Pertemuan 13

Pengujian (testing)

COPYRIGHT (C) Sept 2012

Langkah terakhir sebelum perangkat lunak kita serahkan kepada pengguna adalah melakukan pengujian (*testing)* yang bertujuan menemukan serta menghilangkan “bug” (kesalahan-kesalahan) yang ada dalam sistem.

Kesalahan tersebut diakibatkan beberapa hal, antara lain:

1.Kesalahan saat penentuan spesifikasi sistem/perangkat lunak

2. Kesalahan saat melakukan analisis permasalahan

3. Kesalahan saat perancangan

4. Kesalahan saat implementasi

COPYRIGHT (C) Sept 2012

Strategi Pengujian

[Ali Bahrawi, 1999]

- Black-Box Testing
- White-Box Testing
- Top-Down Testing
- Bottom-Up Testing

COPYRIGHT (C) Sept 2012

Strategi Pengujian
(lanjutan)

- Black-Box Testing

  Pada pengujian ini kita tidak perlu tahu apa yang sesungguhnya terjadi dalam sistem/perangkat lunak. Yang kita uji adalah masukan serta keluarannya. Artinya dengan berbagai masukan yang kita berikan, apakah sistem/perangkat lunak memberikan keluaran seperti yang kita harapkan.

  Dalam pengujian ini kita dapat menggunakan *use case diagram*.

COPYRIGHT (C) Sept 2012

Contoh Black-Box Testing

Input/event	Proses	Output	Hasil
Klik Menu Beranda	link beranda.html	Tampil beranda diframe utama	Sesuai
Klik Menu Artikel	link artikel.php	Tampil artikel diframe utama	Sesuai

COPYRIGHT (C) Sept 2012

Strategi Pengujian
(lanjutan)

- White-Box Testing

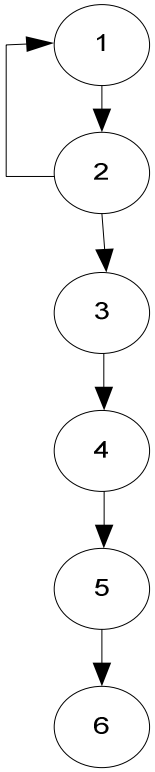
  Pengujian jenis ini mengasumsikan bahwa spesifikasi logika adalah penting dan perlu dilakukan pengujian untuk menjamin apakah sistem/perangkat lunak berfungsi dengan baik. Tujuan utama dari strategi pengujian ini adalah pengujian berbasis kesalahan.

  Beberapa cara yang dapat dilakukan untuk pengujian white-box:

1. Memeriksa semua fungsi dalam setiap objek dengan cara mengeksekusi satu persatu
2. Memeriksa sebagian fungsi dari objek yang ditentukan

COPYRIGHT (C) Sept 2012

White-Box Testing



Gambar disamping menunjukkan contoh simpul yang di ujikan dari sebuah coding program.

Kompleksitas Siklomatis (pengukuran kuantitatif terhadap kompleksitas logis suatu program) dari grafik alir dapat diperoleh dengan perhitungan:

$$V(G) = E – N + 2$$

Dimana:

E = Jumlah *edge* grafik alir yang ditandakan dengan gambar panah

N = Jumlah simpul grafik alir yang ditandakan dengan gambar lingkaran

COPYRIGHT (C) Sept 2012

White-Box Testing

Sehingga kompleksitas siklomatisnya

V(G) = 6 – 6 + 2 = 2

Basis set yang dihasilkan dari jalur independent secara linier adalah jalur sebagai

berikut:

1 – 2 – 3 – 4 – 5 – 6

1 – 2 – 1 – 2 – 3 – 4 – 5 – 6

Ketika aplikasi dijalankan, maka terlihat bahwa salah satu basis set yang dihasilkan adalah 1 – 2 – 3 – 4 – 5 – 6 dan terlihat bahwa simpul telah dieksekusi satu kali. Berdasarkan ketentuan tersebut dari segi kelayakan *software*, sistem ini telah memenuhi syarat.

COPYRIGHT (C) Sept 2012

Strategi Pengujian
(lanjutan)

- Top-Down Testing

  Pengujian ini berasumsi bahwa logika utama atau interaksi antarobjek perlu diuji lebih lanjut. Strategi ini seringkali dapat mendeteksi kesalahan yang serius. Pendekatan ini sesuai dengan strategi pengujian berbasis skenario. Misalnya, pengujian antarmuka pengguna dengan strategi atas-bawah berarti menguji dari layar utama hingga layar dengan peringkat tingkat rendah dan memastikan semuanya berfungsi dengan baik serta sesuai dengan harapan dan kebutuhan pengguna.

COPYRIGHT (C) Sept 2012

Strategi Pegujian
(lanjutan)

- Bottom-Up Testing

  Strategi pengujian ini dimulai dengan rincian sistem kemudian beranjak keperingkat yang lebih tinggi.

  Dalam metodologi berorientasi objek, kita mulai dengan menguji metode-metode dalam kelas, menguji kelas-kelas serta interaksi antar kelas dan selanjutnya hingga pada peringkat yang paling tinggi.

COPYRIGHT (C) Sept 2012

Kualitas

American Heritage Dictionary (dikutip dari buku **Software Engineering : A Practitioner's Approach** karya Roger Pressman, 1997) mendefinisikan kata kualitas sebagai "sebuah karakteristik atau atribut dari sesuatu". Sebagai atribut dari sesuatu, kualitas mengacu pada karakteristik yang terukur, sesuatu yang dapat kita bandingkan dengan standar lain yang sudah kita ketahui seperti panjang, warna, volume dll.

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

Steve McConnell dalam buku yang berjudul CODE COMPLETE : *A Practical Handbook of Software Construction, 1944* mengusulkan bahwa sistem/perangkat lunak seharusnya memiliki karakteristik-karakteristik luar dan dalam sebagai berikut:

1.Karakteristik Luar

Meliputi karakteristik-karakteristik yang dapat diamati secara langsung oleh pengguna, seperti:

- Ketepatan

  Tingkat kebebasan sistem dari kesalahan analisis, perancangan serta implementasi. Ketepatan disini maksudnya adalah sejauh mana sistem yang kita kembangkan memenuhi harapan dan kebutuhan pengguna

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

- Daya Guna

  Tingkat kemudahan pengguna untuk belajar serta menggunakan sistem. Karena pengguna bukanlah orang yang mau belajar tentang komputer tetapi hanya menggunakan sistem untuk keperluan tertentu, maka sistem/perangkat lunak yang dikembangkan harus bisa digunakan oleh semua pengguna.

- Efisiensi

  Pemanfaatan seminimal mungkin dari sumberdaya yang ada baik penggunaan memori, sumber daya CPU, ruang harddisk termasuk juga spesifikasi hardware yang tidak terlalu tinggi.

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

- Keandalan

  Mencakup kemampuan sistem/perangkat lunak untuk menjalankan semua fungsi yang dibutuhkan dan diharapkan

- Integritas

  Tingkat kemampuan sistem/perangkat lunak dalam mencegah akses yang tidak sah atau tidak pada tempatnya ke program atau data didalamnya.

- Kemampuan Adaptasi

  Bagaimana sistem/perangkat lunak dapat dipakai dalam aplikasi atau lingkungan yang lain dari lingkungan saat sistem/perangkat lunak itu diciptakan

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

- Keakuratan

  Tingkat kebebasan sistem/perangkat lunak dari kesalahan-kesalahan (*bugs*), khususnya ditinjau dari keluaran yang dihasilkan.

- Kekuatan

  Tingkat kemampuan sistem/perangkat lunak dalam melanjutkan fungsinya bila terdapat masukan yang tidak sah.

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

2. Karakteristik Dalam

   Mencakup karakteristik-karakteristik yang dipikirkan oleh pengembang baik itu analis, perancang atau pemrogram yang meliputi:

   - Daya Tahan

     Kemudahan mengubah sistem/perangkat lunak untuk mengubah atau menambah kemampuan, memperbaiki unjuk kerja, atau memperbaiki kerusakan. Ini dapat dicapai dengan dokumentasi yang baik saat perencanaan, analisis, perancangan serta implementasi.

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

- Keluwesan

  Seberapa jauh kita dapat menempatkan sistem/ perangkat lunak dalam lingkungan yang berlainan dengan lingkungan saat sistem/perangkat lunak diciptakan.

- Kemudahan di Install

  Sejauh mana sistem/perangkat lunak yang dikembangkan mudah diterapkan dilingkungan pengguna.

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

- Daya Guna Ulang

  Kemudahan dalam menggunakan bagian (komponen) sebuah sistem dalam sistem-sistem yang lain. Modularisasi serta karakteristik-karakteristik yang dijanjikan bahasa-bahasa pemrograman berorientasi objek memungkinkan penggunaan ulang bagian dari sistem/perangkat lunak yang diciptakan pada suatu sistem dalam sistem yang lain

- Kemudahan Dipahami

  Kemudahan kita sebagai pengembang untuk memahami keseluruhan sistem

COPYRIGHT (C) Sept 2012

Kualitas (lanjutan)

- Kemudahan Dibaca

  Jika kemudahan dipahami membahas yang bersifat makro, maka kemudahan dibaca lebih bersifat mikro, yaitu kemudahan membaca serta memahami kode sumber sebuah sistem/perangkat lunak, khususnya pada tingkat pernyataan yang dirinci

- Daya Uji

  Tingkat dimana kita sebagai pengembang dapat memeriksa setiap unit dan sistem secara keseluruhan atau sesuai dengan spesifikasi yang ditentukan dalam *Software Requirement Specification*

COPYRIGHT (C) Sept 2012