# BAB II

# LANDASAN TEORI

## 2.1 Pengembangan Sistem

Didalam pengembangan sistem ini akan menjelaskan tentang defnisi pengembangan sistem, tahapan pengembangan sistem dan konsep siklus pengembangan sistem.

### 2.1.1 Definisi Pengembangan Sistem

Pengembangan sistem (*system development*) dapat berarti menyusun suatu sistem baru untuk menggantikan sistem yang lama secara keseluruhan atau memperbaiki sistem yang telah ada. Sistem yang lama perlu diperbaiki atau diganti (Jogiyanto, 2005).

### 2.1.2 Tahapan Pengembangan Sistem

Proses pengembangan sistem terdiri dari proses standar atau langkah yang dapat digunakan pada semua proyek pengembangan sistem. Meskipun proses bisnis pada masing-masing organisasi berbeda, mereka memiliki karakteristik umum yang sama, yaitu kebanyakan proses pengembangan sistem pada organisasi mengikuti pendekatan *problem-solving*. Berikut ini adalah langkah *problem solving* secara umum:

1. **Analisis Sistem**

   Analisis sistem (*system analysis*) dapat didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya



https://lib.mercubuana.ac.id

dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan perbaikannya (Jogiyanto, 2005).

Tahap analisis sistem dilakukan setelah tahap perencanaan sistem (*system planning*) dan sebelum tahap desain sistem (*system design*). Tahap analisis merupakan tahap yang kritis dan sangat penting, karena kesalahan di dalam tahap ini akan menyebabkan juga kesalahan di tahap selanjutnya (Jogiyanto, 2005). Didalam tahap analisis terdapat langkah-langkah dasar yang harus dilakukan oleh analis sistem sebagai berikut (Jogiyanto, 2005):

a. *Identify,* yaitu mengidentifikasi masalah.
b. *Understand,* yaitu memahami kerja dari sebuah sistem yang ada.
c. *Analyze,* yaitu menganalisis sistem.
d. *Report,* yaitu membuat laporan dari hasil analisa.

UNIVERSITAS MERCU BUANA

**2. Desain Sistem**

Desain sistem dapat diartikan sebagai berikut (Jogiyanto, 2005):

a. Tahap setelah analisis dari siklus pengembangan sistem.
b. Pengidentifikasian dari kebutuhan-kebutuhan fungsional.
c. Persiapan untuk merancang bangunan implementasi.
d. Menggambarkan bagaimana suatu sistem di bentuk.

https://lib.mercubuana.ac.id

e. Penggambaran berupa, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi.
f. Konfigurasi dari komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem.

Desain sistem (*system design*) dapat dibagi dalam dua bagian, yaitu deasin sistem secara umum (*general system design*) dan desain sistem terinci (*detailed system design*). Desain sistem secara umum (*general system design*) disebut juga dengan desain konseptual (*conceptual design*) atau desain logikal (*logical design*) atau desain secara makro (*macro design*). Desain sistem terinci disebut juga dengan desain sistem phisik (*physical system design*) atau desain internal (*internal design*). (Jogiyanto, 2005)

Tujuan dari desain secara umum adalah untuk memberikan gambaran secara umum kepada *user* tentang sistem yang baru. Desain secara umum mengidentifikasikan komponen- komponen sistem informasi yang akan di desain secara rinci. Desain terinci dimaksudkan untuk pemogram komputer dan ahli teknik lainnya yang akan mengimplementasi sistem (Jogiyanto, 2005).

### 3. Seleksi Sistem

Menyeleksi atau memilih teknologi untuk sistem informasi merupakan tugas yang tidak mudah. Tahap seleksi sistem (*system selection*) merupakan tahap

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

untuk memilih perangkat keras dan perangkat lunak untuk sistem informasi (Jogiyanto, 2005).

**4. Implementasi Sistem**

Tahap implementasi sistem (*system implementation*) merupakan tahap meletakkan sistem supaya siap untuk dioperasikan. Tahap ini termasuk juga kegiatan menulis kode program jika tidak digunakan paket perangkat lunak aplikasi dan pengetesan program (Jogiyanto, 2005).

## 2.2 Perancangan Sistem

Perancangan sistem adalah proses penyiapan spesifikasi yang terperinci untuk mengembangkan sistem yang baru. (Ladjamudin, 2005). Spesifikasi tersebut meliputi:

a. Spesifikasi keluaran sistem, yang didalamnya mencakup isi, format, *volume*, frekuensi laporan-laporan dan dokumen-dokumen.
b. Desain semua hal yang penting mengenai langkah-langkah pengolahan, prosedur-prosedur dan pengandalian.
c. Penyiapan rencana implementasi sistem yang baru.

Perancangan sistem ditujukan untuk menghilangkan kekurangan dan meningkatkan kelebihan sistem yang sedang berjalan. (Jogiyanto, 2005). Rancangan sistem dapat diartikan sebagai berikut (Jogiyanto, 2005):

a. Tahap setelah analisis dari siklus pengembangan sistem.
b. Pendefinisian dari kebutuhan-kebutuhan fungsional.
c. Persiapan untuk rancang bangun implementasi.

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

d. Menggambarkan bagaimana suatu sistem dibentuk.

e. Yang dapat berupa penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi.

f. Termasuk menyangkut mengkonfigurasi dari komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem.

Dengan demikian maka dapat disimpulkan bahwa perancangan sistem adalah proses penyiapan spesifikasi yang terperinci untuk menghindari kekurangan dan menigkatkan kelebihan sistem yang sedang berjalan atau belum ada sebelumnya dalam membangun sistem baru.

## 2.3 Pengertian *Helpdesk*

Pada dasarnya *helpdesk* merupakan sebuah *centerpoint* dimana masalah atau *issue* dilaporkan dan dikelola secara terurut dan terkoordinasi. Dari perspektif umum atau luas, *helpdesk* merupakan bagian pelengkap dari sebuah fungsi pelayanan, dan bertanggung jawab sebagai sumber dari pemecahan masalah atau *issue* lainnya. (http://www.help-desk-world.com/help-desk.htm).

Dalam sebuah perusahaan, *helpdesk* adalah sebuah bagian atau departemen tertentu yang melayani atau menanggapi pertanyaan teknis pengguna. *Helpdesk* biasanya digunakan untuk menjawab pertanyaan-pertanyaan atau menangani keluhan teknis dari *client* atau pengguna. Secara umum pertanyaan dan jawaban dapat disampaikan melalui telepon, *email*, *website*, *fax* atau menggunakan aplikasi khusus yang mendukung pekerjaan *helpdesk*.

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

Helpdesk menjadi fokus sentral karena dapat menangkap metrik penting SLA (*Service Level Agreement*) untuk rata-rata waktu, respon dan kepuasan pelanggan atas permintaan layanan IT. Kemudian mampu memberikan analisis prediktif untuk kemungkinan masalah-masalah yang akan terjadi dimasa depan. Dan memungkinkan IT untuk melakukan *resolved bugs* lebih awal, sebelum hal itu menjadi masalah baru dan kompleks.

## 2.4 Keuntungan Penggunaan HelpDesk

Secara umum keuntungan yang diharapkan dari sebuah program atau aplikasi *helpdesk*, antara lain: (http://www.help-deskworld.com/help-desk-htm)

1. Dapat memberikan solusi atas pertanyaan-pertanyaan dalam kurun waktu singkat.
2. Dapat mengecek permasalahan yang ada dan mengatur pembagian tugas kepada para staff.
3. Dapat meningkatkan efisiensi perusahaan dalam menangani pertanyaan dan keluhan yang datang dari *client* atau pengguna.
4. Dapat memberi laporan seputar perkembangan kinerja para *staff* dan pimpinan perusahaan.
5. Dapat menangani pertanyaan dan keluhan yang sejenis dengan lebih cepat karena pertanyaan dan keluhan dicatat.

## 2.5 *Ticketing System*

Dalam aplikasi *helpdesk* setiap keluhan atau permasalahan dapat dilakukan pencatatan dengan menggunakan *ticketing system*, yaitu dengan membuat dan

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

mengimkan *trouble ticket* kepada *staff* yang bertugas menangani keluhan dari client. *Trouble ticket* berisi pertanyaan-pertanyaan ataupun permasalahan-permasalahan dari sisi *client*. *Trouble ticket* pada umumnya dibuat oleh seorang operator atau *front-end* yang nantinya *trouble ticket* tersebut akan diteruskan ke teknisi yang bertanggung jawab atau *back-end* untuk mengatasi permasalahan dalam *ticket* tersebut.

Proses pemecahan masalah *helpdesk* dalam sebuah perusahaan biasanya memiliki prosedur mekanisme tersendiri berdasarkan aturan dan struktur organisasi. Namun biasanya proses penanganan masalah ini dilakukan dalam dua tahapan, yaitu tahapan admin *helpdesk* dan tahapan engineer. Seorang admin *helpdesk* biasanya adalah seorang operator yang bertugas secara terus-menerus untuk melakukan pelayanan dalam menerima, menjawab dan mencatat keluhan dari client. Pada saat sebuah *trouble ticket* sampai di sistem *helpdesk*, biasanya yang membuka *trouble ticket* pertama kali adalah bagian admin *helpdesk*. Apabila permasalahan tidak dapat ditangani, maka selanjutnya *trouble ticket* tersebut akan dialihkan atau diteruskan kepada seorang engineer yang mempunyai kompetensi yang lebih tinggi hingga permasalahan tersebut dapat segera diatasi.

UNIVERSITAS
MERCU BUANA

## 2.6 *Analytical Hierarchy Process* (AHP)

*Analytical Hierarchy Process* (AHP) merupakan suatu model pendukung keputusan yang dikembangkan oleh Thomas L. Saaty. AHP menguraikan masalah multi faktor atau multi kriteria yang kompleks menjadi suatu hirarki. Menurut Saaty (1993), hirarki didefinisikan sebagai suatu representasi dari sebuah permasalahan yang kompleks dalam suatu struktur multi-level dimana level pertama adalah tujuan,

https://lib.mercubuana.ac.id

yang diikuti level faktor, kriteria, sub kriteria, dan seterusnya ke bawah hingga level terakhir dari alternatif.

Dengan hirarki, suatu masalah yang kompleks dapat diuraikan ke dalam kelompok-kelompoknya yang kemudian diatur menjadi suatu bentuk hirarki sehingga permasalahan akan tampak lebih terstruktur dan sistematis.

## 2.7 UML (*Unified Modelling Language*)

Pada bagian sebelumnya, telah dibahas tentang pengembangan *model-driven* yang di dalamnya terdapat pemodelan berorientasi obyek. Teknik pemodelan obyek menyajikan penggunaan metode dan notasi diagram yang sama sekali berbeda dengan teknik lainnya. UML merupakan pemodelan standar berorientasi obyek yang telah dikembangkan oleh Grady Booch, James Rumbaugh, dan Ivar Jacobson (Whitten, 2004). Menurut Jeffrey L. Whitten (2004) UML merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan obyek. UML menawarkan diagram yang dikelompokan menjadi beberapa perspektif berbeda untuk memodelkan suatu sistem; seperti satu set cetak biru (*blueprint*) yang digunakan untuk membangun sebuah rumah (Whitten, 2004). Beberapa diagram UML yang digunakan dalam perancangan *helpdesk ticketing system* ini yaitu: *usecase diagram, activity diagram; class diagram, sequence diagram, comunication diagram,* dan *deployment diagram.*

### 2.7.1 *Usecase Diagram*

*Usecase* adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Usecase* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan (Fowler, 2004). *Usecase diagram* merupakan diagram yang menggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna. Dengan kata lain, secara grafis menggambarkan siapa yang akan menggunakan sistem dan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem (Whitten, 2004). Simbol-simbol yang digunakan dalam *usecase diagram* adalah sebagai berikut:

1. *Usecase*

   Pemodelan *usecase* mengidentifikasi dan menggambarkan fungsi-fungsi sistem dari sudut pandang pengguna eksternal dan dalam sebuah cara dan terminologi yang mereka pahami. *Usecase* merupakan urutan langkah langkah yang secara tindakan saling terkait (*scenario*), baik otomatis maupun secara manual (Whitten, 2004).

2. *Actor* (Pelaku)

   *Actor* merupakan segala sesuatu yang perlu berinteraksi dengan sistem untuk pertukaran informasi. *Actor* dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun (Whitten, 2004).

3. *Relationship* (Hubungan)

   Pada diagram *usecase*, *relationship* digambarkan sebagai sebuah garis antara dua simbol. Pemaknaan *relationship* berbeda-beda tergantung bagaimana garis tersebut digambar dan tipe simbol apa yang digunakan untuk menghubungkan garis tersebut. Berikut adalah perbedaan diantara *relationship* yang ada pada sebuah diagram *usecase* (Whitten, 2004):

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

a. *Extends*

*Extension usecase* merupakan *usecase* yang terdiri dari langkah yang terekstraksi dari *usecase* yang lebih kompleks untuk menyederhanakan masalah dan karena itu memperluas fungsinya.

*b. Includes*

Hubungan *include* menggambarkan bahwa satu *usecase* seluruhnya meliputi fungsionalitas dari *usecase* lainnya.

*c. Depends on*

Hubungan yang memperlihatkan *usecase* mana yang memiliki ketergantungan pada *usecase* lain untuk menetapkan rangkaian *usecase* yang perlu dikembangkan.

### 2.7.2 *Activity Diagram*

*Activity diagram* adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja (Fowler, 2004). *Activity diagram* secara grafis digunakan untuk menggambarkan rangkaian aliran aktifitas baik proses bisnis atau *usecase* (Whitten, 2004).

Diagram ini berbeda dengan *flowchart* dimana diagram ini menyediakan sebuah mekanisme untuk menggambarkan kegiatan yang tampak secara pararel (Munawar, 2005).

### 2.7.3 *Sequence Diagram*

*Sequence diagram* secara grafis menggambarkan bagaimana *object* berinteraksi dengan satu sama lain melalui pesan pada eksekusi sebuah *usecase* atau

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

operasi. Sebuah *sequence* diagram, secara khusus, menjabarkan *behavior* (perilaku) sebuah skenario tunggal (Flower, 2004). Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara *object* dan *sequence* (ruang waktu) (Whitten, 2004).

### 2.7.4 *Class Diagram*

*Class diagram* mendeskripsikan jenis-jenis obyek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka (Fowler, 2004). *Class diagram* merupakan gambar grafis mengenai struktur obyek statis dari suatu sistem, menunjukkan kelas-kelas obyek yang menyusun sebuah sistem dan juga hubungan antara kelas obyek tersebut (Whitten, 2004).

### 2.7.5 *Statechart Diagram*

*Statechart diagram* adalah teknik yang umum digunakan untuk menggambarkan *behavior* sebuah sistem (Fowler, 2004). *State diagram* mengilustrasikan siklus hidup *object* dan keadaan yang dapat diasumsikan oleh *object* dan *events* yang menyebabkan *object* beralih dari satu *state* ke *state* yang lain (Whitten, 2004). Tidak semua obyek yang terdapat pada sistem dibuat *state diagram*-nya, hanya obyek yang memiliki perubahan status yang akan dibuat *state*-nya dan *state diagram* digunakan hanya untuk dokumentasi (Bogs, 2002).

### 2.7.6 *Deployment Diagram*

*Deployment diagram* menunjukkan susunan fisik sebuah sistem (Fowler, 2004). Deployment diagram digunakan untuk memahami susunan fisik sistem dan

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

mengkomunikasikan bagaimana sistem akan dikembangkan kepada pengguna (Bogs, 2002).

Seluruh simbol yang terdapat pada diagram-diagram di atas dapat dilihat pada halaman daftar simbol.

## 2.8 Basis Data (Database)

Basis data (*database*) adalah koleksi terpadu dari data-data yang saling berkaitan dari suatu *enterprise* (perusahaan, instansi pemerintah atau swasta). (Ladjamudin, 2005). Basis data (*database*) terdiri dari data yang akan digunakan atau diperuntukkan terhadap banyak "*user*", dimana masing-masing "*user"* akan menggunakan data tersebut sesuai dengan tugas dan fungsinya, dan "*user"* lain dapat juga menggunakan data tersebut dalam waktu yang bersamaan (Ladjamudin, 2005).

Basis data adalah suatu penyusunan data terstruktur yang disimpan dalam media pengingat (*hardisk*) yang tujuannya adalah agar data tersebut dapat di akses dengan mudah dan cepat. (Kadir, 2003).

Beberapa keuntungan dari basis data terhadap sistem pemrosesan berkas. (Ladjamudin, 2005):

1. Kemubaziran data terkurangi.
2. Penggunaan data lebih mudah.
3. Sekuriti data lebih mudah dilakukan.
4. Berbagi data dapat selalu dilakukan oleh setiap "*user*".

Beberapa kelemahan dari basis data adalah sebagai berikut;

1. *Storage* yang digunakan menjadi besar.
2. Dibutuhkan tenaga yang terampil dalam mengelola data.

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

3. Perangkat lunaknya mahal.
4. Kerusakan pada sistem basis data dapat mempengaruhi departemen yang terkait.
5. Terjadi *deadlock*.

Beberapa Tujuan Basis data adalah sebagai berikut:adalah:

1. Efisiensi meliputi *speed, speace* dan *accurancy*.
2. Menangani data dalam jumlah besar.
3. Kebersamaan pemakai (*Sharebility*).
4. Meniadakan duplikasi dan data yang tidak konsisten.

## 2.9 My SQL

*MySQL* berfungsi untuk mengolah database menggunakan bahasa *SQL*. Hal menarik lainnya adalah *MySQL* juga bersifat *multiplatform*. *MySQL* dapat dijalankan dalam berbagai sistem operasi. Berikut ini beberapa keuntungan menggunakan *MySQL* adalah:

1. Banyak ahli berpendapat bahwa *MySQL* merupakan *server* tercepat.
2. Seperti yang telah dijelaskan di atas bahwa *MySQL* merupakan sistem manajemen *database* yang *Open Source*.
3. *MySQL* mempunyai performa yang tinggi tetapi simpel.
4. *Database MySQL* mengerti bahasa *SQL*.
5. *MySQL* dapat diakses melalui *protocol ODBC (Open Database Connectivity)* buatan *Microsoft* ini menyebabkan *MySQL* dapat diakses oleh banyak *software*.

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

6. Semua klien dapat mengakses *server* dalam satu waktu, tanpa harus menunggu yang lain untuk mengakses *database*.
7. *Database MySQL* dapat diakses dari semua tempat di Internet dengan hak akses tertentu.
8. *MySQL* merupakan *database* yang mampu menyimpan data berkapasitas besar sampai berukuran *Gigabyte*.
9. *MySQL* dapat berjalan di berbagai *operating system* seperti *Linux, Windows, Solaris*, dan lain-lain.

## 2.10 PHP

PHP merupakan bahasa pemrograman berbentuk script yang ditempatkan dalam server dan diproses di server. Hasil dari pengolahan akan dikirimkan ke klien, tempat pemakai menggunakan *browser*. Secara khusus, PHP dirancang untuk membentuk *web* dinamis. Artinya, ia dapat membentuk suatu tampilan berdasarkan permintaan terkini. Misalnya, kita bisa menampilkan isi *database* ke halaman web. Pada prinsipnya, PHP mempunyai fungsi yang sama dengan *script* seperti ASP (Actives Server Page), Cold Fusion, ataupun Perl (Kadir, 2003).

UNIVERSITAS MERCU BUANA

## 2.11 SQL (*Structure Query Language*)

SQL *(Structure Query Language)* adalah bahasa pemrograman standar yang digunakan untuk mengakses *server database*. Sejak tahun 70-an bahasa ini telah dikembangkan oleh IBM, yang kemudian diikuti oleh adanya *Oracle*, *Informix*, dan *Sybase*. Dengan menggunakan SQL, proses akses *database* menjadi *user-friendly*

https://lib.mercubuana.ac.id

dibandingkan pemrograman lainnya seperti *dBase* ataupun *clipper* yang masih menggunakan perintah-perintah pemrograman murni.

Bahasa SQL adalah bahasa yang deklaratif, tidak procedural walaupun ada varian bahasa SQL untuk menulis *stored procedure* yangbersifat *procedural*. Oleh kareanya SQL tidaklah secara *explisit* menudukung deklarasi *variable*, *statement* untuk *looping*, *statement* untuk percabangan dan lain sebagainya. Bahasa SQL tidak bersifat *case-sensitive* tetapi bersifat *high level* dan tidak mengurusi lokasi fisik seperti *offset byte* sebuah *record* atau nama *file* untuk sebuah *table*, melainkan mengijinkan kita untuk memanipulasi *database, table,* baris, dan kolom tanpa mengetahui dimana sebetulnya objek-objek itu berada di *disk* atau dalam format apa disimpannya.

SQL menyediakan beberapa jenis tipe dasar: Boolean, bilangan bulat, bilangan pecahan desimal, text, data biner, tanggal/ waktu. Sebagaimana *data base* sistem yang lain dalam SQL nya juga dikenal hierarki server dengan *data base - data base.* Tiap *database* memiliki tabel-tabel, tiap *table* memiliki *field-field*. Umumnya informasi tersimpan dalam tabel-tabel yang secara logis merupakan struktur-struktur dimensi terdiri atas baris dan kolom. *Field-field* tersebut dapat berupa data seperti *int, real, char, date, time* dan lainnya.

UNIVERSITAS MERCU BUANA

## 2.12 Pengujian

Pengujian adalah proses menjalankan program dengan maksud untuk mencari kesalahan (*error*). Pengujian dikatakan berhasil bila dapat memunculkan kesalahan yang belum diketahui. Jadi, pengujian yang baik bukan untuk memastikan tidak ada kesalahan tetapi untuk mencari sebanyak mungkin kesalahan yang ada di program.

https://lib.mercubuana.ac.id

Pengujian tidak dapat menunjukkan ketidakhadiran *defect*, pengujian hanya menunjukkan bahwa kesalahan perangkat lunak ada. Adapun prinsip dasar pengujian adalah sebagai berikut :

1. Semua pengujian harus dapat dirunut ke *requirement.*
2. Pengujian harus direncanakan jauh sebelum dilakukan.
3. Prinsip Pareto berlaku pada pengujian perangkat lunak yaitu 80 % dari keseluruhan kesalahan yang didapat saat pengujian dapat dirunut ke 20 % dari keseluruhan modul tetapi persoalannya bagaimana menemukan 20% modul tersebut.
4. Pengujian harus dimulai dari lingkup kecil ke lingkup besar.
5. Tidak mungkin ada pengujian yang *exhaustive.*
6. Supaya efektif, pengujian harus dilakukan oleh pihak lain yang independent.

Prosedur pengujian terbagi menjadi dua bagian yaitu:

1. Pengujian *Black Box*

   *Black box testing* memfokuskan pada kebutuhan fungsional dari *software*. Hal ini berarti bahwa pengujian ini memperbolehkan *software engineer* menurunkan sejumlah *input* yang ditujukan untuk menguji kebutuhan fungsional dari program tersebut. Pengujian ini berusaha menemukan *error* dengan kategori sebagai berikut:

   1. Fungsi yang salah atau hilang
   2. Kesalahan antarmuka, struktur data atau pengaksesan data eksternal, unjuk kerja, inisialisasi dan penghentian.

UNIVERSITAS MERCU BUANA

https://lib.mercubuana.ac.id

2. Pengujian *White box/GlassBox*

   Sebuah metoda perancangan kasus uji yang menggunakan struktur kontrol dari perancangan prosedur. Pengguna metoda pengujian akan membuat *software engginer* dapat;

   1. Menjamin bahwa semua jalur independent dalam sebuah modul telah dilewati paling tidak satu kali.
   2. Memeriksa semua keputusan logika baik pada sisi sebenarnya maupun pada sisi salahnya.
   3. Menguji struktur data internal untuk menjamin validasinya.

UNIVERSITAS
MERCU BUANA

https://lib.mercubuana.ac.id