UNIVERSITAS IPWIJA

Pengujian Perangkat Lunak

Pengujian (Testing)

Testing adalah proses eksekusi suatu program dengan tujuan untuk menemukan error (Berad, 1994).

Merupakan proses kritis untuk menjamin kualitas perangkat lunak dan merepresentasikan spesifikasi, desain dan pengkodean.

Pengujian sebaiknya menemukan kesalahan yang tidak disengaja dan pengujian dinyatakan sukses jika berhasil memperbaiki kesalahan tersebut.

Pengujian bertujuan untuk menunjukan kesesuaian fungsi-fungsi perangkat lunak dengan spesifikasi.

Pertanyaan dasar yang akan dijawab pada software testing

- Misi dari pengujian
  Mengapa anda melakukan testing? Apa yang ingin anda pelajari?

- Pemilihan strategi
  Bagaimana anda mengoperasikan pengujian anda untuk meraih misi anda?

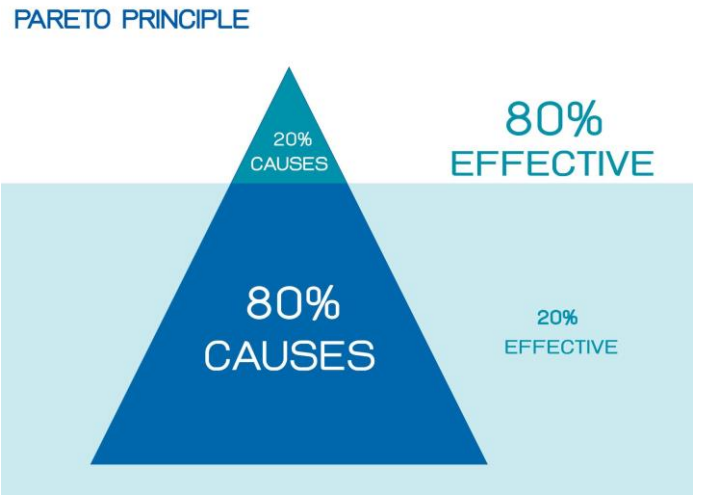
- Oracel/sematic/arti
  Bagaimana anda mengetahui bahwa program itu lolos testing atau gagal?

- Adanya kemungkinan testing tidak sempurna
  Apa yang harus dilakukan untuk menyelesaikan testing secara lengkap?

- Mengukur permasalahan
  Sejauh mana testing dianggap cukup?

Prinsip dasar pengujian (sebagai dasar membuat test case)

- Semua pengujian harus ditelusuri sampai ke persayaratan yang diinginkan oleh pelanggan.
- Pengujian harus direncanakan jauh sebelum pengujian itu dilakukan.
- Prinsip pareto berlaku untuk pengujian perangkat lunak, maksudnya 80% kesalahan yang ditemukan selama pengujian, dapat ditelusuri sampai 20% dari semua modul program.
- Pengujian harus dimulai “dari yang kecil” dan berkembang menjadi pengujian “yang besar”.
- Agar lebih efektif, pengujian harus dilakukan oleh pihak ketiga yang independent.



Tahapan dalam pengujian PL

1. **Tentukan sasaran pengujian**

a. Test case yang baik adalah test case yang memiliki profitabilitas tinggi untuk menemukan kesalahan yang belum pernah ditemukan sebelumnya.

b. Pengujian yang sukses adalah pengujian yang mengungkapkan semua kesalahan yang belum pernah ditemukan sebelumnya

Tahapan dalam pengujian PL

2. Tentukan karakteristik pengujian

a. Testing dimulai dari level modul dan kemudian testing ke arah intergrasi pada sistem berbasis komputer.

b. Lakukan teknik testing yang berbeda sesuai dengan poin-poin yang berbeda pada waktunya.

c. Testing dilakukan oleh *Software developer.* Untuk proyek yang besar dilakukan oleh *group testing* yang independent.

d. Testing dan *debugging* adalah aktivitas yang berbeda tetapi debugging harus diakomodasi pada setiap strategi testing.

Tahapan dalam pengujian PL

3. Tentukan testabilitas software

a. Seberapa mudah sebuah program komputer dapat diuji.

b. Karena pengujian sangat sulit, perlu diketahui apa yang dapat dilakukan untuk dapat membuat lebih mudah.

Kualitas / kriteria software yang baik

- ***Operability***

  Semakin baik software itu berkerja, semakin efisien ia dapat diuji.

- ***Observability***

  Apa yang anda lihat adalah apa yang anda uji.

- ***Controllability***

  Semakin baik kita mengontrol software, semakin banyak pengujian yang dapat di otomatisasi dan dioptimalkan.

- ***Decomposability***

  Dengan mengontrol ruang lingkup pengujian, kita dapat lebih cepat mengisolasi masalah dan melakukan pengujian kembali secara lebih detail.

Kualitas / kriteria software yang baik

- ***Simplicity***

  Semakin sedikit yang diuji, semakin cepat pengujiannya

- ***Stability***

  Semakin sedikit perubahan, semakin sedikit gangguan dalam pengujian

- ***Kemampuan dan mudah dipahami***

  Semakin banyak informasi yang dimiliki semakin detail pengujiannya

Atribut pengujian yang baik

- Memiliki probalitas yang tinggi untuk menemukan error.
- Pengujian yang baik tidak redundant/ganda.
- Pengujian yang baik tidak terlalu sederhana namun juga tidak terlalu kompleks.

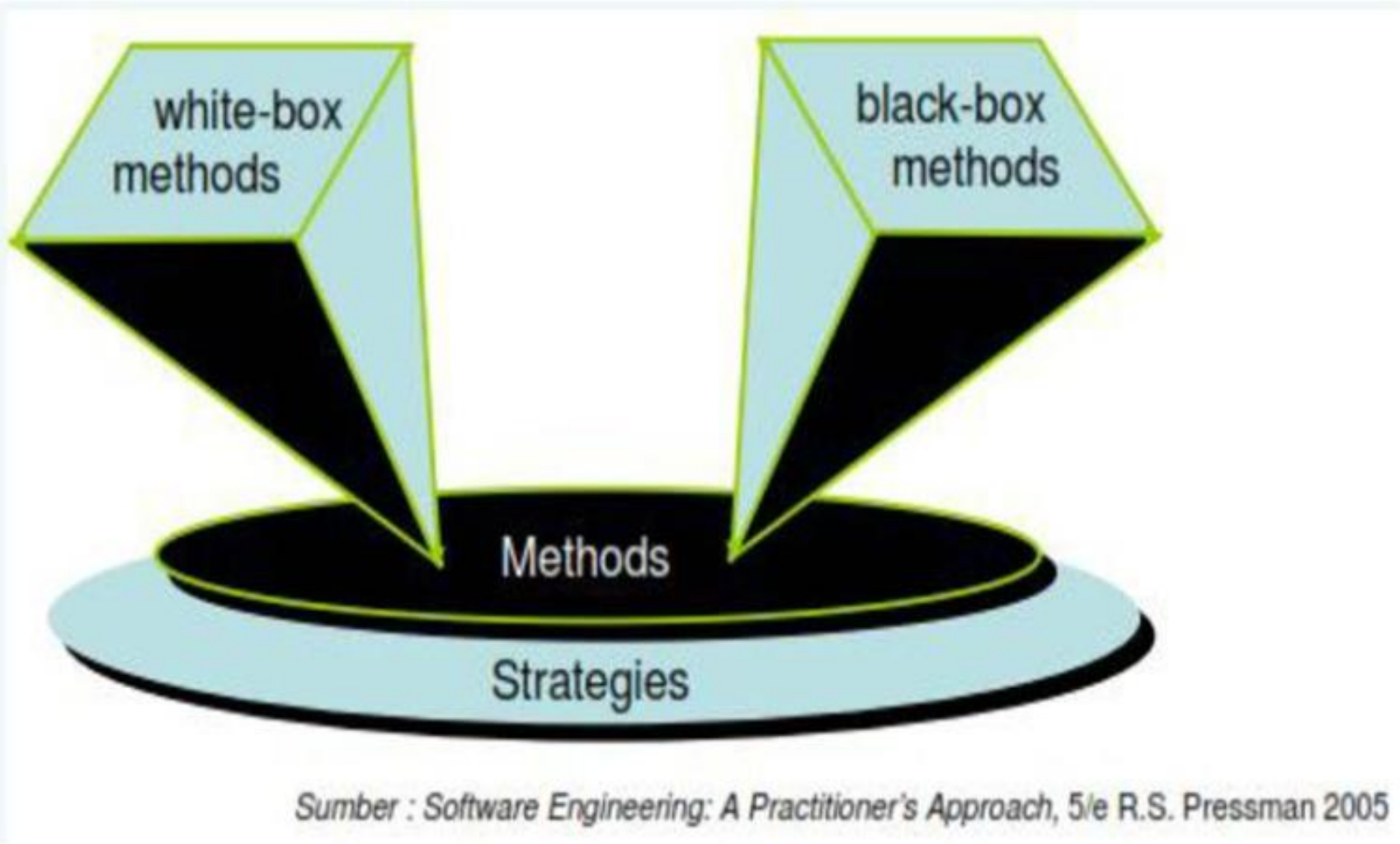
Test case design

Pengetahuan tentang ***fungsi*** dari produk yang telah dirancang untuk diperlihatkan. Test dapat dilakukan dengan menilai masing-masing fungsi apakah telah berjalan sebagaimana yang diharapkan.

Pengetahuan tentang ***cara kerja*** dari produk.

Test dapat dilakukan dengan memperlihatkan cara kerja dari produk secara rinci sesuai dengan spesfikasinya.

Pendekatan dalam testing



Sumber : Software Engineering: A Practitioner's Approach, 5/e R.S. Pressman 2005

Pendekatan dalam testing

Black Box Testing

Test case ini bertujuan untuk menunjukan fungsi PL tentang cara beroperasinya apakah input data dan output telah berjalan sebagaimana yang diharapkan dan apakah informasi yang disimpan secara eksternal selalu dijaga kemutakhirannya.

White Box Testing

Adalah meramalkan cara kerja perangkat lunak secara rinci.
Karenanya logical path (jalur logika) dari PL akan ditest dengan menyediakan test case yang akan mengerjakan sekumpulan kondisi dan atau pengulangan secara intensif sesuai spesifikasi.
Secara sekilas dapat diambil kesimpulan white box testing merupakan tindakan untuk mendapatkan program yang benar secara 100%.

Pengujian Black-Box

Pengujian black-box berfokus pada persyaratan fungsional PL.

Tujuan metode ini mencari kesalahan pada :

- Fungsi yang salah atau hilang
- Kesalahan pada interface
- Kesalahan pada struktur data atau akses database
- Kesalahan performasi
- Kesalahan inisialisasi dan tujun akhir

Pengujian White-Box

Uji coba white box adalah metode perancangan test case yang menggunakan struktur control dari perancangan procedural untuk mendapatkan test case. Dengan menggunakan metode white box analisa sistem akan dapat memperoleh test case yang :

- Menjamin seluruh independent-path di dalam modul yang dikerjakan sekurang-kurangnya satu kali.
- Mengerjakan seluruh keputusan logika.
- Mengerjakan seluruh loop yang sesuai dengan batasnnya.
- Mengerjakan seluruh struktur data internal yang menjamin validitas.

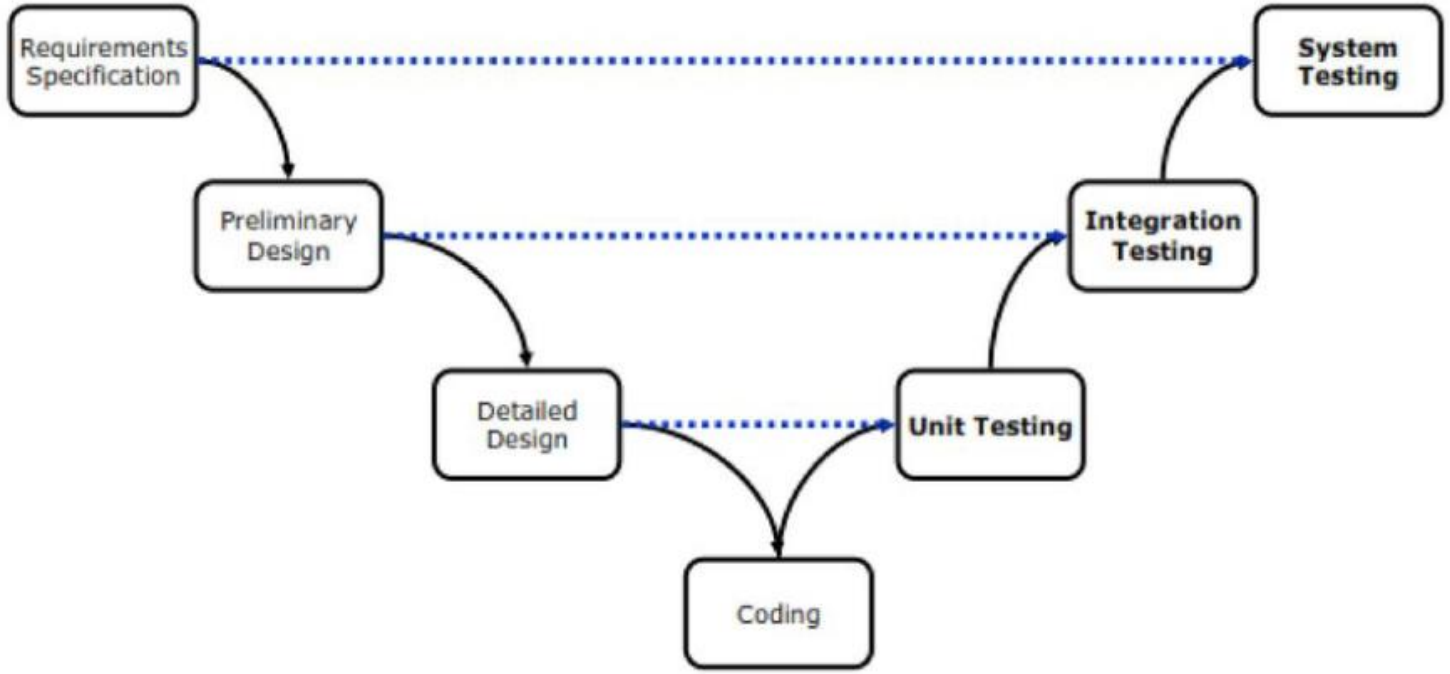
Strategi pengujian perangkat lunak

- **Big Bang**

  Pengujian PL secara keseluruhan, setelah seluruh komponen PL selesai dibuat

- **Incremental**

  Pengujian Secara bertahap.



Rencana pengujian

- **Proses Testing**

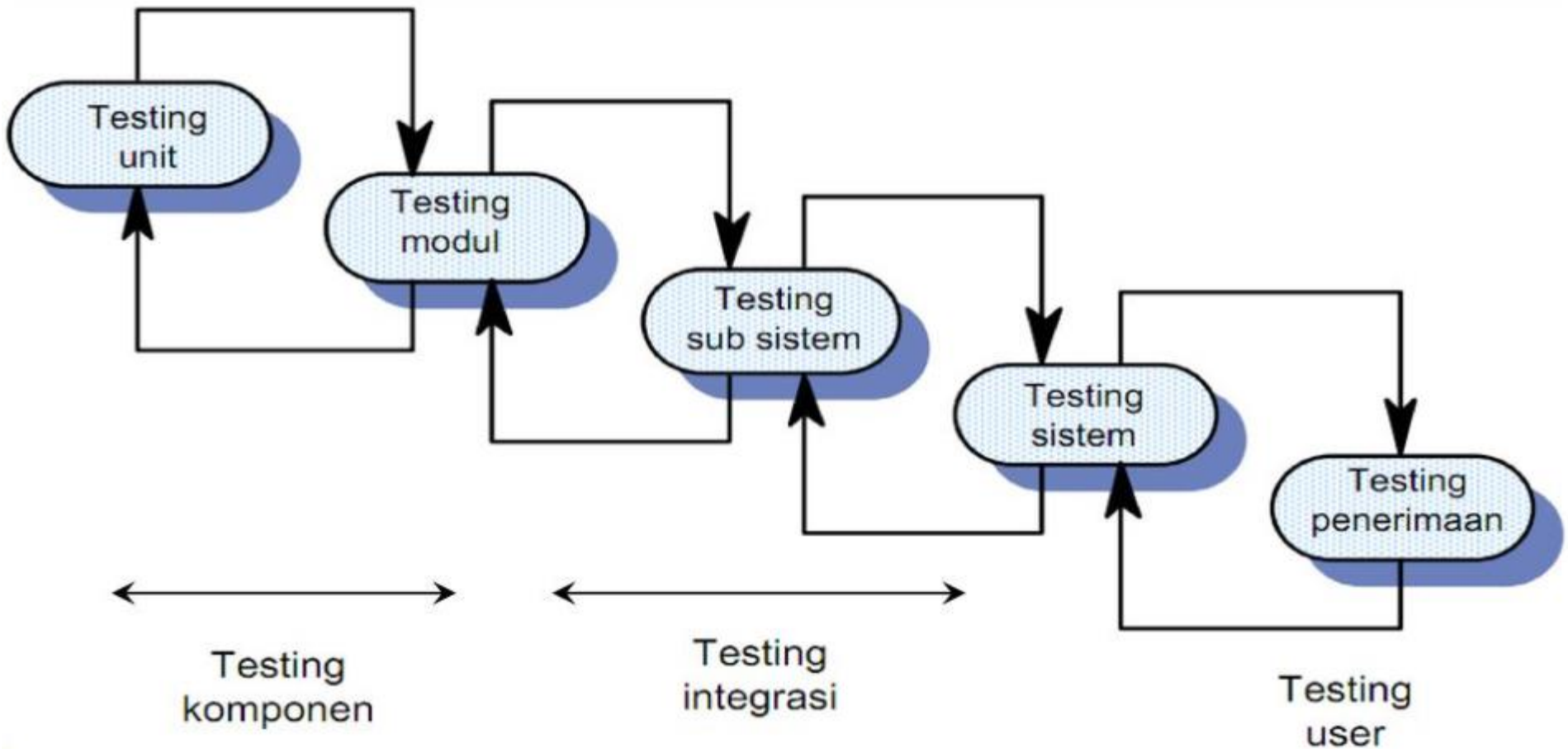
  Deskripsi fase utama dalam pengujian

- **Pelacakan Kebutuhan**

  Semua Kebutuhan user diuji secara individu

- **Item yang diuji**
  - Spesifikasi komponen sistem yang diuji
  - Jadwal testing
  - Prosedur pencatatan hasil dan prosedur
  - Kebutuhan akan hardware dan software
  - Kendala: kekurangan staff, alat, waktu, dll.

Proses testing



Proses testing

- **Unit Testing**

Pengujian masing-masing komponen program untuk meyakinkan bahwa sudah beroperasi secara benar.

- **Modul Testing**

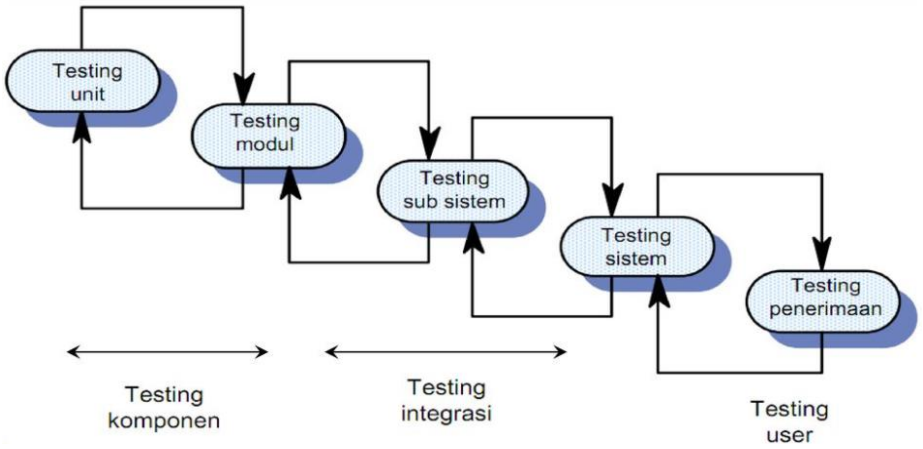
Pengujian terhadap modul-modul komponen yang saling berhubungan.

- **Sub-system Testing**

Pengujian terhadap modul-modul yang membentuk suatu sub-system (aplikasi).

- **System Testing**

Pengujian terhadap intergrasi sub-system, yaitu keterhubungan antara masing-masing sub-system.

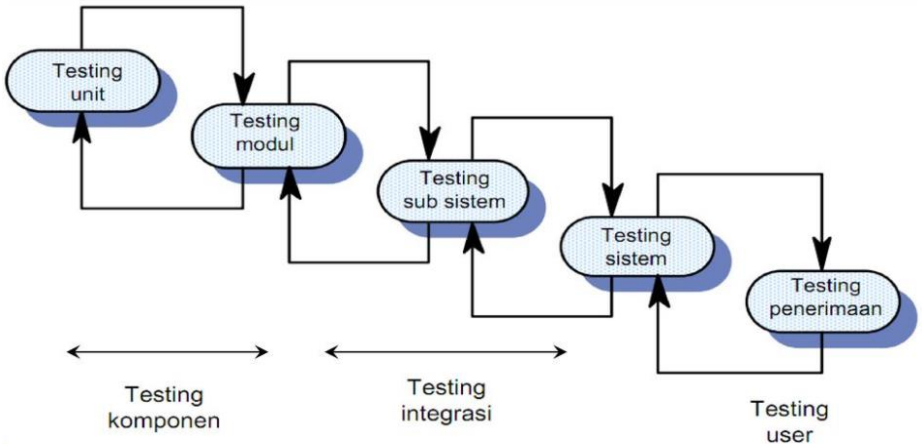


Proses testing

- **Acceptance Testing**

Pengujian terakhir sebelum sistem dipakai oleh user. Melibatkan pengujian dengan data dari pengguna sistem.

Bisa dikenal sebagai “alpha test” (“beta test” untuk software komersial, dimana pengujian dilakukan oleh potensial customer).



Proses testing

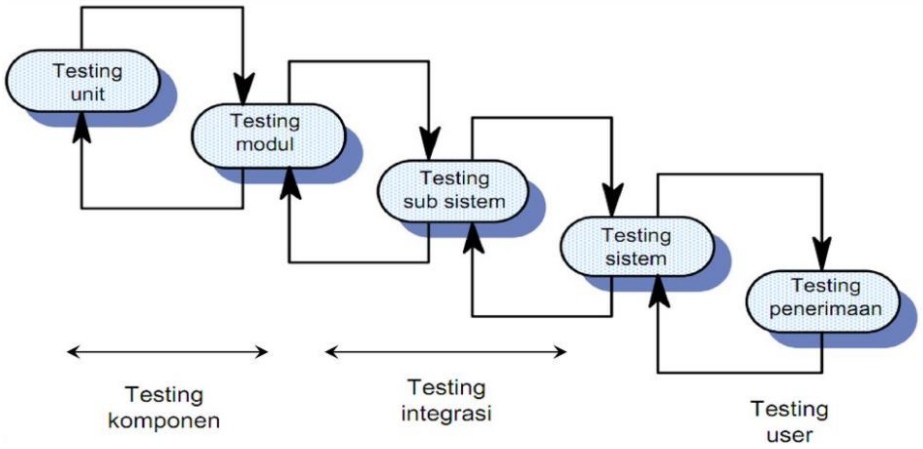
✓ **Component testing**

Pengujian komponen-komponen program biasanya dilakukan oleh komponen developer (kecuali untuk critical-system).

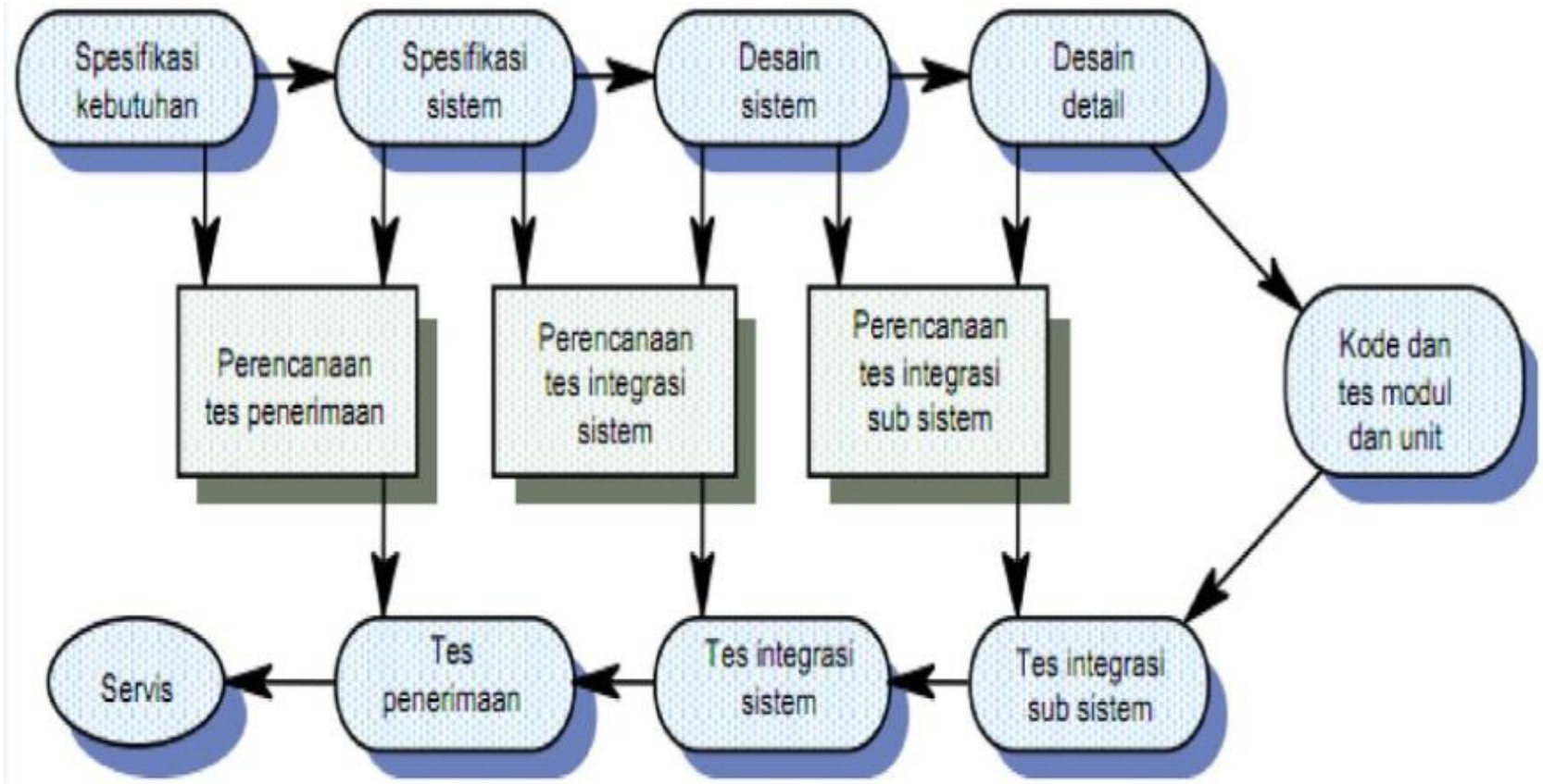
✓ **Intergration testing**

Pengujian terhadap komponen yang terintergrasi untuk membentuk sub-system ataupun system utama.

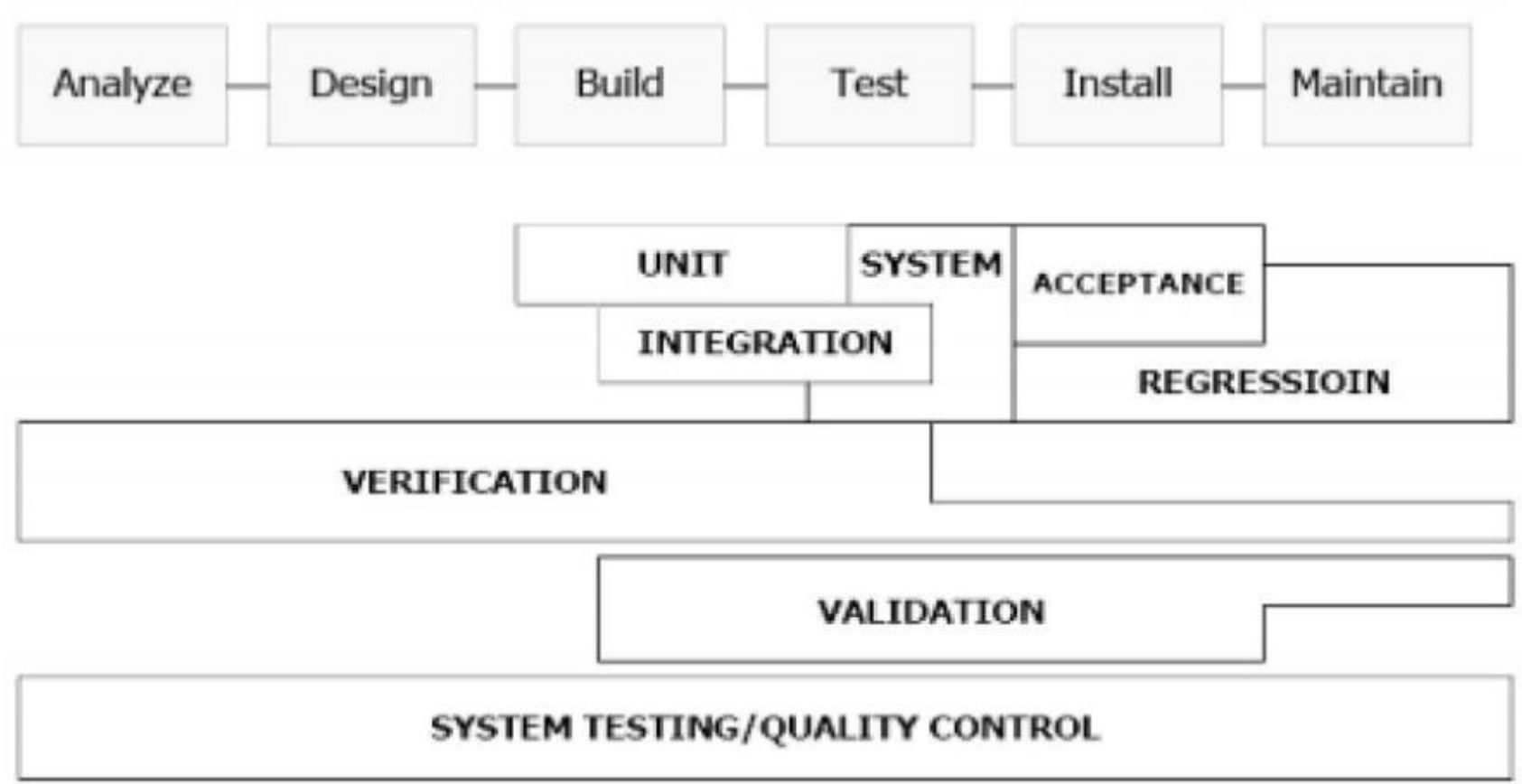
Dilakukan oleh tim penguji yang independent, pengujian berdasarkan spesifikasi system.



Hubungan antara rencana pengujian dan proses pengembangan sistem



Testing lifecycle



Testing lifecycle

✓ **Verifikasi**

Adalah proses evaluasi sebuah sistem atau komponen untuk **mendefinisikan bahwa produk memiliki fase pengembangan yang benar** dimulai dari awal fase. Verifikasi dilakukan berbasis proses.

✓ **Validasi**

Adalah proses evaluasi sebuah sistem atau komponen selama atau pada akhir pengembangan untuk **mendefinisikan bahwa produk sesuai dengan spesifikasi kebutuhan.**

Apa yang diuji ?

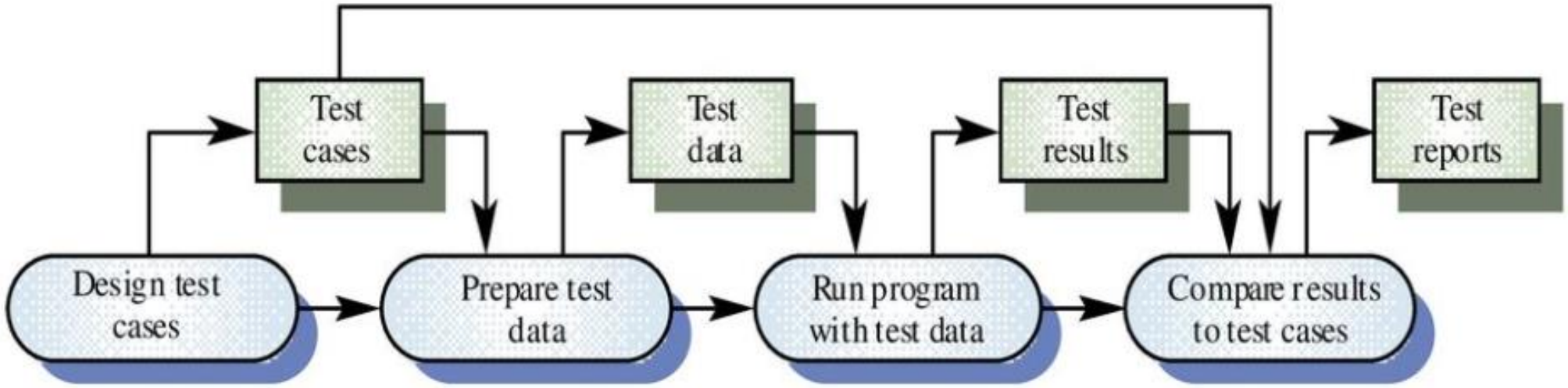
- **Test data**

  Input yang direncanakan digunakan oleh system.

- **Test case**

  Input yang digunakan untuk menguji sistem dan memprediksi output dari input jika sistem beroperasi sesuai dengan spesifikasi.

Teknik dalam pengujian



Failures & Faults

1. ***Failures*** adalah output yang tidak benar/tidak sesuai ketika sistem dijalankan.
2. ***Faults*** adalah kesalahan dalam source code yang mungkin menimbulkan failure ketika code yang fault tersebut dijalankan.

Failure Class	Deskripsi
Transient	Muncul untuk di input tertentu
Permanent	Muncul untuk semua input
Recoverable	Sistem dapat memperbaiki secara otomatis
Unrecoverable	Sistem tidak dapat memperbaiki secara otomatis
Non-corrupting	Failure tidak merusak data
Corruting	Failure yang merusak sistem data

SELESAI