BAB II

LANDASAN TEORI

1.1 Tinjauan Pustaka

Rujukan Penelitian yang pernah dilakukan untuk mendukung Penulisan Skripsi ini antara lain:

1) Umma Ridho Fuadah (2015) melakukan penelitian tentang Pengembangan dan Analisis Laboratorium Jurusan Pendidikan Teknik Elektronika FT UNY "LABORASTORY" Berbasis *WEB*. Dari penelitian ini disimpulkan bahwa Sistem yang telah dibangun di Jurusan Pendidikan Teknik Elektronika FT UNY menggunakan desain arsitektur 2 aktor, 28 *use case*, 5 *class*, 35 prosedur fungsi; 7 tabel data; 2 desain *interface* yang berbeda untuk *admin* dan *user* biasa, serta telah layak untuk digunakan berdasarkan hasil pengujian yang dinilai dari aspek-aspek dalam ISO 9126, yaitu dimana sistem telah terdapat aspek *functionality*, *reliability*, *usability*, *efficiency*, dan *portability*.

2) Penelitian yang dilakukan oleh Ike Puspita Wulan Sari, Bambang Eka Purnama, dan Sukadi yang berjudul "Pembangunan Sistem Informasi Inventaris Barang Sekolah Dasar Negeri (SDN) Pacitan". Jenis studi dalam penelitian ini adalah Penelitian dan Pengembangan (*Research and Development*). Penelitian inidilakukan di SD Negeri di Pacitan, Jawa Timur. Hasil penelitian yang dihasilkan adalah sistem informasi inventaris

barang ini dapat digunakan sebagai pengganti media konvensional yaitu menggunakan media buku. Sistem informasi ini terbukti memberikan waktu lebih singkat dari waktu pengelolaan konvensional. Waktu pembuatan laporan konvensional yang dilakukan selama 60 menit dapat dilakukan menggunakan sistem inventaris ini selama 10 menit. Sistem informasi ini dapat membantu kesulitan pendataan barang inventaris di SDN Pacitan. Hal ini telah dibuktikan berdasarkan kuesioner yang telah penulis edarkan sebagai tindak lanjut dari implementasi sistem informasi tersebut. Relevansi antara penelitian tersebut dengan penelitian yang dilakukan penulis adalah persamaan dalam metode penelitian dan pengembangan sistem informasi inventaris barang. Perbedaan dengan penelitian yang dilakukan penulis adalah macam pengujian dan tambahan mengenai uji kualitas perangkat lunak sesuai ISO 9126.

3) Penelitian yang berjudul "Analisis dan Perancangan Sistem Informasi untuk pengelolaan Inventaris Laboratorium pada STMIK AMIKOM Yogyakarta" yang dilakukan oleh Yudi Sutanto di UPT Laboratorium STMIK AMIKOM Yogyakarta. Jenis studi dalam penelitian ini adalah Penelitian dan Pengembangan (*Research and Development*). Hasil penelitian yang didapat yaitu pengembangan sistem baru telah terlaksana dengan berbagai perbaikan dari sistem awal. Setelah sistem diimplementasikan, baru perlu melakukan pengujian penerimaan sistem (*system acceptance test*). Pengujian ini berbeda dengan pengujian sistem sebelumnya. Pada pengujian ini dilakukan dengan menggunakan data yang

sesungguhnya dalam jangka waktu tertentu yang dilakukan oleh analis sistem bersama dengan *user*. Setelah uji penerimaan dilakukan, suatu rapat penerimaan (*acceptance meeting*) perlu diselenggarakan oleh manajemen yang dihadiri oleh analis sistem, manajer dan pemakai sistem untuk menentukan sistem diterima atau tidak. Jika disetujui maka diadakan rapat penyerahan sistem. Relevansi antara penelitian tersebut dengan penelitian yang dilakukan penulis adalah persamaan dalam metode penelitian, pengembangan sistem informasi inventaris, dan tempat penelitian di laboratorium. Perbedaan dengan penelitian yang dilakukan penulis adalah macam pengujian dan tambahan mengenai uji kualitas perangkat lunak sesuai ISO 9126.

4) Penelitian yang berjudul "Perancangan Sistem Informasi Inventaris Program Studi Teknik Informatika Universitas Surakarta" yang dilakukan oleh Adita Ayu Prawiyanti dan Ramadhian Agus Triyono. Jenis studi penelitian ini adalah Penelitian dan Pengembangan (*Research and Development*). Penelitian ini menghasilkan sebuah rancangan sistem informasi inventaris pada Program Studi Teknik Informatika Universitas Surakarta sebagai media penyampaian informasi data barang inventaris yang efektif dan efisien. Adanya sistem informasi inventaris ini dapat mempermudah untuk mengetahui data inventaris yang dimiliki, dapat menyajikan laporan data inventaris tepat waktu sehingga pengambilan keputusan dapat dilakukan lebih cepat, dan mempermudah dalam proses *back up* data. Relevansi antara penelitian tersebut dengan penelitian yang

dilakukan penulis adalah persamaan dalam metode penelitian dan pengembangan sistem informasi inventaris. Perbedaan dengan penelitian yang dilakukan penulis adalah macam pengujian dan tambahan mengenai uji kualitas perangkat lunak ISO 9126.

1.2 Studi Pustaka

1.2.1 Sistem Informasi Inventaris

Sistem Informasi didefinisikan sebagai suatu sistem yang menerima sumber data sebagai *input* dan mengolahnya menjadi produk informasi *output*. Sistem Informasi merupakan suatu sistem yang terdiri dari beberapa subsistem (komponen *hardware*, perangkat lunak, *brainware*), data dan prosedur untuk menjalankan *input*, proses, *output*, penyimpanan, dan pengontrolan yang mengubah sumber data menjadi informasi (Marimin, Tanjung, & Prabowo, 2006).

Menurut Kamus Besar Bahasa Indonesia (Tim Penyusun Kamus Pusat Bahasa), Inventaris adalah daftar yang memuat semua barang milik kantor (Sekolah, Perusahaan, Kapal, dan lain-lain) yang dipakai dalam melakukan tugas.

Jadi, Sistem Informasi Inventaris dapat diartikan sebagai sistem pengolah data barang milik kantor sehingga terbentuk suatu informasi. Pengolahan data barang disini meliputi pengolahan input, proses, output, penyimpanan, dan pengontrolan.

1.2.2 Perangkat Lunak Berbasis *WEB*

Perangkat lunak berbasis *web* (*web based software*) merupakan perangkat lunak yang dapat diakses dengan menggunakan *browser* (S. & Shalahuddin, 2013, hal. 3). Definisi lain aplikasi *web* yaitu program yang berjalan di dalam keseluruhan atau pada sebagian *server web* dan dapat dijalankan oleh pengguna melalui situs *web* (Simarmata, 2010). Jadi, perangkat lunak berbasis *web* adalah program yang berjalan pada *server web* dan dapat diakses menggunakan *browser*.

1.2.3 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak (*software engineering*) merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunakan mesin (S. & Shalahudin, 2013).Rekayasa perangkat lunak lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat kepada pelanggan (*customer*) (S. & Shalahuddin, 2013). Metode rekayasa perangkat lunak merupakan pendekatan terstruktur terhadap pengembangan perangkat lunak yang bertujuan memfasilitasi produksi perangkat lunak kualitas tinggi dengan cara yang efektif dalam hal biaya (Sommervile, 2003)

Berdasarkan beberapa teori ahli, dapat disimpulkan bahwa rekayasa perangkat lunak adalah proses pengembangan perangkat lunak yang diharapkan menjadi tahapan yang efisien. Salah satu model proses perangkat lunak yang sering digunakan adalah model air terjun (*waterfall*).

Menurut Ian Sommerville (2003, hal. 42), model air terjun mengambil kegiatan proses dasar; seperti spesifikasi, pengembangan, validasi, dan evaluasi; dan mempresentasikannya sebagai fase-fase proses yang berbeda seperti spesifikasi persyaratan, perancangan perangkat lunak, implementasi, pengujian, dan perawatan. Kelebihan dari metode ini, seperti pada tulisan Rosa A. S. dan M. Shalahuddin (2013), adalah model pengembangan yang paling sederhana, dan sesuai dengan produk yang spesifikasinya tidak berubah-ubah.

Tahap-tahap utama dari model air terjun ini memetakan kegiatan pengembangan dasar sebagai berikut (Sommerville, 2003).

1) Analisis dan definisi persyaratan

Dalam tahap ini, ditentukan pelayanan, batasan dan tujuan sistem melalui wawancara ataupun observasi terhadap *user* sistem.Persyaratan yang ditentukan dalam tahap ini menghasilkan suatu spesifikasi sistem.

2) Perancangan sistem dan perangkat lunak

Dalam tahap ini, ditentukan arsitektur sistem secara keseluruhan.Persyaratan yang telah didefinisikan dibagi dalam sistem perangkat keras atau perangkat lunak. Menurut Pressman (2002,) tahap desain meliputi representasi data, arsitektur, *interface*, dan prosedur.

3) Implementasi dan pengujian unit

Implementasi atau generasi kode merupakan langkah penerjemah desain ke dalam bentuk bahasa mesin (Pressman, 2002).Tahap ini merupakan saat realisasi dari perancangan, yaitu berupa serangkaian

program.Pengujian unit di sini merupakan verifikasi bahwa setiap unit telah memenuhi spesifikasinya.

4) Integrasi dan pengujian sistem

Pada tahap ini, program individual diintegrasikan dan diuji sebagai sistem yang lengkap untuk memenuhi persyaratan.Pengujian fokus pada perangkat lunak secara dari segi logik dan fungsional dan memastikan bahwa semua bagian sudah diuji (S. & Shalahuddin, 2013, hal. 30).Tahapan pengujian secara keseluruhan adalah sebagai berikut (S. & Shalahuddin, 2013).

1) Pengujian Unit: Pengujian pada kumpulan fungsi atau kelas, dapat berupa modul yang dikenal sebagai *package*.
2) Pengujian Integrasi: Pengujian pada dua atau lebih unit.
3) Pengujian Sistem: Pengujian pada sistem perangkat lunak secara keseluruhan dan diuji secara satu sistem.
4) Pengujian Penerimaan : Pengujian yang dilakukan untuk mengetahui kepuasan pelanggan atau *user* terhadap perangkat lunak yang sudah dibuat.

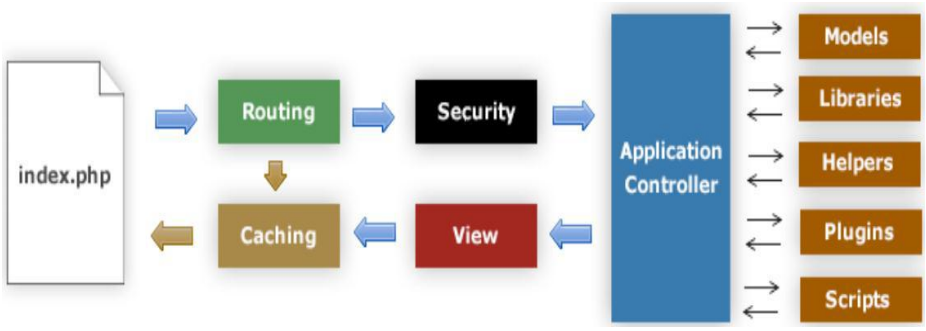
5) Operasi dan pemeliharaan

Tahap ini adalah tahap yang menghabiskan waktu paling lama.Sistem diinstal dan digunakan.Pemeliharaan pada sistem dilakukan untuk menyelesaikan kekurangan yang ditemukan, perbaikan implementasi, dan pengembangan pelayanan sistem.

1.2.4 *Framework* CodeIgniter

CodeIgniter (CI) (www.codeigniter.com) adalah salah satu *framework* PHP yang tangguh dan popular. CodeIgniter tergolong *framework* yang digunakan untuk membuat sebuah aplikasi berbasis web yang disusun dengan menggunakan bahasa PHP. Didalam CI ini terdapat beberapa macam kelas yang berbentuk *library* dan *helper* yang berfungsi untuk membantu pemrogram dalam mengembangkan aplikasi. CI juga mempunyai file dokumentasi yang sangat memadai untuk menjelaskan setiap fungsi yang ada pada library dan helper. File dokumentasi ini disertakan secara langsung pada saat Pengunduhan paket *framework* CI.

CodeIgniter menggunakan konsep MVC (*Model View Controller*). Menurut Akhmad Sofwan (2003), Konsep MVC adalah konsep pemisahan antara logika dengan tampilan dan *database*. Manfaat dari konsep ini adalah membuat pengodean logika lebih *simple*, karena sudah dipisah dengan kode untuk tampilan dan membuat *programmer* dapat bekerja secara terpisah dengan desainer.*Programmer* mengerjakan logika, sedangkan desainer berkutat dengan desain dan tampilan.*Model*(M) berisi kode penghubung ke database, *View*(V) berisi kode desain tampilan, dan *controller*(C) berisi kode logika. Gambar 2.1 berikut adalah gambaran kerja *Framework* CodeIgniter.



Gambar 2.1 Cara Kerja CodeIgniter

Keunggulan CodeIgniter sesuai dalam *user guide* CodeIgniter sendiri (EllisLab, 2014), *Framework* ini memungkinkan untuk mengembangkan proyek jauh lebih cepat daripada pengodean PHP tanpa *framework*. Penyediaan satu set dengan banyak *library* untuk tugas yang biasa diperlukan, serta antarmuka yang sederhana dan struktur logis untuk mengakses *library* tersebut, CodeIgniter memungkinkan pengembang bekerja fokus pada proyek dengan meminimalkan jumlah kode yang dibutuhkan.

1.2.5 Pengujian Aplikasi *Web*

Pengujian perangkat lunak adalah elemen kritis dari jaminan kualitas perangkat lunak dan merepresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean (Pressman, 2002). Sedangkan pengujian jaminan kualitas aplikasi berbasis *web*, seperti tulisan James C. Helm (2000), adalah suatu pola tindakan terencana dan sistematis yang diperlukan untuk membuktikan bahwa produk yang dihasilkan sesuai

dengan kebutuhan klien. Pengujian ini merupakan bagian paling penting dalam jaminan kualitas.

Berikut adalah beberapa *tool* yang dapat digunakan untuk menguji perangkat lunak, khususnya perangkat lunak berbasis *web*.

1) WebPageTest

Sesuai informasi yang tercantum dihalaman *website*-nya (webpagetest.org, 2014), WebPageTest adalah proyek terbuka yang dikembangkan dan didukung oleh Google sebagai bagian dari usaha untuk membuat *web* lebih cepat.*Website* ini menyediakan pengujian menggunakan browser dan alat pembuka yang berbeda-beda. Selain bisa digunakan untuk uji *portability*, website ini bisa menghasilkan informasi mengenai lama *load time*, *performance review*, *page speed*, *content breakdown*, *domain*, dan *screen shot* dari *web* yang diuji.

2) LoadImpact

Menurut *website* resminya, LoadImpact adalah *web* penyedia jasa gratis *load testing* serta pencatatan mengenai suatu *web*.LoadImpact dapat memberikan simulasi puluhan atau ribuan *user* yang mengakses *website*secara bersamaan. Penyedia jasa ini sudah digunakan lebih dari 1.000.000 pengujian *web* (LoadImpact AB, 2014).

3) Webserver Stress Tool

Dalam panduan manualnya (Paessler AG, 2008), Webserver Stress Tool dapat digunakan untuk membuat simulasi berbagai macam *load patterns* untuk menguji *web server*.Hal ini dapat membantu menemukan masalah pada *web server* yang digunakan.Dengan *tool* ini pengembang bisa melihat berapa banyak *load* yang bisa ditangani oleh *server* sebelum terjadi masalah di kemudian hari.

4) GTmetrix

GTmetrix adalah *tool* yang berfungsi untuk melihat performa *website*.GTmetrix menggunakan Google Page Speed dan Yahoo! Yslow untuk menilai performa dan memberikan rekomendasi untuk memperbaiki performa *website* yang diuji (Gossamer Threads, 2014). Menurut Jean Galea (2012), dengan dua tes tersebut dapat membantu pengembangan yang lebih cepat, lebih efisien, dan meningkatkan kenyamanan *user* saat menggunakan *website* secara menyeluruh.

1.2.6 Jaminan Kualitas Perangkat Lunak (*Software Quality Assurance*)

Jaminan kualitas perangkat lunak, sesuai dengan tulisan Roger S. Pressman (2002, hal. 223), merupakan pola tindakan yang terencana dan sistematik yang digunakan untuk menjamin kualitas perangkat lunak.Sedangkan kualitas perangkat lunak didefinisikan sebagai

kesesuaian yang diharapkan pada semua perangkat lunak yang dibangun dengan mengutamakan fungsi, untuk kerja, standar pembangunan yang terdokumentasi dan karakteristik yang ditunjukkannya.

Menurut Conflair Inc. (2012), Jaminan Kualitas Perangkat Lunak (SQA) dan Pengujian sering dipahami sebagai aktivitas yang tidak bersangkutan. Jaminan kualitas fokus pada proses, sedangkan pengujian mengevaluasi produk. Namun keduanya mempunyai tujuan sama yaitu untuk mencapai jaminan kualitas. Hubungan keduanya bisa disebut sebagai komplemen seperti Yin dan Yang.

Gambar 2.2 Komplemen antara Jaminan Kualitas dan Pengujian

Pengukuran kualitas perangkat lunak didasarkan pada standar kualitas tertentu atau sering disebut model kualitas.*Quality Model* atau model kualitas, seperti yang ditulis oleh Syahrul Fahmy, Nurul Haslinda, Wan Roslina dan Ziti Fariha (2012, hal. 116), adalah himpunan karakteristik dan hubungan antar karakter tersebut yang bisa dijadikan dasar untuk menentukan syarat kualitas dan untuk mengevaluasi produk. Terdapat beberapa model pengujian perangkat lunak yang banyak

digunakan, antara lain adalah model McCall, Boehm, FURPS, Dromey, Bayesian, dan ISO 9126.

Tiap model kualitas terdiri dari beberapa karakteristik, yang mempunyai cabang yang lebih spesifik disebut subkarakteristik. Karakteristik dan subkarakteristik ini mempunyai pengertian khusus seperti pada jurnal susunan Botella, et al.(2013). Karakteristik dan subkarakterikstiknya menghasilkan hirarki yang sempurna.Karakteristik dalam model kualitas diartikan sebagai faktor kualitas yang tidak bisa diukur dan digunakan dengan tujuan pengklasifikasian subkarakteristik dari model tersebut. Subkarakteristik dalam model kualitas dapat didefinisikan sebagai faktor kualitas yang secara subyektif dapat diukur sesuai kebutuhan, dan dapat dikomposisi menjadi subkarakteristik lain atau dengan alternatif menggunakan atribut yang membantu dalam pengukurannya.

Berdasarkan jurnal tulisan Dr Rafa E. Al-Qutaish (2010), model kualitas ISO 9126-1 yang dibuat oleh *International Organization for Standardization* (ISO) dan *International Electrotechnical Commission* (IEC) ini adalah model yang paling efisien karena pengembangannya berdasarkan konsensus internasional dan merupakan persetujuan dari semua Negara anggota organisasi ISO. Kelebihan lain dari ISO 9126; menurut Anita Hidayati, Sarwosri, dan Ariadi Retno Tri Hayati Ririd (2009, hal. 2); adalah pada struktur hirarki, kriteria evaluasi, bentuk dan ekspresi yang komprehensif, definisi yang akurat dan sederhana, serta

hubungan *one-to-many* pada setiap layernya. Kelebihan lain menurut Anita Hidayati, Sarwosari, dan Ariadi Retno Tri Hayati Ririd (2009, hal. 4), berdasarkan struktur model kualitas, ISO 9126 memiliki analisis lebih baik jika dibandingkan dengan keempat model kualitas yang lain.

Karakteristik kualitas internal dan eksternal dalam ISO 9126 menurut jurnal tulisan Dr. Rafa E. Al-Qutaish (2010) terdiri dari enam karakteristik kualitas yaitu *Functionality*, *Reliability*, *Usability*, *Maintainability*, dan *Portability*; yang dibuat menjadi 21 subkarakteristik. Pembahasan setiap karakteristik yaitu sebagai berikut:

1) *Functionality*

Functionality atau fungsionalitas adalah kemampuan perangkat lunak untuk menyediakan fungsi yang sesuai kebutuhan pengguna ketika digunakan dalam kondisi tertentu. Karakteristik *functionality* ini terdiri dari 4 subkarakteristik sebagai berikut:

1) *Suitability*: Kemampuan perangkat lunak untuk menyediakan fungsi yang tepat untuk tugas tertentu sesuai kebutuhan dan tujuan *user*
2) *Accuracy*: Kemampuan perangkat lunak untuk memberikan hasil kerja yang cermat.
3) *Security*: Kemampuan perangkat lunak untuk menjaga informasi dan data sehingga orang atau sistem yang tidak sah

tidak bisa membaca ataupun mengubah informasi, sedangkan mengizinkan orang yang sah untuk mengakses sistem.

4) *Interoperability*: Kemampuan perangkat lunak untuk bekerja sama dengan sistem lain.

2) *Reliability*

Reliability atau kehandalan yaitu kemampuan perangkat lunak untuk mempertahankan tingkat kinerja tertentu ketika digunakan dalam kondisi tertentu. Subkarakteristik dari karakteristik *reliability* adalah sebagai berikut:

1) *Maturity*: Kemampuan perangkat lunak untuk menghindari kerusakan ketika terjadi kesalahan.
2) *Fault tolerance*: Kemampuan perangkat lunak untuk mempertahankan performa pada level tertentu saat terjadi kesalahan.
3) *Recoverability*: Kemampuan perangkat lunak untuk mengembalikan performa dan memulihkan data ketika terjadi kesalahan.

Menurut Shanmugam dan Florence (2012, hal.40), pengukuran nilai *reliability* berdasarkan dari jumlah masukan atau *test case* yang dijalankan sistem yang dihitung menggunakan *software reliability models*.Pengukuran nilai *reliability* ini dapat dihitung menggunakan *software reliability model* dari Nelson atau disebut dengan Model Nelson.Hal ini dapat dilakukan dengan simulasi yang memberi inputan

pada *web* untuk melakukan kinerja ekstra.Dari simulasi banyak akses *user*, dapat dilihat apakah sistem dapat mengadaptasi kebutuhan *user* di Program Studi Teknik Elektro FT UMY atau tidak.

3) *Usability*

Usability atau kebergunaan merupakan kemampuan perangkat lunak untuk dipahami, dipelajari, digunakan, dan menarik bagi pengguna ketika digunakan dalam kondisi tertentu. Subkarakteristik kualitas aspek ini yaitu sebagai berikut:

1) *Understandability*: Kemampuan perangkat lunak untuk dipahami oleh *user* apakah cocok dan cara penggunaanya.
2) *Learnability*: Kemampuan perangkat lunak untuk memungkinkan *user* mempelajari aplikasi ini.
3) *Operability*: Kemampuan perangkat lunak yang memungkinkan *user* untuk menjalankan dan mengatur aplikasi tersebut.
4) *Attractiveness*: Kemampuan perangkat lunak untuk menarik bagi pengguna.

4) *Efficiency*

Efficiency atau efisien merupakan kemampuan perangkat lunak dalam memberikan kinerja yang sesuai dan relatif terhadap jumlah sumber daya yang digunakan dalam keadaan tersebut. Subkarakteristik kualitas dari karakteristik ini adalah sebagai berikut:

1) *Time behavior*: Kemampuan perangkat lunak untuk menyediakan respon dan waktu proses yang tepat ketika menjalankan suatu fungsi.
2) *Resource behavior*: Kemampuan perangkat lunak untuk menggunakan sejumlah sumber yang tepat saat perangkat lunak melakukan fungsi dalam kondisi tertentu.

5) *Maintainability*

Maintainability atau kemampuan pemeliharaan merupakan kemampuan perangkat lunak untuk dimodifikasi.Modifikasi ini dapat meliputi koreksi, perbaikan atau adaptasi terhadap perubahan lingkungan, persyaratan, dan spesifikasi fungsional (Al-Qutaish, 2010).Menurut Anne Mette Jonassen Hass (2008, hal.2),Perawatan ini meliputi koreksi kesalahan dan kemungkinan produk untuk dilakukan evolusi atau perubahan kearah yang lebih baik. Subkarakteristik kualitas aspek *maintainability* adalah sebagai berikut:

1) *Analyzability*: Kemampuan perangkat lunak untuk ditemukan kekurangan atau penyebab kesalahan sistem.
2) *Changeability*: Kemampuan perangkat lunak untuk dilakukan modifikasi pada sistem.
3) *Stability*: Kemampuan perangkat lunak untuk menangani efek tak terduga dari modifikasi yang dilakukan.
4) *Testability*: Kemampuan perangkat lunak untuk divalidasi setelah dilakukan modifikasi.

Maintainability dapat diuji pada level komponen (Hass, 2008, hal. 11).Sesuai jurnal Rikard Land (2002, hal. 2), *maintainability* dapat diuji secara operasional yang meliputi aspek *instrumentation*, *consistency*, dan *simplicity*.

6) *Portability*

Portability atau portabilitas adalah kemampuan perangkat lunak untuk ditransfer dari satu ke lingkungan lain (Al-Qutaish, 2010).