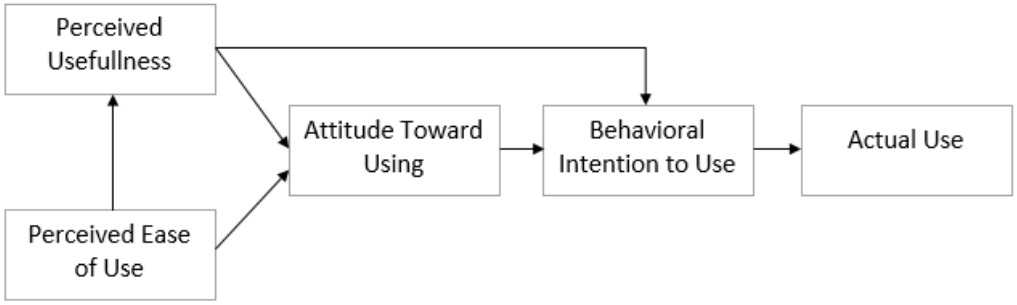
n merupakan jumlah bobot berdasarkan kriteria

Jika terdapat empat kriteria maka bobot dari masing-masing kriteria sebagai berikut : $w_1 = 0.25$, $w_2 = 0.25$ $w_3 = 0.25$ $w_4 = 0.25$, dengan jumlah bobot w jika dijumlahkan harus sama dengan satu.

2.5. Technology Acceptance Model (TAM)

Teori ini merupakan adaptasi dari teori *Theory of Reasoned Action Model (TRA).* TAM ini merupakan sebuah model khusus yang dibuat untuk menilai perilaku/penerimaan pengguna terhadap perangkat lunak. TAM menjelaskan hubungan antara keyakinan/*beliefs* (*usefulness* dan *ease of use*) dengan sikap/*attitude*, tujuan/*intentions* pemakai, serta penggunaan nyata dari sistem [8].

Berikut adalah penjelasan dari masing-masing faktor penerimaan yang terdapat pada model penerimaan *Technology Acceptance Model* (Gambar 2.5):



Gambar 2.5 *Technology Acceptance Model* [7]

1. *Perceived ease of use*

   Perceived ease of use berfokus pada tingkat kepercayaan dari seseorang, jika menggunakan suatu teknologi dapat mempermudah meraka dalam melaukan pekerjaannya.

2. *Perceived usefulness*

   Perceived usefulness akan memfokuskan penilaian berdasarkan kepuasan dari pengguna, rasa percaya dari pengguna jika menggunakan sebuah perangkat lunak dapat meningkatkan kinerjanya.

3. *Attitude toward using*

   Attitude toward using akan menilai mengenai sejauh mana sikap dari pengguna ketika menggunakan sebuah teknologi yang mereka anggap baru, apakah akan terjadi penerimaan atau penolakan.

4. *Behavioral intention to use*

   Behavioral intention to Use menjelaskan mengenai perilaku seorang pengguna, pada saat mereka menggunakan sebuah teknologi yang mereka anggap baru.

5. *Actual use*

   Actual use akan mengukur seberapa banyak waktu yang digunakan oleh pengguna ketika melakukan interaksi dan frekuensi dari penggunaan teknologi/perangkat lunak.

2.6. Metode Pengolahan Kuesioner

Kuesioner merupakan sebuah metode penelitian yang berisi sejumlah pertanyaan ataupun pernyataan yang ditujukan kepada responden tertentu. Kuesioner ini bertujuan dan memiliki maksud untuk mendapatkan sejumlah data yang nantinya akan diolah kembali. Untuk pengolahan data dari hasil kuesioner tersebut, dapat digunakan beberapa metode pengolahan data di antaranya seperti skala linkert dan skala guttman.

1. Skala Linkert

Skala linkert pertama kali dikemukakan pada tahun 1932 oleh Rensis Linkert, yang digunakan ketika melakukan pengukuran sikap terhadap masyarakat. Pengukuran ini menggunakan skala pada masing-masing jawaban dari setiap pertanyaan atau pernyataan yang diajukan terhadap responden berdasarkan tujuan yang telah ditentukan. Skala yang digunakan dalam sebuah pertanyaan atau pernyataan yang biasa diajukan biasanya berisi jawaban Sangat Setuju (SS), Setuju (S), Netral (N), Tidak Setuju (TS) dan Sangat Tidak Setuju (STS). Setiap pertanyaan atau pernyataan yang diajukan bisa memiliki makna yang positif maupun makna yang negatif, tergantung dari konotasi yang digunakan dalam pernyataan atau pernyataan. Penilaian dari skala linkert itu sendiri dapat dilihat pada Tabel 2.1 [9].

Tabel 2.1 Bobot Pengukuran Skala Linkert

Jenis Pertanyaan/Pernyataan	Bobot Penilaian				
	SS	S	N	TS	STS
Positif	5	4	3	2	1
Negatif	1	2	3	4	5

Skor penilaian menurut skala linkert ini akan di tentukan berdasarkan rumus aritmatika mean, yaitu [10]:

$$Z = \frac{X_i}{n.N} \quad \textbf{(2.3)}$$

Dimana:

Z merupakan skor penilaian kinerja

Xi merupakan nilai kuantitatiftotal

n merupakan jumlah responden

N merupakan jumlah item pertanyaan

2. Skala Guttman

Skala guttman biasanya digunakan dalam melakukan pengukuran untuk mendapatkan jawaban yang tegas dari responden. Jawaban dari pertanyaan atu pernyataan dalam skala gutman biasanya berupa Ya atau Tidak, Benar atau Salah, Pernah atau Tidak Pernah dan lain-lain. Berbeda dengan skala linkert yang memiliki lima interval, skala guttman hanya memiliki dua interval sebagai jawaban dari sebuah pertanyaan atau pernyataan yang diajukan.

Tabel 2.2 Bobot Pengukuran Skala Guttman

Jenis Pertanyaan/Pernyataan	Bobot Penilaian	
	Ya	Tidak
Positif	2	1
Negatif	1	2