TESTING TAHAP PEMROGRAMAN

Rini Astuti

Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI
Jl. Ir. H. Juanda 96 Bandung 40132

E-mail: riniastuti@likmi.ac.id

Abstrak

Testing atau pengujian merupakan salah satu tahapan pengembangan perangkat lunak yang harus dilakukan untuk memastikan bahawa program yang dihasilkan sudah sesuai dengan kebutuhan. Kerumitan tahap pemrograman bergantung pada tahap sebelumnya yaitu tahap desain. Spesifikasi yang terukur dan terdefinisi dengan baik akan menyederhanakan tugas pemrograman. Bila terjadi kesalahan pada tahap desain, maka perbaikannya biasanya akan dilakukan dalam tahap pemrograman yang dalam hal ini dilakukan oleh programmer (dan ini bukan orang yang tepat).

Testing yang dilakukan melalui tahap pemrograman bisa dilakukan secara statis atau dinamis. Statis berarti pengujian dilakukan tanpa melakukan eksekusi kode sistem, sedangkan dinamis berarti pengujian dilakukan dengan cara mengeksekusi program. Hampir dalam sebagian besar tahap, spesifikasi, desain dan pengkodean program dilakukan. Testing yang dapat mendeteksi problem pada awal tahap akan mengurangi biaya perbaikan. Tulisan ini menjelaskan pilihan *tool* testing yang tersedia dalam pemrograman.

Kata-kata kunci : testing, pemrograman, tool testing

1. PENDAHULUAN

Umumnya testing program dilakukan setelah program perangkat lunak tersebut dibangun atau selama program tersebut dibuat.

Testing program adalah proses mengoperasikan program komputer pada saat program atau komponen program pada kondisi tertentu, mengamati atau mencatat hasilnya, dan membuat evaluasi terhadap beberapa aspek dari program dan komponen program tersebut.

Tujuan proses testing adalah

1. menilai apakah sistem yang dikembangkan sudah memenuhi kebutuhan yang sudah ditetapkan
2. menilai apakah sistem sudah beroperasi dengan benar
3. menemukan kesalahan pada sistem, terutama yang tidak terlihat sebelumnya

Prosedur yang dilakukan terhadap suatu kegiatan testing diantaranya adalah :

1. tentukan apa yang akan diuji
2. tentukan bagaimana cara pelaksanaan pengujian
3. buatkan kasus uji
4. tentukan hasil yang diharapkan atau hasil yang sebenarnya
5. laksanakan pengujian
6. bandingkan hasil pengujian dengan hasil yang diharapkan

2. TAHAP PEMROGRAMAN

Tahap pemrograman meliputi tiga segmen yaitu :

1. Penulisan spesifikasi program berdasarkan spesifikasi desain
2. Konversi spesifikasi program kedalam perintah–perintah mesin oleh programmer.
3. Programmer melakukan verifikasi perintah–perintah mesin dengan spesifikasi program.

Pemrograman dapat diibaratkan seperti pekerjaan kontruksi dalam pembuatan rumah yang dilakukan oleh tukang batu, tukang kayu, tukang listrik dan tukang ledeng. Orang-orang ini adalah orang yang melakukan spesifikasi desain dan material kemudian mengubahnya menjadi produk yang diinginkan.

Selama tahap pemrograman, banyak hal yang bisa disampaikan, tetapi umumnya adalah :

- Spesifikasi program
- Dokumentasi program
- Daftar program
- Program yang dapat dieksekusi
- Flowchart program
- Intruksi operator
- Hasil testing program

3. LINGKUP TESTING

Berkaitan dengan besarnya hal yang dapat disampaikan selama tahap pemrograman, maka cukup penting untuk berkonsentrasi terhadap bagian–bagian utama dalam tahap

testing program ini. Seorang tester seharusnya menyeleksi bagian–bagian yang dianggap penting tersebut dan memilih tool yang tepat yang berkaitan dengan bagian tersebut. Testing yang berkaitan dengan tahap pemrograman meliputi hal–hal berikut :

Kontrol integritas Data

Kontrol dengan spesifikasi khusus perlu diimplementasikan dengan cara tertentu untuk mencapai proses yang dinginkan secara presisi. Implementasi kontrol integritas data secara tidak benar mungkin membuat tingkat toleransi pengontrolan yang baku tidak tercapai dan karena ketidakpahaman tentang kegunaan kontrol, penyederhanaan persoalan mungkin dilakukan ketika suatu kontrol yang kompleks dibutuhkan untuk mencapai pengontrolan yang obyektif.

Testing yang dilakukan antara lain tentang *testing message–message error* data yang tidak berguna, testing kelengkapan pemeriksaan validasi data, testing kelengkapan prosedur penginputan data, testing logis tidaknya prosedur yang menjamin akurasi dan kelengkapan transaksi dan sebagainya.

Aturan Otorisasi

Aturan otorisasi perlu diimplementasikan dengan cara tertentu sehingga akan membuat orang sulit untuk menghindar dari aturan tersebut. Aturan otorisasi tidak hanya berkaitan dengan pelaksanaan suatu aturan tetapi juga berkaitan dengan catatan tentang metode–metode untuk menghindari aturan tersebut.

Testing yang dilakukan antara lain tentang testing logis tidaknya pemilihan metode otorisasi, testing spesifikasi program atau program untuk menentukan apakah aturan ini sudah diterapkan dengan baik, verifikasi bahwa sistem melakukan autentifikasi terhadap transaksi dimana transaksi tersebut melakukan otorisasi terhadap dirinya sendiri, verifikasi bahwa prosedur diimplementasikan untuk mengidentifikasi *authorizer* pada setiap transaksi dan sebagainya

Kontrol Integritas File

Kontrol terhadap integritas file seharusnya diimplementasikan dengan suatu cara yang dapat meminimalkan probabilitas dari *loss of file integrity* dan juga mencegah terjadinya *loss of integrity* serta mendeteksi terjadinya *loss* dalam waktu tertentu.

Testing yang dilakukan antara lain tentang verifikasi bahwa penempatan individu sudah sesuai dengan skill dan ketersediaan waktu, bandingkan implementasi kontrol terhadap

kebutuhan integritas yang ditetapkan selama tahap requarement, review logis tidaknya frekuensi verifikasi integrasi file, konfirmasi dengan user apakah semua subset file telah diamankan oleh kontrol yang yang terintegrasi dan sebagainya.

Audit Trail

Audit trail dapat diasumsikan sebagai proses penelusuran suatu program. Proses ini perlu diimplementasikan dengan suatu cara yang dapat memfasilitasi pencarian informasi tentang alur suatu pemeriksaan. Bila langkah–langkah pemeriksaan berisi informasi yang dibutuhkan tetapi biayanya terlalu mahal atau memakan waktu yang lama, maka nilainya berkurang secara signifikan. Implementasinya mencakup sejumlah informasi yang disimpan, penyederhanaan pencarian keterangan terhadap informasi itu, referensi informasi silang untuk keperluan pencarian informasi, seperti lamanya waktu yang digunakan untuk memeriksa pencarian informasi alur yang disimpan.

Testing yang dilakukan antara lain tentang testing kelengkapan audit trail dari dokumen ke control total atau sebaliknya, testing apakah audit trail record sudah memuat semua definisi bagian audit trail, verifikasi apakah perencanaan uji untuk audit trail telah dinyatakan, review kelengkapan proses rekontruksi transaksi dan sebagainya.

Contigency Plan Written

Contigency plan (rencana cadangan) adalah sekumpulan prosedur yang detail dalam bentuk langkah per langkah yang task–task nya dieksekusi dalam *event* persoalan. Dalam perencanaan, seharusnya langkah–langkah pencegahan disebutkan sehingga data dan resource lain yang dibutuhkan dapat tersedia ketika contigency plan diaktifkan. Pendekatan desain *Contigency plan* merupakan nilai kecil sampai ia didokumentasikan dan berada ditangan orang yang butuh menggunakannya.

Testing yang dilakukan antara lain tentang konfirmasi dengan manager operasi apakah semua orang telah diidentifikasi dalam *contigency pla* , konfirmasi apakah semua *resource* yang diperlukan telah diidentifikasi, konfirmasi bahwa rencana alternatif tersedia sebagai cadangan, testing tentang kelengkapan rencana testing dan sebagainya.

Perancangan tingkat layanan sistem

Tingkat layanan yang diinginkan hanya bisa menjadi kenyataan ketika metode–metode dan prosedur–prosedur dibangun. Salah satu prosedur yang seharusnya dibangun adalah pemantauan tingkat layanan untuk menjamin pemenuhan spesifikasinya. Pemantauan

secara rutin dalam waktu yang lebih dilakukan untuk menjamin tingkat layanan yang ingin dicapai atau bila tidak deteksi dini harus dilakukan sehingga aksi perbaikan bisa dilakukan.
Testing yang dilakukan antara lain tentang verifikasi tentang kriteria performansi program selama testing, verifikasi tentang kriteria performansi sistem selama testing, testing tentang kelengkapan *trainning program*, konfirmasi dengan personel operasi bahwa software pendukung tersedia dan sesuai dengan kriteria performansi, konfirmasi dengan personel operasi bahwa hardware pendukung tersedia dan sesuai dengan spesifikasi dan sebagainya.

Prosedur Keamanan

Keamanan merupakan gabungan antara pelatihan dan kesadaran pegawai, ditambah teknik dan tool–tool keamanan yang diperlukan. Prosedur yang menjamin bahwa dua hal tersebut bekerja sama harus dikembangkan selama tahap program.
Testing yang dilakukan antara lain tentang konfirmasi dengan bagian security apakah terdapat security hardware dan security software, testing tentang password dissemination dan rencana maintenance, testing tentang prosedur security trainning dan sebagainya.

Menyesuaikan Program dengan Metodologi

Prosedur seharusnya dilaksananakan untuk menjamin pemenuhan standar–standar pengembangan, kebijakan, prosedur–prosedur dan metode–metode. Bila pelanggaran pemenuhan terdeteksi, maka pengukur yang sesuai harus diambil untuk mendapatkan simpangan terhadap metodologi atau untuk memodifikasi sistem atau desain sehingga pemenuhan tercapai. Bila metodologi tidak diperlukan untuk memuaskan user, maka perlu untuk memuaskan desain layanan informasinya.
Testing yang dilakukan antara lain tentang testing program untuk menjamin bahwa ia sesuai dengan prosedur dan aturan organisasi, serta metode layanan informasi, aturan accounting, pemerintah, standar industri yang diperlukan dan sebagainya.

Mencocokkan Program dengan Kebenaran Rancangan

Pengubahan kondisi menyebabkan banyak personel layanan informasi proyek menghiraukan obyektif proyek selama tahap program. Alasannya adalah terdapat perubahan yang cukup besar sehingga pemantauan pemenuhan terhadap obyektif sistem menjadi tak berarti. Tim penguji seharusnya mengabaikan pemikiran ini dan secara kontinu memonitor implementasi terhadap obyektif–obyektif tersebut. Bila obyektif–obyektif

belum dijumpai, salah satunya seharusnya dirubah atau sistem diubah untuk dibawa kearah pemenuhan spesifikasi fungsional aplikasinya.

Testing yang dilakukan antara lain tentang konfirmasi ke user management bahwa obyektif software tidak berubah, Bandingkan hasil program dengan obyektif yang ada, verifikasi bahwa implementasi sistem mengeluarkan hasil yang benar, Konfirmasi dengan user bahwa input ke siatem sudah mencapai konsistensi dan reabilitas yang diinginkan dan sebagainya

Menyesuaikan Program dengan kemudahan pemakaian

Implementasi spesifikasi sistem boleh mengurangi beberapa aspek desain mudah untuk menggunakan **kecuali** untuk aspek–aspek yang secara khusus telah didefinisikan sebelumnya. Pemrograman adalah suatu proses penerjemahan spesifikasi desain dan mungkin gagal untuk mencapai maksud mudah untuk menggunakan ini. Pemrograman harus mencapai spesifikasi desain mudah untuk menggunakan ini seperti spesifikasi fungsional lainnya.

Testing yang dilakukan antara lain tentang testing kesesuaian dokumen aplikasi terhadap spesifikasi desain, testing tentang usability dari instruksi-instruksi interface manusia, verifikasi tentang kemudahan mencetak dokumen, verifikasi tentang kemudahan untuk menginput dokumen dan sebagainya.

Program harus dapat dirawat

Metode pada desain program dan pengkodean boleh memiliki signifikasi yang lebih besar dalam kemampuan pemeliharaannya daripada spesifikasi desainnya. Aturan untuk kode yang dapat di*maintain* seharusnya secara parsial didefinisikan oleh bagian yang berwenang dan oleh spesifikasi sistem. Programmer seharusnya menggunakan gagasan dan pengalamannya untuk mengembangkan kode- kode yang dapat dimodifikasi.

Testing yang dilakukan antara lain tentang verifikasi bahwa program sudah sesuai dengan spesifikasi maintenance, review dokumentasi tentang kelengkapan dan kegunaan program, review program untuk logika program yang rumit dan sebagainya

Program harus dapat dimodofikasi (Portability)

Portabilitas program tergantung dari bahasa yang dipilih dan bagaimana bahasa digunakan. Spesifikasi seharusnya memberikan indikasi bisa tidaknya pemrograman untuk protability, dan *coding* seharusnya menyesuaikan dengann spesifikasi tersebut. Bila protability

merupakan lingkup yang utama dan spesifikasi program gagal untuk mendefinisikan pengkodean portabilitas secara cukup, programmer seharusnya membuat setiap penulisan metode yang mungkin secara jelas.

Testing yang dilakukan antara lain tentang review apakah sistem menghindari penggunaan hardware yang khusus, Review apakah sistem menghindari penggunaan software yang khusus, tentukan kelengkapan dokumentasi portabilitas, tentukan kelengkapan dokumentasi pengoperasian, review apakah nilai data digunakan pada mesin yang independen dan sebagainya

Penyesuaian Program dengan rancangan parameter

Spesifikasi desain seharusnya mengindikasikan pengiriman parameter ke dan dari sistem aplikasi lainnya. Biasanya pengalaman programmerlah untuk menentukan spesifikasi sistem yang sesuai. Jaminan ini tidak hanya berkaitan dengan program menyesuaikan dengan desain saja, tetapi juga jaminan tantang spesifikasi dari aplikasi interkoneksi belum berubah ketika desain didokumentasi.

Testing yang dilakukan antara lain tentang verifikasi apakah layout record, nilai data, struktur data yang umum digunakan oleh aplikasi interface, verifikasi kelengkapan dokumen interface dan sebagainya

Membuat Prosedur

Prosedur seharusnya dikembangkan selama pemrograman untuk mengoperasikan sistem aplikasi. Selama tahap berikutnya, program siap eksekusi akan dioperasikan dan intruksi–intruksi yang diperlukan seharusnya dikembangkan diawal tahap. Prosedur pengoperasian seharusnya konsisten dengan kebutuhan operasional sistem aplikasi.

Testing yang dilakukan antara lain tentang review program untuk menentukan ukuran maksimun program, review apakah perubahan yang dibuat dalam pemrograman berakibat ke operasi, review kelengkapan dokumen pengoperasian dan sebagainya

Program memiliki performansi sesuai kriteria

Pemberian kesempatan pengoperasian program yang pertama kali kepada user adalah untuk menjamin bahwa sistem dapat mencapai tingkat performansi yang diinginkan. Pada point ini, instruksi–instruksi untuk membangun kebutuhan-kebutuhan user telah didefinisikan dan dievaluasi. Sebuah jaminan performansi yang potensial diawal tersedia sebagai kesempatan untuk membuat penyesuaian performansi bila diperlukan.

Testing yang dilakukan antara lain tentang testing terhadap anggaran proyek untuk verifikasi bahwa biaya nyata mendekati biaya yang dianggarkan, konfirmasi dengan manager proyek bahwa anggaran proyek dimonitor, konfirmasi dengan manajemen user bahwa benefitnya cukup masuk akal, periksa apakah proyek sesuai jadwal yang direncanakan dan sebagainya.

4. TANGGUNG JAWAB TES

Tahapan programming merupakan tahap yang tidak banyak membutuhkan keterlibatan user kecuali ada pertanyaan muncul yang berhubungan dengan tahap spesifikasi atau desain.

Pada kenyataannya perubahan-perubahan yang diinginkan user banyak terjadi pada tahap programming ini tetapi hal ini dapat diminalkan dengan melakukan evaluasi pada tahap desain atau dengan menundanya sampai sistem dioperasikan, jika tidak dapat ditunda maka perubahan yang dilakukan hendaknya diuji terlebih dahulu sebelum mengubah spesifikasi yang telah diberikan.

Tujuan utama dari testing pada tahap ini adalah untuk menjamin bahwa spesifikasi desain telah diimplementasikan dengan benar. Testing program tidak berhubungan langsung dengan pemenuhan kebutuhan user tetapi lebih menitikberatkan pada kesesuaian antara program yang dikembangkan dengan spesifikasi desain dan memastikan program dapat berjalan dengan baik.

5. BEBERAPA TOOL

Terdapat dua macam tool (alat bantu) yang telah banyak digunakan pada test tahap programming ini, yang keduanya saling melengkapi yaitu :

a. Desk Debugging

Test yang dilakukan oleh programmer secara individu untuk mengecek kelengkapan dan kebenaran program agar tidak menimbulkan kesalahan yang terus merambat ke proses lebih lanjut. Desk debugging dapat dilakukan secara ekstensif atau secara minimal tergantung kepada :

i. Waktu tunggu sampai deliverable program selanjutnya diterima
ii. Jadwal Implementasi
iii. Sumber daya Testing

iv. Efisiensi Test tools
v. Kebijakan Departemen

Desk debugging dapat berupa pengecekan sintaktikal, struktural maupun fungsional, berikut ini adalah gambaran apa yang dilakukan pada masing-masing test tersebut.

- ***Syntatical Desk Debugging***

  Pada syntatical debugging ini programmer mengecek apakah spesifikasi program dan statement program telah dibangun berdasarkan metodologi yang dipakai dan telah memenuhi persyaratan yang ditetapkan compiler. Syntactical cheking menjawab pertanyaan-pertanyaan seperti:

  i. Apakah penulisan program telah sesuai dengan ketetentuan kompiler?
  ii. Apakah elemen data telah teridentifikasi dengan benar?
  iii. Apakah tipe data yang digunakan telah sesuai untuk menampung nilai yang akan digunakan?

- ***Structural Desk Checking***

  Kesalahan struktural program kadang terlihat seperti kesalahan fungsional program sehingga pendeteksiannya karena memiliki efek yang sama yaitu fungsi dalam program tidak berjalan sebagaimana mestinya. Pertanyaan-pertanyaan yang dikemukakan pada Structural Checking ini antara lain:

  i. Apakah seluruh elemen data yang didefinisikan telah digunakan dalam program?
  ii. Apakah seluruh percabangan dalam program telah menuju ke bagian yang benar?
  iii. Apakah batasan-batasan nilai elemen data telah ditetapkan?

- ***Functional Desk Debugging***

  Pada fungsional debugging ini programmer melakukan pengecekan fungsi-fungsi yang dimiliki oleh program apakah telah berjalan seperti yang didinginkan. Pertanyaan-pertanyaan yang dikemukakan pada Functional Debugging antara lain:

  i. Apakah program akan dapat menjalankan fungsinya sesuai dengan ketentuan?
  ii. Apakah system dapat mendeteksi data yang tidak akurat atau data yang tidak masuk akal?

b. Program Peer Review

Program Peer Review ini dilakukan oleh rekan sekerja programmer untuk melakukan review terhadap fungsionalitas, struktur maupun sintaks program. Review ini dapat berjalan efektif jika tidak dalam keadaan mendesak. Peer Review ini dapat dilakukan secara formal maupun informal. Jika dijalankan secara fromal maka peer review manjadi bagian dari proses programming sedangkan jika secara informal maka review tersebut menjadi kebijakan dari kepala programmer.

Tim peer review sebaiknya terdiri dari dari tiga sampai enam anggota. Tim terdiri minimal tiga anggota agar mendapat opini yang lebih bervariasi dan diskusi dapat berjalan. Anggota yang sebaiknya terlibat dalam tim peer review meliputi :

- Programmer Komputer (minimal 2 orang)
- Job control specialist
- Operator komputer
- Pegawai Control
- Programming supervisor

Program Peer Review dilakukan dengan menjalankan langkah-langkah berikut :

1. **Tentukan aturan dasar peer review**

   Sebaiknya ditentukan terlebih dahulu aturan dasar dari peer review agar dapat berjalan sesuai rencana, aturan tersebut antara lain :

   - Poin-poin yang akan di review
   - Cara penentuan peer review
   - Cara pemilihan ketua peer review

2. **Lakukan pemilihan tim peer review**

   Pemilihan tim perlu direncanakan terlebih agar masing-masing anggota dapat mengalokasikan waktunya.

3. **Lakukan pelatihan anggota tim**

   Pada awal terbentuknya tim atau jika ada anggota baru perlu dilakukan pelatihan anggota agar mengetahui aturan-aturan dasar dan cara-cara melakukan peer review.

4. **Pilih metode review**

   Ketua tim menentukan metode yang akan dipakai dalam peer review. Review terdiri dari dua bagian yaitu : (i) review global terhadap spesifikasi program dan (ii) review terhadap program. Ada empat macam meode yang dapat digunakan dalam peer review, yaitu :

   - Flowchart
   - Source Code
   - Contoh input dan output
   - Spesifikasi program

5. **Lakukan peer Review**

   Kegiatan peer review ini dikoordinasi oleh ketua tim programmer

6. **Buat kesimpulan**

   Setelah peer review selesai dijalankan ketua tim programmer membuat catatan-catatan, kesimpulan dan rekomendasi dari hasil peer review

7. **Persiapkan laporan**

   Dalam formal peer review dianjurkan untuk dibuat dokumentasi dari proses review tersebut.

6. PROSES TES

Banyaknya acuan atau perhatian test yang menjadi acuan dalam testing tahap programming ini tergantung pada tingkat kepercayaan tim testing atas hasil testing tahap sebelumnya yaitu tahap desain. Semakin tinggi tingkat kepercayaan tim atas hasil testing pada tahap desain, semakin sedikit acuan tes yang dimiliki. Pada testing tahap programming ini tim tes harus mengidentifikasi test concern dan membangun proses tes untuk tiap acuan tersebut. Tujuan dari tes pada tahap ini yang perlu terus diperhatikan antara lain :

- System Maintainability
- Kesesuaian spesifikasi system
- Perancangan test yang memadahi untuk evaluasi program
- Kelengkapan dokumentasi program

Proses testing meliputi penentuan kriteria tes, proses tes yang direkomendasikan, teknik dan tool .

7. KESIMPULAN

- Kompleksitas testing tahap pemrograman bergantung pada hasil testing tahap desain.
- Testing pada tahap pemrograman dilakukan secara statik (tanpa mengeksekusi program) atau dinamik (dg mengeksekusi program).
- Pendeteksian kesalahan di awal tahap dapat memperkecil biaya untuk perbaikan program.
- Tujuan utama testing tahap pemrograman adalah kesesuaian antara program yang dikembangkan dengan spesifikasi desain.
- Testing program dapat dilakukan programmer secara perorangan (Desk debugging) atau dalam suatu tim (Program Peer Review).

8. DAFTAR PUSTAKA

1. Perry William, "*Effective Method for Software testing*", John Willey Sons, 1995
2. Roger S. Pressman, "*Software Engineering*", McGraw-Hill International, 2002
3. Ian Sommerville, "*Software Engineering*", Addison Willey, 1996
4. Ron Patton, "*Software Testing*", **Sams Publishing, 2005**