BAB II
LANDASAN TEORI

2.1. Tinjauan Pustaka

Beberapa penelitian yang berkaitan dengan penerapan aplikasi *chatbot* berdasarkan jurnal penelitian terlihat yaitu:

Tabel 2. 1 Tinjauan Pustaka

No Literatur	Penulis	Judul	Hasil
Literatur 01	(Safitri *and* Rosadi, 2021)	Rancang Bangun Penyedia Layanan Informasi Pelayanan Masyarakat Kantor Kecamatan Pandaan Menggunakan *Chatbot*	Hasil penelitian yaitu *Chatbot* berjalan dengan baik namun masih ada delay dengan rata-rata waktu 58,54 detik sebagaimana terlihat dari pengujian *blackbox testing.* Dan untuk pengujian keefektifisan dan efesiensi *WhatsApp chatbot y*ang didapatkan dari uji kuesioner mendapatkan sebesar 81% dari staf Kantor Kecamatan Pandaan dan 87% dari masyarakat Kecamatan Pandaan yang menunjukkan sangat setuju *WhatsApp chatbot* ini berhasil menjadi penyedia layanan informasi pelayanan masyarakat yang efektif dan efisien.

No Literatur	Penulis	Judul	Hasil
Literatur 02	(Eldi *and* Syaputra, 2020)	Implementasi *chatbot* untuk mendukung sistem informasi pada rumah sakit muhamadiyah palembang	Pembuatan Sistem informasi *chatbot* di Rs Muhammadiyah Palembang dilakukan untuk mempermudah pasien mengetahui tentang informasi pelyanan Rs.Seperti Jumlah kamar Pasien yang tersedia
Literatur 03	(Guntoro, Costaner *and* Lisnawita, 2020)	Aplikasi *Chatbot* untuk Layanan Informasi dan Akademik Kampus Berbasis *Artificial Intelligence Markup Language (AIML)*	aplikasi *chatbot* yang dapat digunakan sebagai layanan informasi kampus dan akademik bagi masyarakat umum maupun bagi civitas akademik kampus Universitas Lancang Kuning Adapun tahapan dalam pengembangan aplikasi *chatbot* inidiantaranya adalah pengumpulan kebutuhan, desain, membuat *prototype*, evaluasi dan perbaikan. Adapun metode yang digunakan untuk pembelajaran *chatbot* menggunakan *Artificial Markup Language (AIML). Knowledge*
Literatur 04	(Murhadi, 2019)	Rancang Bangun Aplikasi *Chatbot*	Hasil perancangan dan

No Literatur	Penulis	Judul	Hasil
		Sebagai Bentuk Pelayanan Prima Untuk Penerimaan Mahasiswa Baru	pengembangan aplikasi *Chatbot* yaitu implementasi *Chatbot* dapat dijadikan sebagai pengganti peran manusia dalam memberikan layanan prima. *Chatbo*t ini dapat memberikan jawaban yang cepat dan lengkap. Dari segi faktor pelayanan prima, *Chatbot* ini dapat memenuhi faktor sikap *(attitude)*, perhatian *(attention)*, tindakan *(action)*, kemampuan *(ability)*, penampilan *(appearance)* dan tanggung jawab *(accountability)*.
Literatur 05	(Benedictus, Wowor *and* Sambul, 2017)	Rancang Bangun *Chatbot Helpdesk* untuk Sistem Informasi Terpadu Universitas Sam Ratulangi	Menghasilkan aplikasi *Chatbot Helpdesk* yang bisa membantu user dengan menjawab pertanyaan seputar penggunaan aplikasi-aplikasi dalam Sistem Informasi Terpadu Universitas Sam Ratulangi.

Bersadasarkan penelitian terdahulu terdapat beberapa perbedaan dan kelebihan sistem yang dibangun yaitu :

1. Metode yang digunakan yaitu metode AIML
2. Sistem yang dibangun dapat menambah pertanyaan dan jawaban untuk informasi yang ditampilakan di *chatbot.*
3. Menggunakan metode pengembangan sistem *extreme programming*
4. Sistem ini akan diuji menggunakan ISO 25010 dengan aspek *functionality* dan *reliability.*

2.2. Rancang Bangun

Rancang Bangun adalah penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah kedalam suatu kesatuan yang utuh dan berfungsi. Dengan demikian pengertian rancang bangun merupakan kegiatan menerjemahkan hasil analisa ke dalam bentuk paket perangkat lunak kemudian menciptakan sistem tersebut atau memperbaiki sistem yang sudah ada (Pressman, 2015).

2.3. Aplikasi

Aplikasi merupakan sebuah Perangkat lunak (*software*) yang bertugas sebagai *font end* di suatu sistem yang digunakani dalam mengolah bermacam-macam data sehingga menjadi sebuah informasi yang bermanfaat bagi penggunanya dan juga sistem-sistem yang berkaitan (Widianti, 2015).

Aplikasi yakni merupakan kumpulan perintah program yang dibuat dan rancang untuk melakukan kegiatan maupun pekerjaan-pekerjaan tertentu (Pressman, 2015)

2.4. Chatbot

Menurut Utdirartatmo (2017) *Chatbot* merupakan sebuah sistem berbasiskan Natural *Language* (bahasa alami). *Chatbot* merupakan salah satu pengembangan sistem percakapan antara manusia dengan mesin/komputer. Percakapan yang terjadi terbatas atas *knowledge base chatbot* itu sendiri. *Grammar* (aturan berbahasa yang benar) dan struktur bahasa tersebut menjadi sebuah batasan untuk menjadi sebuah kata kunci dalam filtrasi untuk merespon inputan oleh pengguna.

Menurut Wallace (2018) *Chatbot* adalah karakter bahasa alami yang berkomunikasi dengan penggunanya, atau orang-orang yang sedang *chatting* di *internet* bahkan melalui komunikasi suara seperti telepon. Setiap *chatbot* diatur oleh seorang *botmaster,* yaitu orang dibelakang layar yang berandil besar dalam membentuk suatu kepribadian bot serta pengetahuan *chatbot.*

Berdasarkan beberapa pengertian diatas maka dapat disimpulkan bawah *chatbot* adalah karakter bahasa alami yang berkomunikasi dengan penggunanya menggunakan percakapan yang terjadi terbatas atas *knowledge base chatbot* itu sendiri.

2.5. Metode AIML

AIML merupakan subset dari *Extensible Markup Language* (XML) namun dengan fungsi yang mendetail. Salah satu fungsinya adalah membuat sistem input pertanyaan-balasan berbasis pengetahuan, pada AIML tag dapat diartikan berbeda sesuai dengan fungsi masingmasing (Wallace, 2018). Bagian-bagian penting dari AIML adalah sebagai berikut

a. *Category* adalah inti dari pengetahuan, disetiap *category* wajib terdapat pattern dan template. Dibawah ini adalah gambaran penggunaan *category* (Wallace, 2018). Pada saat *category* tersebut dipanggil maka bot akan membalas inputan user "siapa orang tuamu" dengan "kamu tidak perlu tahu orang tuaku".

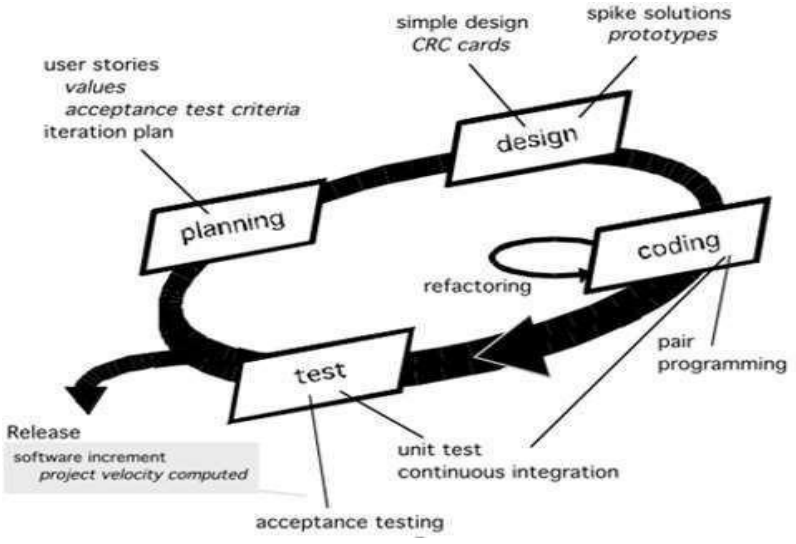
b. *Pattern* merupakan sebuah pertanyaan/inputan, *pattern* dapat direspon dengan satu atau lebih Balasan. Suatu pattern yang dianggap sama disebut *wildcard*, dibawah ini adalah cara penggunaan *wildcard*: Sepadan dengan inputan"siapa nama kamu","siapa nama dosen kamu", dan sebagainya (Wallace, 2018).

c. *Template* merupakan jawaban dari satu pattern atau lebih. Dibawah adalah contoh penggunaan *Template,* Variabel atau sebuah inputan yang berarti sama seperti nama *bot* dan disisipkan ke dalam kalimat. *Template* juga memungkinkan untuk meneruskan ke pattern lain dengan menggunakan elemen AIML bernama srai. Elemen srai dapat digunakan untuk mengimplementasikan persamaan arti seperti pada contoh berikut. Pada *Category* yang berisikan *Pattern* “dadah” yang berarti inputan selamat tinggal yang akan dibalas “selamat tinggal”. *Category* selanjutnya menjawab inputan “sampai jumpa” sama seperti inputan “dadah”, dengan kata lain sebuah inputan akan diteruskan ke category yang dianggap sama dengan di sintakskan <srai> yang menunjukkan bahwa dia satu category inputan, maka dapat diartikan bahwa kedua inputan kata diatas bernilai sama (Wallace, 2018).

d. *That* adalah element AIML yang merujuk pada balasan, tanggapan sebelumnya. *That* digunakan pada category ketika balasan/tanggapan masih bersangkutan dengan inputan atau balasan sebelumnya. Dalam perancangan *template* Wallace (2018) berpendapat bahwa pattern diartikan sebagai inputan bisa berarti sebuah pertanyaan, template diartikan sebagai sebuah balasan, *template* sendiri bisa berarti sebuah jawaban tunggal maupun jawaban random.

2.6. Metode Pengembangan Sistem

Extreme Programming (XP) atau Pemograman Ekstreme yaitu suatu pendekatan yang paling banyak digunakan untuk pengembangan perangkat lunak cepat (Pressman, 2012). Pada pengembangan ini terdapat beberapa konteks kegiatan kerangka kerja, perencanaan, perancangan, pengkodean, dan pengujian.



Gambar 2. 1 Ilustrasi Proses *Extreme Prgamming*

Sumber: (Pressman, 2012)

1. *Planning* (Perencanaan)

Kegiatan Perencanaan (disebut juga *planning game*) biasanya dimulai dengan mendengarkan suatu kegiatan yang bertujuan mengumpulkan

kebutuhan-kebutuhan untuk memahami konteks bisnis dan perlunya keluaran-keluaran (*output*), fungsi utama, dan *fungsionalitas.*

2. *Design* (Perancangan)

   Perancangan yang simple, menarik, dan sederhana selalu memberikan hasil yang lebih disukai daripada gambaran-gambaran yang lebih kompleks. Perancangan XP memberikan panduan implementasi untuk suatu cerita ketika ditulis, tidak kurang, tidak lebih.

3. *Coding* (Pengkodean)

   Pengkodean ini dilanjutkan setelah cerita yang telah dikembangkan dan rancangan yang telah dilakukan oleh tim perangkat lunak. Pengkodean ini tidak langsung mengarah ke kode-kode program. Tim akan mengembangkan serangkaian unit pengujian lalu beralih ke pengkodean.

4. *Pengujian* (Pengujian)

   Unit pengujian yang harus dibuat dan kemudian dijalankan menggunakan kerangka kerja yang memungkinkan mereka untuk diotomatisasi sehingga dapat dijialankan dengan mudah dan dapat dijalankan berulang kali.

2.7. Bahasa Pemodelan Pengembangan Sistem (*UML*)

Bahasa Pemodelan Pengembangan Sistem (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement* (kebutuhan), membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek (Rosa & Shalahuddin, 2018).

2.7.1. Use Case Diagram

Use case diagram atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat (Rosa & Shalahuddin, 2018). *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat menjelaskan simbol-simbol yang ada pada diagram *use case* dapat dilihat pada tabel 2.2 di bawah ini:

Tabel 2. 2 Simbol Diagram *Use Case*

Simbol	Deskripsi
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*
Aktor/*actor* 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama *actor*
Asosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor*

Simbol	Deskripsi
Ekstensi/*extend* <<*extend*>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek biasanya *use case* tambahan memiliki nama depan
Generalisasi/***generalization*** 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
Menggunakan/*Include/uses* <<*include*>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

Sumber : (Rosa & Shalahuddin, 2018).

2.7.2. Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi (Rosa and Shalahudin, 2018). menjelaskan simbol-simbol yang ada pada diagram kelas pada tabel *class diagram* 2.3:

Tabel 2. 3 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas **nama_kelas**	Kelas pada struktur sistem

Simbol	Deskripsi
+atribut +operasi()	
Antarmuka/*Interface* **nama_*interface***	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi/*asociation*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah/*directed association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya disertai dengan *multiplicity*
Generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi(umum khusus)
Kebergantungan/*dependecy*	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi/*agregation*	Relasi antar kelas dengan makna semua bagian (*whole-part*)

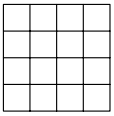
Sumber : (Rosa & Shalahuddin, 2018).

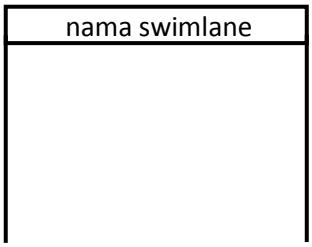
2.7.3. Activity Diagram

Activity diagram atau Diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis (Rosa & Shalahuddin,

2018), menjelaskan Simbol-simbol yang ada pada *activity diagram*dapat dilihat pada tabel 2.4 di bawah ini:

Tabel 2. 4 Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem,sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Tabel 	Suatu file komputer dari mana data bisa dibaca atau direkam selama kejadian bisnis
Dokumen 	Menunjukan dokumen sumber atau laporan
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

Simbol	Deskripsi
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber : (Rosa & Shalahuddin, 2018).

2.8. Alat Implementasi

2.8.1. Xampp

Menurut MADCOMS (2016) Xampp adalah sebuah paket kumpulan software yang terdiri dari Apache, MySQL, PhpMyAdmin, PHP, Perl, Filezilla, dan lain.

Xampp berfungsi untuk memudahkan instalasi lingkungan PHP, di mana biasanya lingkungan pengembangan web memerlukan PHP, Apache, MySQL dan PhpMyAdmin (MADCOM, 2016).

2.8.2. Dreamwever

Adobe Dreamweaver adalah :aplikasi desain dan pengembangan web yang menyediakan editor WYSIWYG visual (bahasa sehari-hari yang disebut sebagai *Design view*) dan kode editor dengan fitur standar seperti *syntax highlighting*, *code completion*, dan *code collapsing* serta fitur lebih canggih seperti *real-time syntax checking* dan *code introspection* untuk menghasilkan petunjuk kode untuk membantu pengguna dalam menulis kode (Destiningrum and Adrian, 2017).

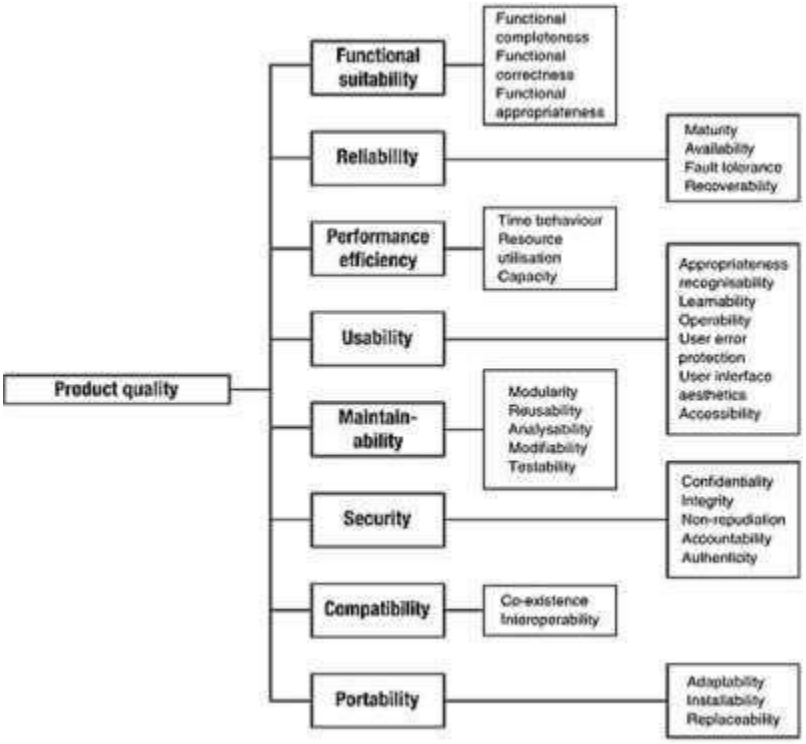
2.8.3. MySQL

Menurut MADCOM (2016) MySQL adalah sistem manajemen Database SQL yang bersifat *Open Source* dan paling popular saat ini. Sistem Database MySQL mendukung beberapa fitur seperti *multithreaded*, *multiuser* dan SQL Database managemen system (DBMS).

2.9. Pengujian Sistem ISO 25010

Model ISO-25010 merupakan bagian dari *Software product Quality Requirements and Evaluation* (*SQuaRE*), yang merupakan pengembangan dari model kualitas perangkat lunak sebelumnya yaitu ISO-9126. Dalam model ISO-25010 ini digunakan untuk melihat kualitas suatu perangkat lunak yang digunakan oleh perusahaan, instansi ataupun organisasi. Metode ISO 25010 ini dapat digunakan untuk mengevaluasi kualitas sistem perangkat lunak secara spesifik berdasarkan dua dimensi umum, yaitu dimensi *product quality*, dimana prosesnya mengacu pada karakteristik intrinsik dari sebuah produk perangkat lunak, memiliki beberapa elemen antara lain meliputi *functional suitability, reliability, operability, performance efficiency, security, compatibility, maintainability* dan *transferability. Quality in use* dan *product quality*. ISO/IEC 25010 merupakan model kualitas sistem dan perangkat lunak yang menggantikan ISO/IEC 9126 tentang *software engineering* (Iqbal, 2016). *Product quality* ini juga digunakan untuk tiga model kualitas yang berbeda untuk produk perangkat lunak antara lain:

1. Kualitas dalam model penggunaan
2. Model kualitas produk
3. Data model kualitas



Gambar 2. 2 Model kualitas produk ISO/IEC 25010

Adapun dimensi yang pertama terdapat beberapa faktor elemen diantaranya:

1) *Functionality* (Fungsionalitas). Kemampuan perangkat lunak Merupakan tingkatan dimana perangkat lunak dapat menyediakan fungsionalitas yang dibutuhkan ketika perangkat lunak digunakan pada kondisi spesifik tertentu dalam hal ini perangkat lunak dapat memenuhi kelayakan dari sebuah fungsi untuk melakukan pekerjaan yang spesifik bagi pengguna dan dapat memberikan hasil yang tepat dan ketelitian terhadap tingkat kebutuhan pengguna.
2) *Reliability* Merupakan tingkatan dimana perangkat lunak dapat bertahan pada tingkatan tertentu ketika digunakan oleh pengguna pada kondisi yang spesifik dalam hal ini perangkat lunak dapat beroperasi dan siap ketika dibutuhkan untuk digunakan dan juga dapat bertahan pada tingkat

kemampuan tertentu terhadap kegagalan, kesalahan serta perangkat lunak kembali pada tingkat tertentu dalam mengembalikan pengembalian data yang disebabkan kegagalan atau kesalahan pada perangkat lunak.

3) *Performance efficiency* Merupakan tingkatan dimana perangkat lunak dapat memberikan kinerja terhadap sejumlah sumber daya yang digunakan pada kondisi tertentu dalam hal ini *performance efficiency* dapat memberikan reaksi dan waktu yang dibutuhkan ketika melakukan aksi dari sebuah fungsi dan perangkat lunak dapat menggunakan sejumlah sumber daya ketika melakukan aksi dari sebuah fungsi.
4) *Usability* Perangkat lunak dapat dimengerti, dipelajari, digunakan dan menarik pengguna ketika digunakan dalam hal ini perangkat lunak mudah dipelajari oleh pengguna, perangkat lunak dapat digunakan dan dioperasikan oleh pengguna.
5) *Security* Merupakan perlindungan terhadap perangkat lunak dari berbagai ancaman atau keganjalan dalam hal ini perangkat lunak memiliki perlindungan terhadap data atau informasi dari pengguna dan merupakan dari kelengkapan, ketepatan dari sejumlah *asset* yang telah dijaga sehingga aksi atau tindakan yang dilakukan telah terbukti dan hal tersebut tidak dapat ditolak.
6) *Compability* Faktor ini merupakan kemampuan dari dua atau lebih komponen perangkat lunak dapat melakukan pertukaran informasi dan melakukan fungsi yang dibutuhkan ketika digunakan pada *hardware* atau lingkungan perangkat lunak yang sama.

7) *Maintainability* Merupakan tingkat dimana sebuah perangkat lunak dapat dimodifikasi. Dalam hal ini modifikasi adalah perbaikan, perubahan atau penyesuaian perangkat lunak untuk dapat berubah pada lingkungan, kebutuhan dan fungsionalitas yang spesifik. Selain itu perangkat lunak dapat dianalisi untuk mengetahui apa yang menyebabkan kegagalan pada perangkat lunak untuk mengidentifikasi bagian yang dapat dimodifikasi.
8) *Transferability* Merupakan kemudahan dimana sistem atau komponen dapat berpindah dari lingkungan satu ke lingkungan yang lain dalam hal ini perangkat lunak dapat beradaptasi dengan cepat pada spesifikasi lingkungan yang berbeda tanpa menerapkan aksi atau cara lain dari pada memberikan tujuan tertentu terhadap perangkat lunak yang telah ada.