BINUS UNIVERSITY ONLINE LEARNING

People
Innovation
Excellence

LECTURE NOTES

Software Engineering

Minggu 6

Application Testing and Security Engineering

LEARNING OUTCOMES

Setelah menyelesaikan pembelajaran ini, mahasiswa akan mampu:

- ☑ LO3 – Mendemonstrasikan penjaminan mutu suatu *software*

Outline Materi (Sub-Topic):

1. *Testing Conventional Applications*
2. *Testing Object Oriented Applications*
3. *Testing Web Applications*
4. *Testing Mobile Applications*
5. *Security Engineering*
6. Studi Kasus

ISI MATERI

1. *Testing Conventional Applications*

Proses testing ini dapat dilakukan secara manual dengan membuat perencanaan *dan test case* sebelum dilakukan testingnya. Kapan metode ini digunakan, tentunya disesuaikan dengan kebutuhan dan karakteristik dari system.

Misalnya, Anda membuat aplikasi HRD berbasis desktop. Pengujian konventional ini dapat dilakukan dengan membuat test case yang berisi:

- Tahap testing yang dibutuhkan
- Aktivitas test untuk setiap fitur
- Hasil yang diharapkan
- Status, apakah sudah lulus tes atau belum
- Action plan, yaitu jika terjadi error, maka aktivitas apa yang perlu dilakukan.

Contoh tabel di dalam pembuatan suatu *test case*

No	Modul	Tahapan Test	Hasil yang diharapkan	Status	Action Plan
1	Cari data pegawai	1. Login ke system 2. Masuk ke menu Search 3. Masukkan NIM pegawai	Sistem dapat menampilkan data pegawai yang diinginkan		

Pada proses pengujian konvensional, berikut adalah beberapa hal yang perlu diperhatikan:

- *Operability*, dapat dioperasikan dengan baik dan bersih
- *Observability*, hasil dari setiap test case siap untuk diobservasi
- *Controllability*, tingkatan dimana pengujian dapat diotomasi dan dioptimasi
- *Decomposability*, pengujian dapat ditargetkan
- *Simplicity*, mengurangi kompleksitas arsitektur dan logika untuk menyederhanakan pengujian.
- *Stability*, hanya sedikit perubahan yang diminta pada saat proses pengujian berlangsung
- *Understandability*, rancangan sistem yang dapat dipahami

Apa yang disebut sebagai pengujian yang "baik"?

- Sebuah pengujian baik memiliki kemungkinan yang tinggi dalam menemukan error
- Pengujian yang baik tidak redundan
- Pengujian yang baik harus dari "keturunan terbaik"
- Pengujian yang baik tidak boleh terlalu sederhana atau terlalu kompleks.

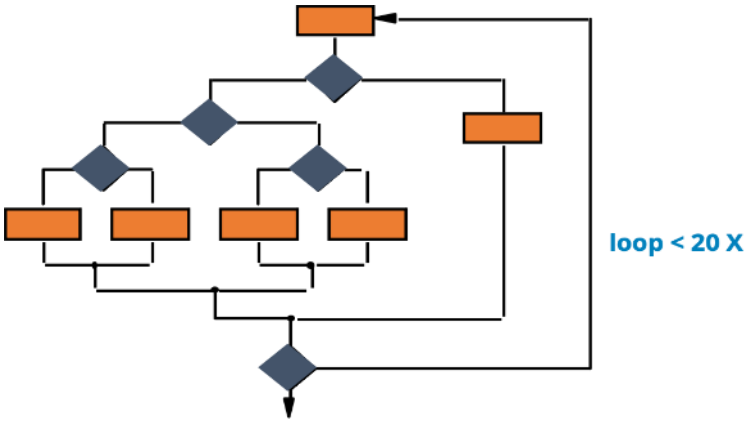
Merancang *test case*:

- Tujuan: untuk menemukan error
- *Criteria*: dengan cara yang lengkap
- Batasan: dengan usaha dan waktu yang minimal

- *Testing convensional* dibagi menjadi dua jenis testing:

a. *Exhaustive Testing*

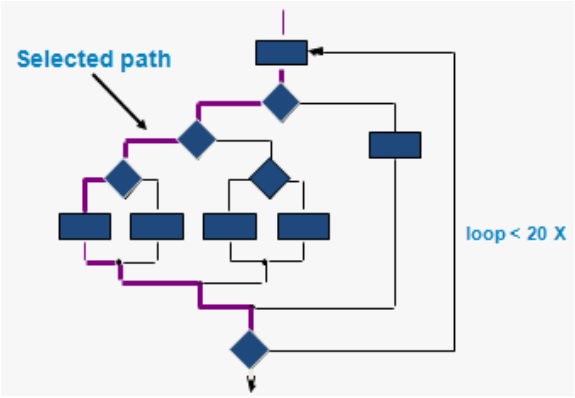
Testing yang mengeksekusi semua kemungkinan path dari logika atau algoritma yang ada di dalam sistem. Pada sebuaha contoh, terdapat kemungkinan 10^{14} path yang dapat diuji, sehingga akan membutuhkan waktu 3.170 tahun untuk melakukan pengujian dengan asumsi satu path dieksekusi selama satu mili second.



b. *Selective Testing*

Selective testing merupakan pengujian yang dilakukan hanya pada path-path yang dipilih. Pemilihan test path dapat mengutamakan logika utama dari fungsionalitas sistem.

Berikut ini adalah contoh *selective testing*:



Sedangkan berdasarkan metode pengujian, terdapat dua kategori testing:

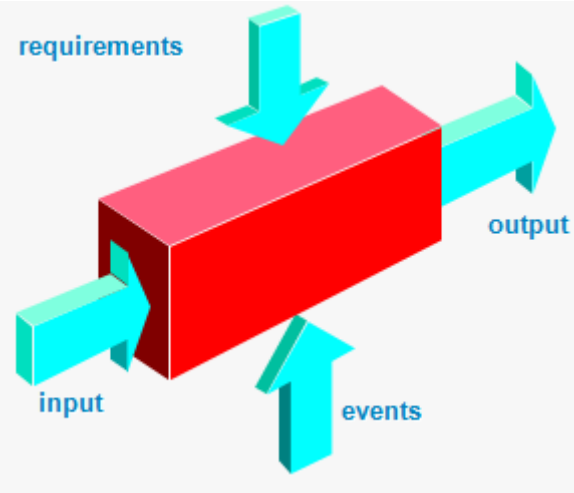
a. Metode *White box*

Tujuan utama dari white box testing adalah untuk memastikan bahwa semua *statement* dan kondisi telah dieksekusi minimal satu kali. Sehingga untuk melakukan *white box testing*, diperlukan *source code* dan dokumentasi yang terkait dengan aplikasi. *Tester* yang biasa melakukan *white box* testing adalah developer/programmer yang mengembangkan sistem.

Pada *white box testing*, biasa dilakukan *basis path testing* dengan menghitung *cyclomatic complexity*.

b. Metode *Black Box*

Metode *black box* merupakan pengujian tanpa melihat isi dari aplikasi. Pengujian dilakukan dengan memberikan input, melakukan sekumpulan *events* dan memberikan requirements sesuai dengan instruksi yang diberikan, kemudian mengecek output yang dihasilkan.



2. *Testing Object Oriented Applications*

Agar dapat menguji aplikasi yang berorientasi obyek dengan layak, tiga hal berikut harus dilakukan:

- Definisi dari pengujian harus diperluas untuk meliputi teknik pencarian error yang dapat diaplikasikan dalam model perancangan dan analisis *object oriented.*
- Strategi dari pengujian integrasi dan uni harus diubah secara significan
- Rancangan dari *use case* harus memfasilitasi karakteristik yang unik dari *software object oriented.*

➢ Terdapat tiga strategi pengujian *object oriented* yang dapat dilakukan:

a. *Unit testing*

Konsep dari unit berubah. Unit terkecil yang dapat diuji adalah kelas yang terenkapsulasi. Satu operasi *single* tidak dapat lagi dioperasi dalam isolasi, tetapi sebagai bagian dari kelas.

b. *Integration testing*

- Pengujian berdasarkan thread mengintegrasikan sekumpulan dari kelas yang dibutuhkan untuk memberikan respon untuk satu *input/event* dari sistem.
- Pengujian berdasarkan "penggunaan" memulai konstruksi dari sistem dengan menguji kelas-kelas yang menggunakan paling sedikit kelar *server*. Setelah kelas yang independen tersebut diuji, lapisan kelas berikutnya disebut sebagai kelas dependen.
- Pengujian *cluster* mendefinisikan cluster dari kelas kolaborasi yang dilakukan dengan merancang *test case* yang berusaha untuk mengungkapkan *error* dalam kolaborasi.

c. *Validation testing*

Detail dari koneksi kelas yang menghilang.

3. Testing *Web Applications*

Pada pengujian aplikasi *web*, terdapat beberapa dimensi kualitasi yang harus diperhatikan:

a. *Content*, dievaluasi pada kedua level *semantic* dan *syntactic*.

b. *Function*, diuji kebenaran, stabilitasnya, dan kesesuaian general hingga standar implementasi yang layak.

c. *Structure*, dinilai untuk memastikan bahwa fungsi dan konten dari aplikasi web disampaikan dengan benar, dapat diperluas, dapat didukung sebagai konten baru atau penambahan fungsionalitas.

d. *Usability*, diuji untuk memastikan bahwa setiap kategori dari user didukung oleh *interface*, serta dapat mempelajari dan mengaplikasikan semua navigasi yang dibutuhkan.

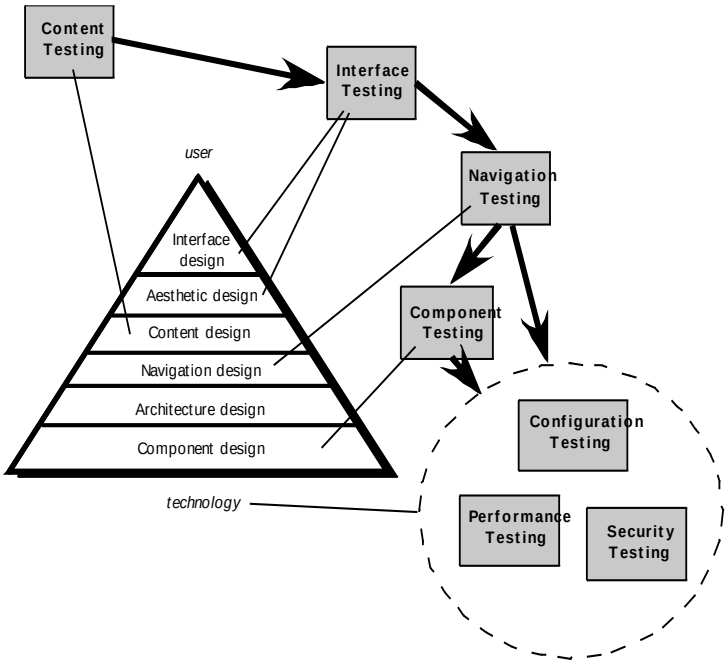
e. *Navigability*, diuji untuk memastikan semua *semantic* dan sintaks navigasi dilaksanakan utnuk menemukan *error* navigasi.

f. *Performance*, pengujian *performance* dilakukan dalam berbagai kondisi operasi, konfigurasi, dan loading untuk memastikan bahwa sistem *responsive* akan interaksi user.

g. *Compatibility*, diuji dengan menjalankan aplikasi *web* dalam berbagai konfigurasi host yang berbeda untuk *client* dan *server*.

h. *Interoperability*, diuji untuk memastikan bahwa aplikasi *web* memiliki interface yang layak sesuai dengan *interface* aplikasi atau/dan *database* lainnya.

i. *Security*, diuji untuk menilai potensi kerentanan dan berusaha untuk mengeksploitasi setiap kerentanan yang ditemukan.



4. *Testing Mobile Applications*

Kriteria lingkungan dan *tool* pengujian aplikasi *mobile*:

- Identifikasi object
- *Security*
- *Devices*
- *Functionality*
- *Emulators and plug ins*
- *connectivity*

5. *Security Engineering*

Bagian penting dalam membangung sistem yang aman adalah mengantisipasi kondisi atau ancaman yang mungkin digunakan untuk merusak sistem atau mengubah sistem sehingga tidak dapat diakses oleh user yang berwenang. Proses ini disebut analisis ancaman.

6. Studi kasus

Beberapa perusahaan memiliki suatu departmen khusus yang dinamakan sebagai QA/QC departmen, yaitu *Quality Assurance* dan *Quality Control*. Tugasnya untuk memastikan semua sistem yang dibangun sudah dilakukan test sehingga layak untuk dilakukan *deployment* ke *production system*, yaitu lingkungan yang terdapat aplikasi yang digunakan untuk kegiatan nyata.

Tahapan awal yang dilakukan adalah membuat perencanaan dari testing, yang mencakup:

- Tujuan testing
- Aplikasi atau system apa yang akan di-test
- Metode testing yang dilakukan
- Jadwal testing
- Penerimaan dari testing tersebut

DAFTAR PUSTAKA

- Software engineering : a practitioners approach : Chapter 23/Pages 496, Chapter 24/Pages 523, Chapter 25/Pages 540, Chapter 26/Pages 567, and Chapter 27/Pages 584