PENGUJIAN PERANGKAT LUNAK

Oleh: Rahmi Hidayati, S.Kom., M.Cs.

Tujuan Pembelajaran

- Mahasiswa mampu menjelaskan prinsip, tujuan, strategi dan tahapan pengujian pada perangkat lunak
- Mahasiswa mampu menjelaskan dan membedakan teknik pengujian white box dan black box

Latar Belakang

- Pengujian perangkat lunak merupakan suatu tahapan penting dalam pembangunan perangkat lunak.
- Pengujian dilakukan dengan cara mengevaluasi konfigurasi perangkat lunak yang terdiri dari spesifikasi kebutuhan, deskripsi perancangan dan program yang dihasilkan.

Latar Belakang

- Hasil evaluasi kemudian dibandingkan dengan hasil uji yang diharapkan.
- Jika ditemukan kesalahan maka perbaikan perangkat lunak harus dilakukan untuk kemudian diuji kembali.

Pengertian Pengujian

- Pengujian perangkat lunak adalah proses menjalankan dan mengevaluasi sebuah PL secara manual maupun otomatis.
- Untuk menguji apakah PL sudah memenuhi persyaratan atau belum dan untuk menentukan perbedaan antara hasil yang diharapkan dengan hasil sebenarnya.

Prinsip Pengujian

1. Dapat dilacak hingga ke persyaratan atau dokumen SKPL/SRS.
2. Pengujian harus direncanakan sejak model dibuat.
3. Pengujian harus dimulai dari hasil yang kecil, diteruskan ke hal-hal yang besar.
4. Tidak bisa semua kemungkinan diuji.
5. Pengujian sebaiknya dilakukan pihak ketiga.

Tujuan Pengujian

1. Menilai apakah perangkat lunak yang dikembangkan telah memenuhi kebutuhan pemakai.
2. Pengujian yang sukses adalah yang berhasil menemukan error yang tersembunyi.
3. Membuat dokumentasi hasil pengujian yang menginformasikan kesesuaian PL yang diuji dengan spesifikasi yang ditentukan.

Tahap Pengujian

1. Tentukan apa yang akan diukur melalui pengujian.
2. Bagaimana pengujian akan dilaksanakan.
3. Membangun suatu kasus uji (*test case*) yaitu sekumpulan data atau situasi yang akan digunakan dalam pengujian.
4. Tentukan hasil yang akan diharapkan atau hasil yang sebenarnya.
5. Jalankan kasus pengujian.
6. Bandingkan hasil pengujian dan hasil yang diharapkan.

Testablility

- Kemudahan untuk diuji.
- Karakteristiknya:
  - *Operability* : mudah digunakan
  - *Observability* : mudah diamati
  - *Controlability* : mudah dikendalikan
  - *Decomposability* : mudah diuraikan
  - *Simplicity* : lingkup kecil, semakin mudah diuji
  - *Stability* : jarang berubah
  - *Understandability* : mudah dipahami

Jenis Pengujian

- Pengujian yang dilakukan meliputi **pengujian unit** (berupa prosedur atau fungsi) dan **pengujian sistem**.
- Dalam pengujian unit, unit-unit yang diuji meliputi unit-unit yang ada dalam sistem, sedangkan pengujian sistem dilakukan terhadap sistem secara keseluruhan.
- Setiap pengujian dilakukan dengan menggunakan berbagai data masukan yang valid maupun tidak.

Teknik Pengujian

- Ada dua teknik pengujian yang dapat digunakan untuk menguji perangkat lunak yaitu :

  1. Teknik *Black Box Testing.*
  2. Teknik *White Box Testing.*

Teknik Pengujian

- *Black Box Testing*
  - Memastikan fungsional PL berjalan.
  - Kesesuaian *input* dengan *output*.
  - Tidak memperhatikan proses *logic internal*.
- *White Box Testing*
  - Pengamatan detail prosedur.
  - Mengamati sampai level percabangan, kondisi dan perulangan.

Pengujian *White Box*

- Teknik ini digunakan untuk mengetahui cara kerja PL secara *internal*.
- Pengujian dilakukan untuk menjamin operasi - operasi *internal* sesuai dengan spesifikasi yang telah ditetapkan dengan menggunakan struktur kendali dari prosedur yang dirancang.

Pengujian *White Box*

- Pelaksanaan pengujian *White Box* : Menjamin seluruh *Independent Path* dieksekusi paling sedikit satu kali.
- ***Independent Path*** adalah jalur dalam program yang menunjukkan paling sedikit satu kumpulan proses atau kondisi baru.
- Mengeksekusi pengulangan dalam batas - batas yang ditentukan.
- Menguji struktur data *internal*.

White Box Testing

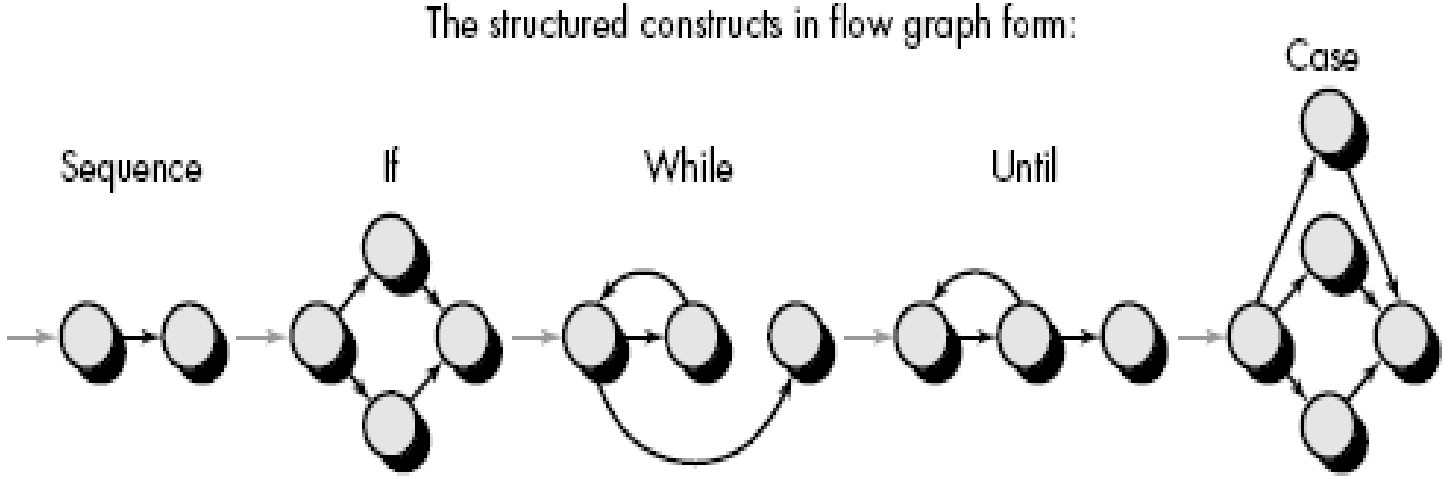
Jenis pengujian *White-Box testing* antara lain:

1. *Basis Path Testing*.
2. *Control Structure Testing*, yang terdiri dari: *Condition Testing, Data Flow Testing* dan *Loop Testing*.

White Box Testing

- *Basic Path Testing*
- Menggunakan notasi *flow graph* yang menggambarkan aliran kontrol logika yang menggunakan notasi pada gambar di bawah ini:

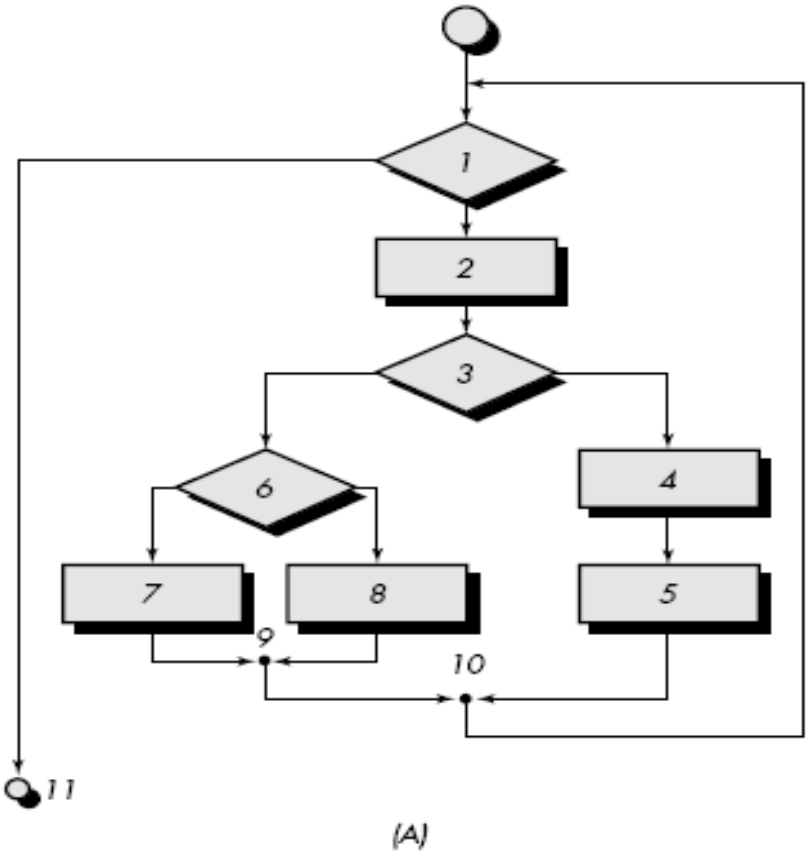
FIGURE 17.1
Flow graph notation



White Box Testing

- Menunjukkan jumlah skenario pengujian yang harus dilakukan untuk menjamin cakupan seluruh program.

FIGURE 17.2
Flowchart, (A) and flow graph (B)



Pengujian *Black Box*

- Pengujian *Black Box* digunakan untuk menguji fungsi - fungsi khusus dari PL yang dirancang.
- Pada teknik ini kebenaran PL yang diuji hanya dilihat berdasarkan keluaran yang dihasilkan dari data atau kondisi masukan yang diberikan untuk fungsi yang ada tanpa melihat bagaimana proses untuk mendapatkan keluaran tersebut.

Pengujian *Black Box*

- Dari keluaran yang dihasilkan, kemampuan program dalam memenuhi kebutuhan pemakai dapat diukur sekaligus dapat diketahui kesalahan - kesalahannya.

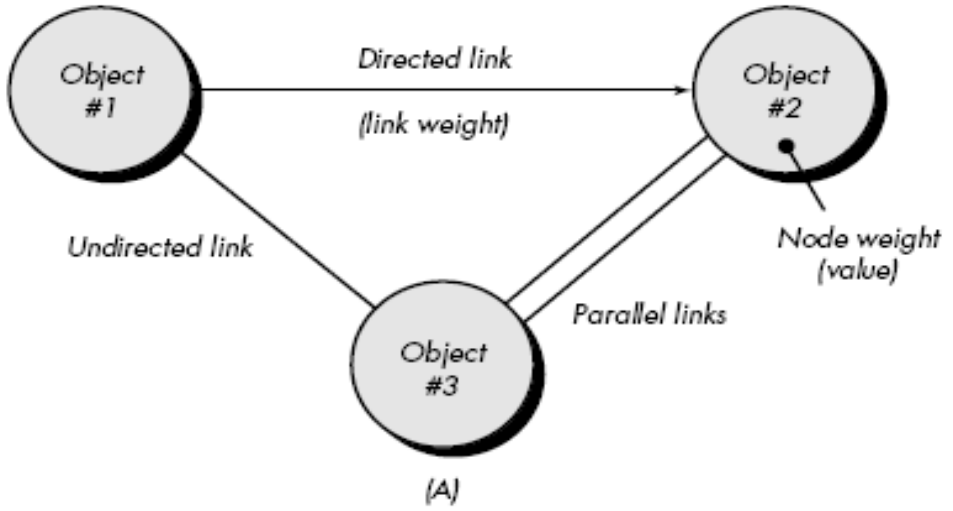
Pengujian *Black Box*

Beberapa jenis kesalahan yang dapat diidentifikasi :

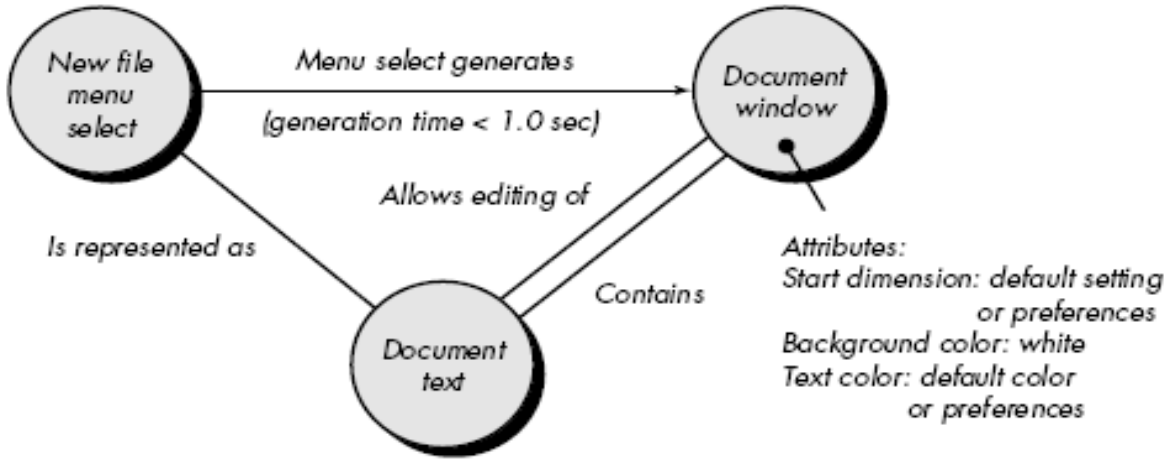
1. Fungsi tidak benar atau hilang.
2. Kesalahan antar muka.
3. Kesalahan pada struktur data (pengaksesan basis data).
4. Kesalahan Inisiasi dan akhir program.
5. Kesalahan performansi.

Black Box Testing – Graph Based

FIGURE 17.9
(A) Graph notation
(B) Simple example



(A)



(B)

Black Box Testing – Equivalence Partitioning

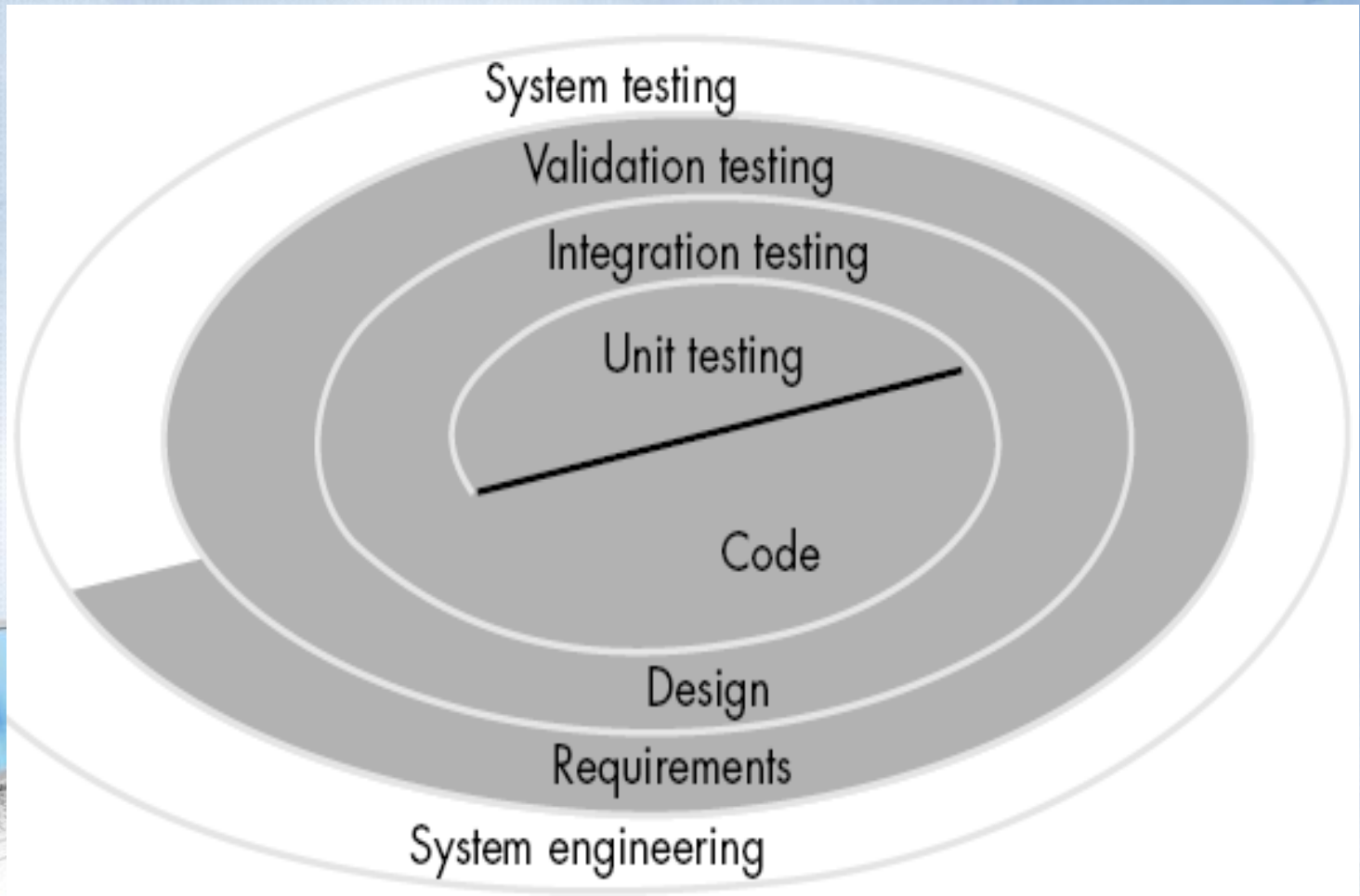
1. If an input condition specifies a *range*, one valid and two invalid equivalence classes are defined.
2. If an input condition requires a specific *value*, one valid and two invalid equivalence classes are defined.
3. If an input condition specifies a member of a *set*, one valid and one invalid equivalence class are defined.
4. If an input condition is *Boolean*, one valid and one invalid class are defined.

- Contoh: Input NPM dalam SIAMIK
  - Jika dikosongi?
  - Jika diisi dengan format yang salah?
  - Jika diisi dengan NPM yang benar?

Black Box Testing – Perbandingan

- Spesifikasi kebutuhan yang sama dimungkinkan menghasilkan aplikasi atau perangkat lunak yang berbeda.
- Skenario pengujian pada aplikasi yang demikian bisa digunakan untuk skenario pengujian aplikasi serupa yang lain.

Strategi Pengujian



Pengujian Unit Program

- Pengujian difokuskan pada unit terkecil dari suatu modul program. Dilaksanakan dengan menggunakan *driver* dan stub.
- *Driver* adalah suatu program utama yang berfungsi mengirim atau menerima data kasus uji dan mencetak hasil dari modul yang diuji.
- Stub adalah sub program untuk menggantikan modul yang merupakan sub ordinat dari modul yang diuji.

Pengujian Unit Program

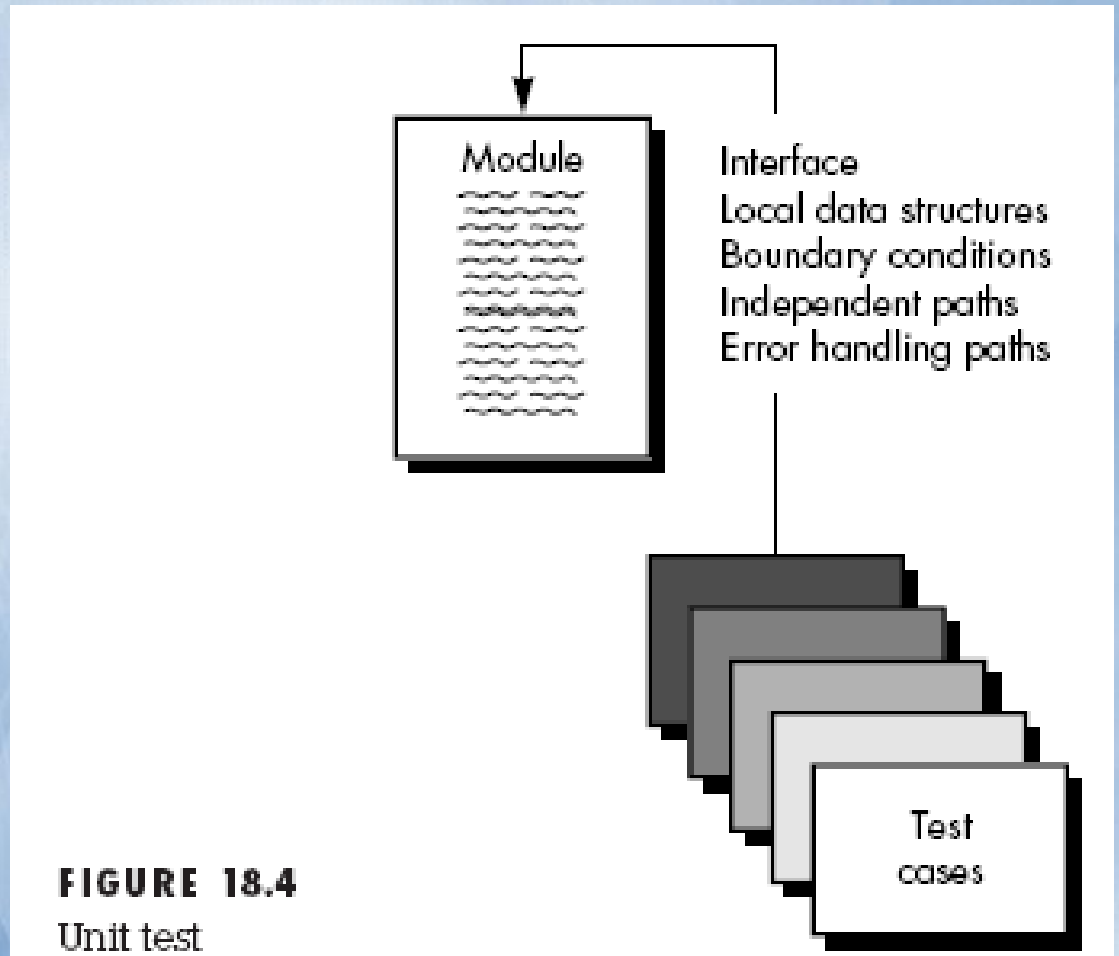


FIGURE 18.4
Unit test

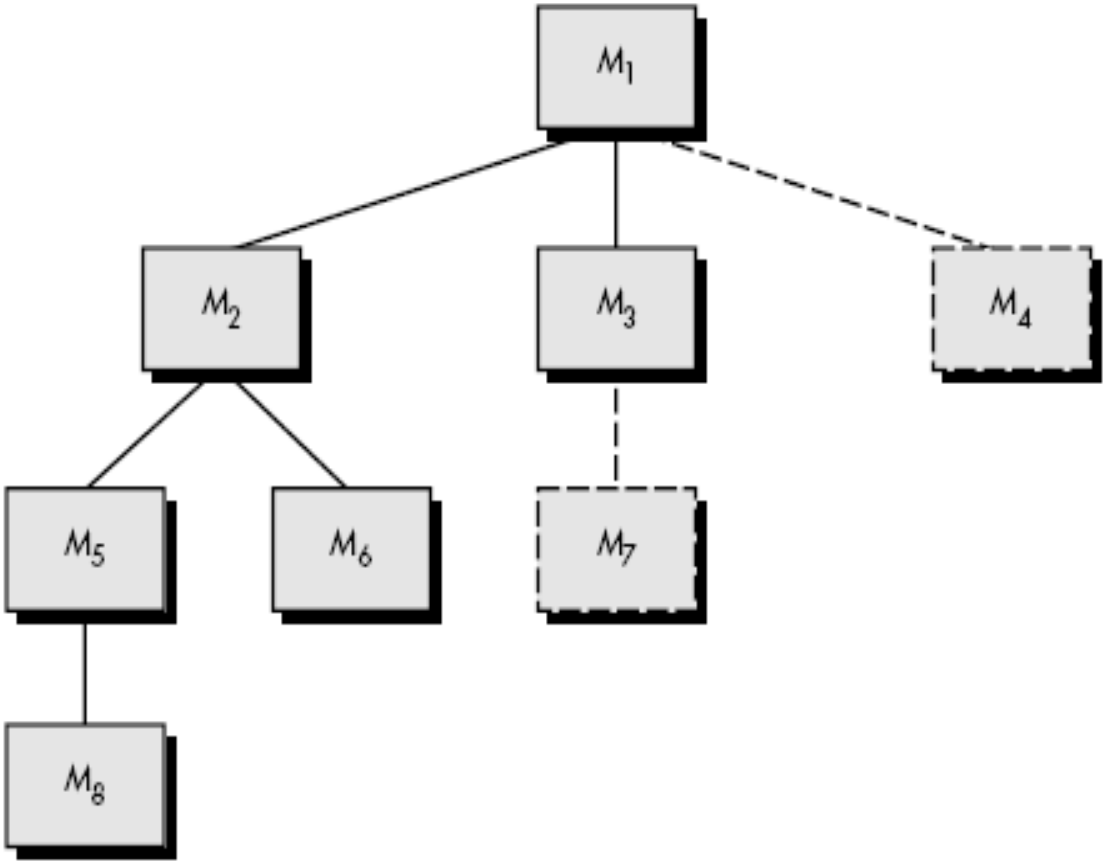
Pengujian Integrasi

- Pengujian terhadap unit - unit program yang saling berhubungan (terintegrasi) dengan fokus pada masalah *interfacing*.
- Dapat dilaksanakan secara *top down integration* atau *bottom up integration*.

Pengujian Integrasi

- *Top – down integration*

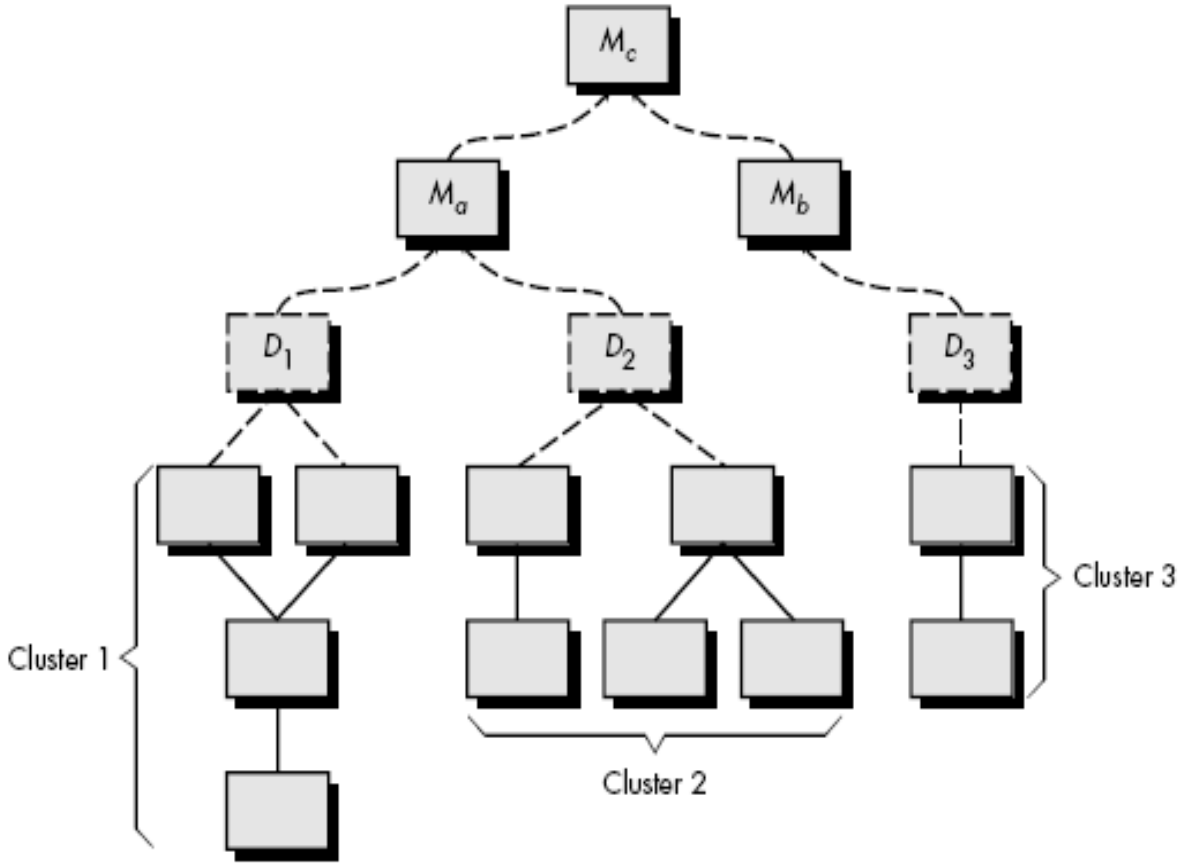
FIGURE 18.6
Top-down integration



Pengujian Integrasi

- *Bottom – up integration*

FIGURE 18.7
Bottom-up integration



Pengujian Validasi

- Pengujian ini dimulai jika pada tahap integrasi tidak ditemukan kesalahan.
- Suatu validasi dikatakan sukses jika PL berfungsi pada suatu cara yang diharapkan oleh pemakai.

Pengujian Validasi

- Disebut sukses jika fungsi PL dapat diterima oleh customer (berdasarkan dokumen SKPL).
- *Alpha test* : dilakukan di tempat *developer* oleh *customer* pada lingkungan yang terkendali.
- *Beta test* : dilakukan di tempat *customer* tanpa melibatkan *developer* pada lingkungan yang tak terkendali.

Pengujian Sistem

- Pengujian yang dilakukan sepenuhnya pada sistem berbasis komputer.
- *Recovery Testing* → pengujian dilakukan dimana sistem diusahakan untuk gagal dengan berbagai cara dan memeriksa apakah proses perbaikan dilakukan dengan tepat.
- *Security Testing* → dilakukan untuk menguji mekanisme proteksi.

Pengujian Sistem

- *Stress Testing* → pengujian yang dirancang untuk menghadapi suatu PL pada situasi yang tidak normal.

Debugging

- *Debugging* bukan merupakan pengujian, namun merupakan konsekuensi dari pengujian yang berhasil.
- Jika sebuah kasus uji berhasil menemukan kesalahan, maka proses *debugging* bertujuan untuk menghilangkan kesalahan tersebut.

Latihan

- Sebutkan tahapan dalam pengujian perangkat lunak.
- Apa fungsi whitebox dan blackbox pada pengujian perangkat lunak.
- Sebutkan teknik umum yang digunakan untuk pengujian perangkat lunak.
- Jelaskan teknik-teknik yang terdapat pad pengujian sistem.
- Pengertian dari independent path.

TERIMA KASIH