Program Studi Teknik Informatika

FAKULTAS ILMU KOMPUTER
UNIVERSITAS DIAN NUSWANTORO

Prinsip-prinsip Software Testing

Latar belakang

- Software tester harus mencapai hasill pengujian yang optimal Ketika melakukan pengujian perangkat lunak, tanpa menyimpang dari tujuan.
- Tetapi bagaimana caranya bahwa yang kita lakukan sudah mengikuti strategi yang tepat untuk pengujian?

- Pikirkan dan bayangkan !

Contoh pengujian :
Misalnya kita akan memindahkan file yang ada di Folder A ke Folder B

Selain skenario biasa, bisa juga menguji kondisi berikut :

- Mencoba memindahkan file saat Terbuka
- Kita tidak memiliki hak keamanan untuk menempelkan file di Folder B
- Folder B ada di drive bersama dan kapasitas penyimpanan penuh.
- Folder B sudah memiliki file dengan nama yang sama, sebenarnya list tidak ada habisnya.

Atau misalkan kita memiliki 15 variabel masukan untuk diuji, masing-masing memiliki 5 kemungkinan nilai, jumlah kombinasi yang akan diuji adalah 5 ^ 15.
Jika kita menguji seluruh kemungkinan kombinasi, proyek WAKTU & BIAYA PELAKSANAAN akan meningkat secara eksponensial.
Untuk hal tersebut perlu strategi pengujian.

7 Prinsip Pengujian Perangkat Lunak

- Pengujian menunjukkan adanya cacat
- Pengujian yang lengkap adalah tidak mungkin
- Pengujian sejak awal
- Pengelompokan yang cacat
- Paradox Pestisida
- Pengujian tergantung pad konteks
- Tidak ada Error adalah keliru

Exhaustive Testing is not possible (1)

- **Pengujian menyeluruh tidak dimungkinkan**.
- Ingat batasan pengujian :
  - Tidak mungkin menguji semua program dengan lengkap
  - Kita hanya menguji berdasarkan kebutuhan system.
    - Bisa jadi tidak ada kesalahan pada requirementnya.
    - Requirement yang tidak lengkap atau ambigu dapat menyebabkan pengujian yang tidak memadai atau salah.
  - Pengujian lengkap (total) tidak mungkin dalam skenario saat ini.
  - Batasan waktu dan anggaran biasanya memerlukan perencanaan yang sangat cermat dari upaya pengujian.
  - Kompromi antara ketelitian dan anggaran.
  - Hasil tes digunakan untuk membuat keputusan bisnis untuk tanggal rilis.
  - Meskipun menemukan bug terakhir, Anda tidak akan pernah mengetahuinya
  - kehabisan waktu sebelum selesai menguji kasus uji
  - Tidak dapat menguji setiap jalur
  - Tidak dapat menguji setiap masukan yang valid
  - Tidak dapat menguji setiap masukan yang tidak valid

FAKULTAS ILMU KOMPUTER

Exhaustive Testing is not possible (1)

- Lalu bagaimana agar terhindar dari risiko yang tidak diinginkan? Lets to try !
- Pikirkan, operasi mana yang paling mungkin menyebabkan sistem Operasi kita gagal?
  - Saya yakin sebagian besar akan menebak, ***"Membuka 10 aplikasi berbeda sekaligus"***

Jadi jika akan menguji sistem Operasi, kita akan menyadari bahwa cacat/ kesalahan mungkin ditemukan dalam aktivitas multi-tasking dan perlu diuji secara menyeluruh

Defect Clustering (Pengelompokan Cacat) (2)

- Defect Clustering yang menyatakan bahwa sejumlah kecil modul berisi banyaknya cacat yang terdeteksi. Ini adalah penerapan Prinsip Pareto untuk pengujian perangkat lunak: sekitar 80% masalah ditemukan di 20% modul.
- Berdasarkan pengalaman, kita dapat mengidentifikasi modul berisiko tersebut. Namun pendekatan ini memiliki masalah tersendiri
- Jika pengujian yang sama diulang terus menerus, pada akhirnya kasus pengujian yang sama tidak akan menemukan bug baru lagi.

FAKULTAS ILMU KOMPUTER

Paradox Pestisida (3)

- Penggunaan campuran pestisida berulang-ulang yang sama untuk membasmi serangga selama bertani lama-kelamaan akan menyebabkan serangga mengembangkan resistensi terhadap pestisida. Sehingga pestisida tidak efektif pada serangga.
- Hal yang sama berlaku untuk pengujian perangkat lunak. Jika serangkaian pengujian berulang yang sama dilakukan, metode ini tidak akan berguna untuk menemukan cacat baru.
- Bagaiman mengatasinya ?
  - Tinjau ulang kasus uji
  - Revisi secara berkala
  - Tambah kasus uji baru & berbeda untuk membantu menemukan lebih banyak cacat.
- Penguji tidak bisa hanya bergantung pada teknik pengujian yang ada, harus terus menerus memperbaiki metode yang ada untuk membuat pengujian lebih efektif.

Pengujian menunjukkan adanya Cacat (4)

- Pada prinsip pengujian menyatakan bahwa - ***Pengujian berbicara tentang adanya cacat dan tidak berbicara tentang tidak adanya cacat.***
- Yaitu Pengujian Perangkat Lunak **mengurangi** kemungkinan cacat yang belum ditemukan yang tersisa di perangkat lunak tetapi bahkan jika tidak ada cacat yang ditemukan, bukan berarti perangkat lunak sudah 100 % bebas dari kesalahan.
- Tetapi bagaimana jika, kita bekerja ekstra keras untuk mengambil semua tindakan pencegahan & membuat produk perangkat lunak kita 99% bebas bug. Dan….. perangkat lunak masih tidak memenuhi kebutuhan & persyaratan klien.
- Ini membawa kita ke prinsip berikutnya, yang menyatakan bahwa- Tidak Ada Kesalahan

FAKULTAS ILMU KOMPUTER

Tidak ada Error ? Keliru (5)

- Ada kemungkinan perangkat lunak yang 99% bebas dari bug masih tidak dapat digunakan. Hal ini bisa terjadi bila system yang diuji secara menyeluruh ternyata untuk requirement yang salah.
- Pengujian perangkat lunak tidak hanya untuk menemukan cacat, tetapi juga untuk memeriksa bahwa perangkat lunak memenuhi kebutuhan bisnis.
- Tidak adanya kesalahan adalah suatu kekeliruan.
- Menemukan dan memperbaiki cacat tidak akan membantu jika system yang dibangun tidak dapat digunakan dan tidak memenuhi kebutuhan dan persyaratan pengguna.
- Untuk mengatasi masalah ini, maka perlu dilakukannya Pengujian Awal

Early Testing – Pengujian sejak dini (6)

- Pengujian harus dimulai sedini mungkin dalam Siklus Hidup Pengembangan Perangkat Lunak. Sehingga setiap cacat dalam persyaratan atau fase desain ditangkap pada tahap awal.
- Jauh lebih murah untuk memperbaiki Cacat pada tahap awal pengujian.
- Tetapi seberapa awal seseorang harus memulai pengujian?
- Direkomendasikan agar kita mulai menemukan bug saat requirement ditentukan.

FAKULTAS ILMU KOMPUTER

Pengujian bergantung pada Konteks (7)

- Pada dasarnya cara menguji yang dilakukan berbeda untuk setiap konteksnya.
- Misalnya ketika menguji situs e-niaga akan berbeda caranya Ketika menguji aplikasi komersial.
- Semua perangkat lunak yang dikembangkan tidak identik, yaitu pendekatan, metodologi, Teknik, dan jenis pengujian yang berbeda tergantung pada jenis aplikasinya.
- Misalnya pengujian POS (Point of Sale) apapun di toko ritel akan berbeda dari pengujian mesin ATM.

Mitos “Prinsip-prinsip pengujian hanya untuk referensi “

- “ Bahwa prinsip-prinsip pengujian hanya ada di referensi, dan saya tidak akan menggunakan pada prakteknya.”
- Ini sangat menyesatkan, mengapa ?
- Prinsip Pengujian akan membantu Anda membuat Strategi Pengujian yang efektif dan membuat draf kasus pengujian yang menangkap kesalahan.

- Tetapi mempelajari prinsip-prinsip pengujian sama seperti belajar mengemudi untuk pertama kalinya.

- Awalnya, saat Anda belajar mengemudi, Anda memperhatikan setiap hal seperti perpindahan gigi, kecepatan, penanganan kopling, dll. Tetapi dengan pengalaman, Anda hanya fokus pada mengemudi, sisanya akan datang secara alami. Sehingga Anda bahkan melakukan percakapan dengan penumpang lain di dalam mobil.

- Hal yang sama berlaku untuk prinsip pengujian. Penguji berpengalaman telah menginternalisasi prinsip-prinsip ini ke tingkat yang mereka terapkan bahkan tanpa berpikir. Oleh karena itu, mitos bahwa prinsip-prinsip tersebut tidak digunakan dalam praktik tidaklah benar.

FAKULTAS ILMU KOMPUTER

Bahan Diskusi (Buat kelompok 3-4 orang)

- Link video : https://www.guru99.com/software-testing-seven-principles.html
- Setelah melihat link tersebut, ceritakan pengalamanmu terkait dengan software testing