INFORMATIKA
UPN "VETERAN" JAWA TIMUR

Pengujian dan Implementasi Perangkat Lunak

Radical Rakhman Wahid

07 Mei 2019
02 Ramadhan 1440

Yang ada pada presentasi ini :

1. Dasar-dasar, prinsip, dan objektifitas pengujian *software*

2. Strategi pengujian *software*(*unit*, *integration*, *validation*, & *system testing*)

1. Dasar-dasar, Prinsip, dan Objektifitas

Pengujian *software*

Dasar-dasar pengujian perangkat lunak setidaknya mencakup dua hal, yakni :

1. Konfigurasi *software*, termasuk *software requirement specification*, *design specification* dan *source code*.
2. Konfigurasi uji, termasuk *test plan* & *procedure*, perangkat *testing* yang akan digunakan, *test case* dan hasil yang diharapkan.

Beberapa prinsip pengujian yang harus diperhatikan :

1. Semua pengujian harus dapat ditelusuri sampai ke persyaratan pelanggan.
2. Pengujian harus direncanakan lama sebelum pengujian itu dimulai.
3. Prinsip Pareto berlaku untuk pengujian perangkat lunak. Prinsip Pareto mengimplikasikan 80% dari semua kesalahan yang ditemukan selama pengujian sepertinya akan dapat ditelusuri sampai 20% dari semua modul program.
4. Pengujian harus mulai "dari yang kecil" dan berkembang ke pengujian "yang besar".
5. Pengujian yang mendalam tidak mungkin.
6. Paling efektif, pengujian dilakukan oleh pihak ketiga yang independen.

Objektifitas/tujuan dari pengujian perangkat lunak adalah sebagai berikut :

1. Agar dapat bertahan hidup di dunia bisnis perangkat lunak
2. Dapat bersaing dengan perangkat lunak yang lain
3. Penting untuk pemasaran global
4. Mengefektifkan biaya agar tidak banyak membuang perangkat lunak karena kegagalan pemasaran/produksi
5. Mempertahankan pelanggan dan meningkatkan keuntungan

2. Strategi

Pengujian *software*

Dengan mempertimbangkan proses dari titik pandang prosedural, pengujian di dalam konteks rekayasa perangkat lunak secara aktual merupakan empat langkah yang diimplementasi secara berurutan, berikut keempat langkah tersebut :

1. Pada awalnya, pengujian berfokus pada setiap modul secara individual, dengan memastikan bahwa modul berfungsi secara tepat sebagai suatu unit, karena itu dinamakan *unit testing*.
2. *Integration testing* menekankan pada masalah-masalah yang berhubungan dengan masalah-masalah verifikasi dan konstruksi program.
3. *Validation testing* memberikan jaminan akhir di mana perangkat lunak harus memenuhi semua persyaratan fungsional, tingkah laku dan kinerja.
4. Pengujian sistem membuktikan bahwa semua elemen sistem saling bertautan dengan tepat dan keseluruhan fungsi/kinerja sistem dapat dicapai.

2.1 *Unit testing*

Dalam pemrograman komputer, *unit testing* adalah pengujian bagian terkecil dari sebuah kode. Dalam pemrograman prosedural unit bisa menjadi modul seluruh tetapi lebih umum fungsi individu atau prosedur. Dalam pemrograman berorientasi objek unit sering merupakan seluruh antarmuka, seperti kelas, tetapi bisa menjadi metode individu.

Unit testing dilakukan untuk membuktikan apakah kode yang telah ditulis berfungsi sesuai dengan kehendak pemrogram. *Unit testing* dilakukan setelah pemrogram selesai menuliskan suatu kode/fungsi/method yang ada dalam suatu *class*. Dapat juga dilakukan setelah menambahkan sebuah fungsionalitas baru atau setelah melakukan *refactoring*.

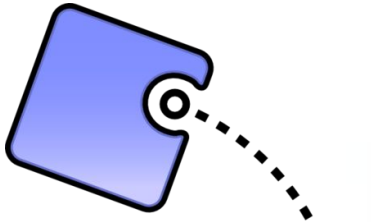
Tujuan *unit testing*

1. Mengisolasi setiap bagian dari program dan menunjukkan bahwa setiap bagian sudah benar
2. Mencari *error* di awal siklus pengembangan. Hal ini termasuk mengecek kebenaran *programmer* implementasi program dan mengatur alur atau ada bagian yang terlewat dari spesifikasi yang diperlukan
3. Mengurangi tingkat keraguan dalam unit tersebut dan dapat digunakan dalam pendekatan *testing buttom-up*

Framework/library unit testing

RSpec

pytest



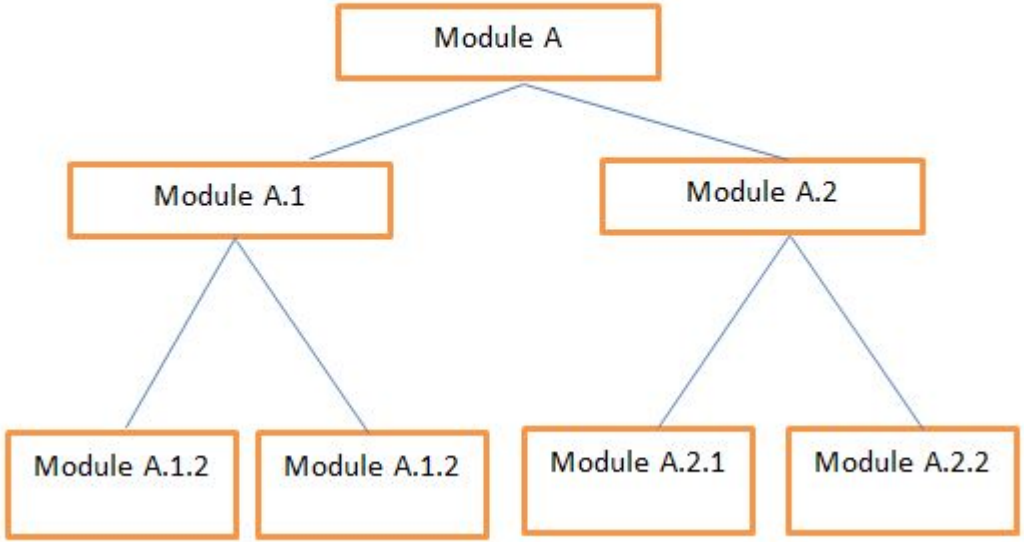
PHPUnit

2.2 *Integration testing*

Pengujian integrasi adalah sebuah tahapan pengujian yang sistematik untuk mengonstruksi struktur program seiring dengan menggabungkan fungsi program dengan antarmukanya. Pengujian terintegrasi bertujuan untuk mempergunakan komponen unit program yang sudah diuji dan membangun struktur seperti yang telah didesain sebelumnya.

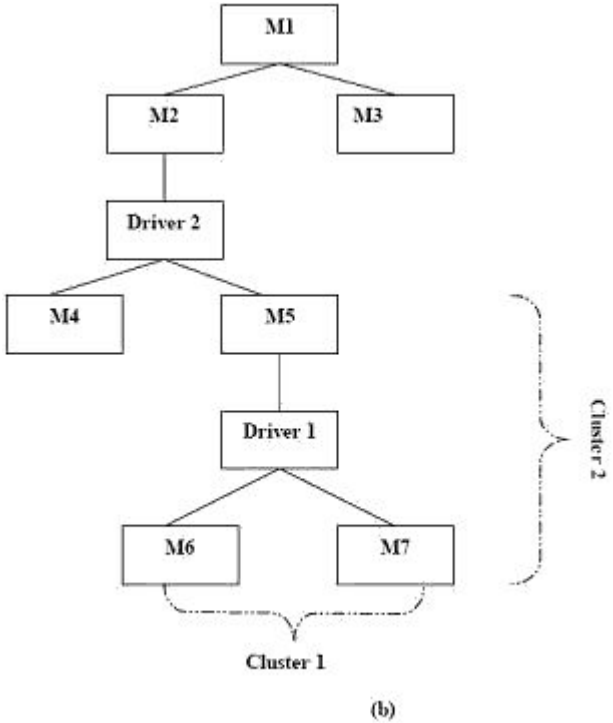
Tipe pengujian integrasi ini dibagi menjadi empat, yakni *Top-Down Integration, Bottom-Up Integration, Regression Integration,* dan *Smoke Integration*.

Top-Down Integration



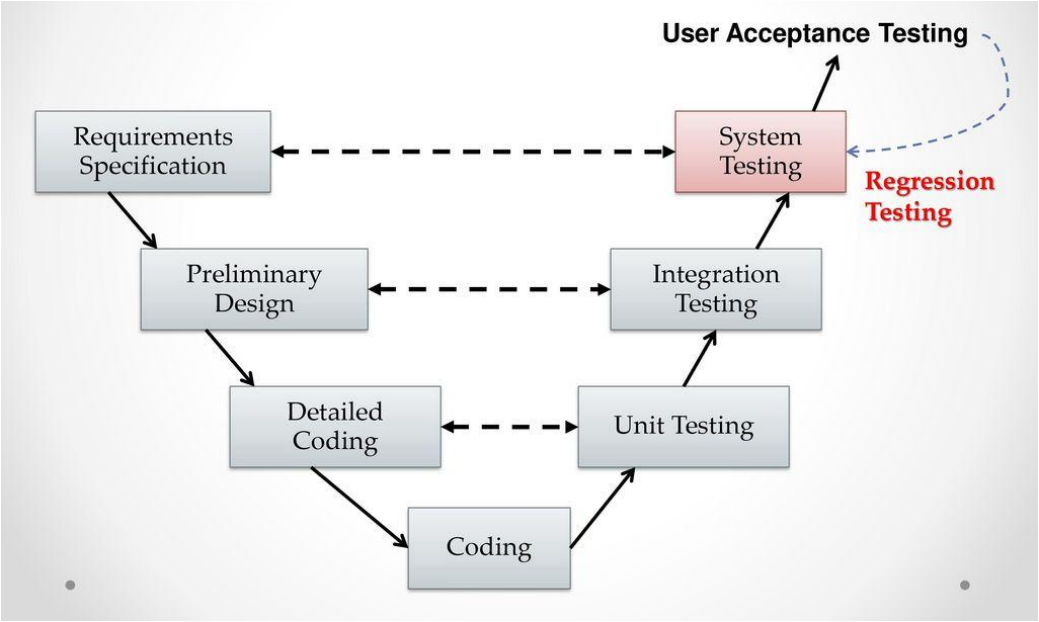
Pengujian ini dimulai dari modul yang lebih besar lalu didekomposisi ke modul yang lebih kecil

Bottom-Up Integration



Beberapa modul yang memiliki hirarki paling rendah dibuat menjadi beberapa cluster(kelompok) berdasarkan keterkaitan fungsinya. Dari setiap cluster dibuat program untuk menguji setiap cluster di mana setiap program yang digunakan untuk menguji ini harus mampu menguji fungsionalitas cluster disebut driver. Setelah setiap cluster selesai diuji dengan menggunakan program driver maka dilanjutkan pengujian dengan modul yang lebih tinggi.

Regression Integration



Pengujian ini merupakan eksekusi dari beberapa subset pengujian yang sudah terhubung atau saling terkait untuk menjamin bahwa modul yang baru masuk pengujian tidak mengubah fungsionalitas yang sudah diuji sebelumnya.

Smoke Integration

Pengujian ini merupakan pengujian terintegrasi yang biasa digunakan ketika waktu pengerjaan perangkat lunak cukup singkat dan biasanya untuk komponen atau modul yang ditambahkan pada perangkat lunak.

Kelebihan pengujian ini jika dilakukan pada perangkat lunak yang dikembangkan dengan waktu pendek adalah, sebagai berikut :

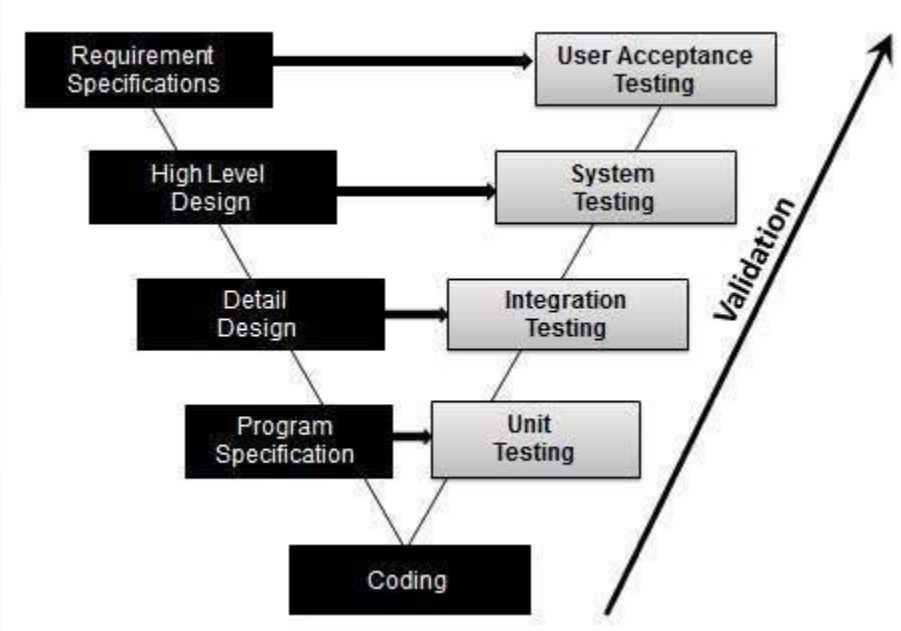
1. Risiko integrasi dapat diminimalisir karena pengujian dilakukan per hari sehingga ketidaksesuaian antar komponen cepat terdeteksi
2. Kualitas produk dapat diperbaiki karena fokus pada pengujian fungsionalitas produk
3. Diagnosis kesalahan dan perbaikan menjadi lebih mudah karena lebih jelas bahwa kesalahan biasanya muncul pada komponen yang baru saja diubah/diperbaiki
4. Proses mudah diperbaiki setiap harinya

2.3 *Validation testing*

Proses mengevaluasi perangkat lunak selama proses pengembangan atau pada akhir proses pengembangan dilakukan untuk menentukan apakah memenuhi persyaratan bisnis yang ditentukan.

Pengujian Validasi memastikan bahwa produk benar-benar memenuhi kebutuhan klien. Pengujian validasi juga dibutuhkan untuk menjawab pertanyaan, “Apakah kita membangun produk yang tepat?”

Ilustrasi :


Requirement Specifications
User Acceptance Testing
High Level Design
System Testing
Detail Design
Integration Testing
Program Specification
Unit Testing
Coding
Validation

2.4 *System testing*

System testing dapat dikatakan sebagai uji dari keseluruhan sistem dari sebuah *software* yang ada. Pengujian dilakukan secara lengkap dan sistem yang telah terintegrasi dapat dievaluasi apakah sistem yang dibuat telah sesuai dengan kebutuhan pengguna atau tidak. Setelah pengujian sistem telah selesai, para pengembang pun melakukan *acceptance testing*.

Pengujian sistem adalah suatu proses yang dilakukan untuk menilai kesempurnaan rancangan, mengevaluasi keunggulan dan kelemahan terhadap kualitas produk, mengevaluasi terhadap urutan yang sistematis, dan mengevaluasi keseimbangan jumlah pelaksanaan sesuai prosedur kegiatan *software*.

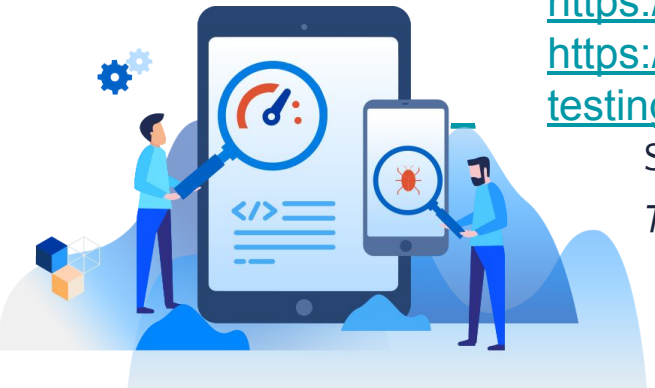
Tujuan *system testing*

1. Memastikan mutu dari suatu produk sehingga perlu diadakan pengujian apakah produk yang dihasilkan telah sesuai dengan harapan *developer*
2. Proses analisa dan entitas *software* yang mana bertugas untuk mendeteksi adanya perbedaan antar kondisi perangkat lunak yang ada dengan kondisi yang diinginkan oleh *user* dan *developer*
3. Melihat kerusakan atau kejanggalan suatu produk dengan melakukan evaluasi fitur-fitur *software* yang diluncurkan perusahaan

Referensi

https://docplayer.info/43429943-Dasar-dasar-pengujian-perangkat-lunak-fakultas-ilmu-komputer-dan-teknologi-informasi-jurusan-sistem-informasi-univesitas-gunadarma.html
https://docplayer.info/35606334-Teknik-pengujian-perangkat-lunak-pertemuan-14.html
http://www.kumpulancontohmakalah.com/2017/01/Strategi.Pengujian.Perangkat.Lunak.dan.Validasi.Pengujian.html
http://www.academia.edu/25239071/PENJELASAN_UNIT_TESTING
https://slideplayer.info/slide/12093991/
http://eprints.binadarma.ac.id/709/1/SOFTWARE%20QUALITY%20ASSURANCE%20MATERI%205.pptx
https://spaceku.com/apa-itu-system-testing/
https://www.tutorialspoint.com/software_testing_dictionary/validation testing.htm

S, Rosa A. & Shalahuddin, M. 2018. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek Edisi Revisi*. Bandung : Informatika



Terima kasih!