PENGGUNAAN METODE KOTAK HITAM DAN KOTAK PUTIH DALAM MENGUJI SEBUAH PRODUK SISTEM INORMASI

Kristina[1], Genrawan Hoendarto[2], Sandi Tendean[3]
[1] Sistem Informasi, STMIK Widya Dharma Pontianak
[2,3] Teknik Informatika, STMIK Widya Dharma Pontianak
e-mail: [1]vinalim@yahoo.com, [2]genrawan@yahoo.com [3]sanditendean@gmail.com

Abstract

Black box testing method is a method that allows random, unplanned and done testing by people who do not understand in detail about product specifications. This method takes a long time but is usually able to reveal unexpected product weaknesses. If using the method of blackbox testing in testing an information system product then the most appropriate to conduct testing is the consumer. While white box testing method is a testing method that prioritizes the knowledge of clear system structure and input data that has been prepared in accordance with the system structure. White box testing method in testing an information system product is very suitable implemented by the system development team. By using the method of black box testing and white box testing in testing an information system product, it is very likely a product information system to meet the needs of users because it involves the parties involved in the development and those who use the information system.

Keywords : black box testing, white box testing, information system testing, customer, system developer.

Abstraksi

Metode Black box testing adalah metode yang yang mengijinkan pengujian acak, tidak terencana dan dilakukan oleh orang-orang yang tidak mengerti secara detil mengenai spesifikasi produk. Metode ini memerlukan waktu yang cukup lama namun biasanya mampu mengungkapkan kelemahan produk yang tidak diduga. Jika menggunakan metode blackbox testing dalam menguji sebuah produk sistem informasi maka yang paling sesuai untuk melakuan pengujian adalah konsumen. Sedangkan metode white box testing adalah metode pengujian yang mengutamakan pengetahuan akan struktur sistem yang jelas dan data input yang sudah disiapkan sesuai dengan struktur sistem. Metode white box testing dalam menguji sebuah produk sistem informasi sangat sesuai dilaksanakan oleh tim pengembang sistem. Dengan menggunakan metode black box testing dan white box testing dalam menguji sebuah produk sistem informasi, maka sangat besar kemungkinan sebuah produk sistem informasi memenuhi kebutuhan user karena melibatkan pihak yang terlibat dalam pengembangan dan pihak yang menggunakan system informasi tersebut.

Kata kunci : pengujian kotak hitam, pengujian kotak putih, sistem informasi, konsumen , pengembang sistem.

1. PENDAHULUAN

Dalam masyarakat digital baik disadari maupun tidak disadari bahwa sistem informasi dan sudah merupakan komponen penting dalam kehidupan masyarakat digital. Hal ini menuntut perubahan dari cara bagaimana sebuah sistem itu dikembangkan. Sistem informasi dan system perangkat lunak juga merupakan satu kesatuan yang saling mendukung agar bidang kehidupan sosial yang ditangani dapat berjalan sukses. Selain itu dalam masyarakat digital sangat diperlukan para ahli pengembangan sistem. Berbagai metode, teknik dan perangkat baru dikembangkan untuk mendukung pembuatan, pembangunan dan perawatan sistem. Menurut Al Fatta (2007:14): "Sistem informasi merupakan sistem dengan komponen-komponen yang bekerja untuk mengolah data menjadi informasi".

Sistem informasi memegang peranan yang penting dalam kehidupan masyarakat baik secara ekonomi maupun sosial, maka dengan alasan ini memberikan tekanan yang kuat bagi para profesional pengembang sistem informasi memfokuskan perhatian mereka pada isu kualitas. Sistem informasi yang berkualitas rendah akan menghilang dan tidak mampu bertahan dan tidak akan diterima dalam masyarakat. Kondisi ini membutuhkan staff pengembang sistem informasi yang punya minat dan terlatih dalam menekuni masalah kualitas produk. Staff yang berkualitas tinggi memastikan bahwa produk sistem informasi dibangun tepat waktu, dengan biaya yang terencana, dan memenuhi realibility, correctness, usability, dan ability dalam memenuhi semua kebutuhan user.

Sebagai respon bagi permintaan sistem informasi yang berkualitas tinggi dan kebutuhan akan tenaga profesional yang terdidik baik, maka ada langkah-langkah yang dilakukan untuk mengubah cara suatu sistem informasi dibangun dan dirawat dan pendidikan bagi para pengembang sistem informasi. Pada kenyataannya, profesi pengembang sistem informasi secara perlahan berubah menjadi suatu bidang ilmu yang formal. Sebagai bidang ilmu yang baru, maka bidang ilmu sistem informasi berelasi dengan bidang ilmu teknik lainnya dan berasosiasi dengan inti dari pengetahuan, kode etika dan proses sertifikasi. Tujuan dari tim perumus ini adalah mendefinisikan inti dari pengetahuan yang dapat mewakili disiplin ilmu eksak, dengan mendiskusikan perubahan secara alami dari profesi pengembang software, dan juga mendefinisikan kode etika dari pengembang software.

Selain itu ada kelompok orang yang sangat menentukan kualitas dari produk sistem informasi yang dihasilkan oleh tim pengembang.Kelompok orang tersebut sering disebut dengan jaminan kualitas sistem. Software quality assurance (SQA) merupakan sebuah tim kelompok orang dengan pelatihan dan keterampilan yang diperlukan untuk memastikan bahwa semua tindakan yang perlu diambil selama proses de-velopment sehingga perangkat lunak yang dihasilkan sesuai dengan persyaratan teknis yang telah ditetapkan. Fungsi utama dari tim SQA adalah untuk memberikan informasi yang relevan dengan kualitas produk kepada pihak manajemen.

Untuk mendukung peran ini, jurnalis dan lain-lain menyarankan bahwa sebuah tim SQA menyiapkan rencana SQA untuk setiap proyek yang menggambarkan audit dan tinjauan yang harus dilakukan, standar-standar yang relevan dengan proyek, tata cara pelaporan dan pelacakan kesalahan, dan dokumen-dokumen mereka akan menghasilkan. Tim juga harus menjelaskan sifat dari hubungan komunikasi antara mereka sendiri, kelompok pengembangan, dan kelompok pengujian. Jika tim SQA mengidentifikasi masalah-masalah yang berdampak pada kualitas produk dan kemudian laporan ini kepada manajemen, maka manajemen harus menangani masalah-masalah dan menyediakan sumber daya untuk mengatasinya sehingga kualitas ditingkatkan.

Selama ini orang-orang hanya menfokuskan diri pada kualitas sebuah perangkat lunak sehingga metode pengujian yang dikembangkan diterapkan untuk sistem perangkat lunak. Sistem perangkat lunak merupakan sebuah sistem informasi yang bekerja dalam lingkungan sistem perangkat lunak. Atas dasar bahwa sistem perangkat lunak merupakan sistem informasi maka diharapkan bahwa metode pengujian kotak putih dan kotak hitam bisa diterapkan untuk produk sistem informasi.

2. METODE PENELITIAN

2,1. Metode dan bentuk penelitian

Dalam penelitian ini menggunakan metode penelitian deskriptif. Penelitian deskriptif juga merupakan penelitian yang berusaha mendeskripsikan dan menginterprestasikan sesuatu, misalnya kondisi atau hubungan yang ada, pendapat yang berkembang, proses yang sedang berlangsung dan berdasarkan fakta yang sebenarnya.

2.2 Metode Pengumpulan, Pengolahan dan Penyajian Data.

Teknik pengumpulan data yang digunakan antara lain :

a. Observasi

   Metode ini dilaksanakan dengan terjun langsung ke lapangan dan melihat secara dekat proses-proses kegiatan kerja yang berlangsung pada sebuah organisasi sehingga penulis dapat mengumpulkan data dan informasi secara lengkap.

b. Wawancara

   Metode ini adalah dimana penulis melakukan tanya jawab kepada pihak-pihak yang terlibat baik dalam proses pengembangan maupun proses penggunaan produk sistem informasi.

c. Studi Kepustakaan

   Metode pengumpulan data terakhir yang digunakan oleh penulis adalah dengan cara mengumpulkan data berupa bahan-bahan pendukung seperti teori-teori, konsep-konsep yang berasal dari literatur-literatur yang ada dan karya ilmiah yang dapat dipertanggung jawabkan secara ilmiah.

3. HASIL DAN PEMBAHASAN

3.1 Permasalahan

Pengujian produk sistem informasi merupakan isu penting yang harus diperhatikan oleh para pengembang sistem informasi. Pengujian merupakan proses yang harus dimulai sejak memulai rencana pengembangan sistem informasi. Selain itu anggaran yang disiapkan untuk menguji sistem seharusnya lebih besar dibandingkan dengan anggaran yang disiapkan untuk mengembangkan sistem informasi. Hal ini bertujuan untuk meminimalisir tingkat kerugian yang akan dihadapi ketika produk sistem informasi sudah digunakan secara aktif. Dalam menguji sebuah sistem informasi maka banyak teknik pengujian yang bisa diterapkan seperti pengujian kotak putih dan kotak hitam. Dua pengujian ini akan dianalisis dengan tujuan mencari

kelebihan dan kekurangan dari setiap teknik ini dan diharapkan kelebihan dan kekurangan tersebut akan dapat saling melengkapi.

3.2 Hasil dan Pembahasan

Proses pengembangan software telah digambarkan sebagai serangkaian fase, prosedur, dan langkah-langkah yang akhirnya menghasilkan pembuatan sebuah produk software. Pengujian sendiri mempunyai hubungan yang erat dengan dua proses lain yang dikenal proses validasi dan verfikasi. Validation (mengesahkan) adalah sebuah proses mengevaluasi sistem software atau komponen- komponennya atau pada akhirnya siklus pengembangan dalam menentukan apakah software memuaskan atau memenuhi kebutuhan yang dispesifikasi. Validasi biasanya berasosiasi dengan eksekusi traditional yang berbasis pengujian, yang mana melatih atau menguji koding dengan kasus ujicoba.

Jika menggunakan pendekatan teknik bagi pengembangan sistem informasi berarti menerapkan pemahaman pengembangan proses yang jelas dan baik, terencana baik, model siklus pengembangan terdefinisikan dan sudah menjadi standar pengembangan, patokan yang tetap bagi produk-produk software yang ada, ukuran yang harus dikerjakan untuk mengevaluasi kualitas proses dan produk, komponen-komponen yang bisa digunakan ulang, proses validasi dan verifikasi memegang kunci penting dalam penentuan kualitas software serta para pengembang dan penguji mempunyai pendidikan, latihan dan sertfikasi yang tepat.

Verifikasi adalah suatu proses mengevaluasi system software atau komponen terkait apakah produk yang dikembangkan berdasarkan fase yang telah disepakati memuaskan atau memenuhi kondisi yang telah diinginkan pada permulaan. Verifikasi biasanya berasosiasi dengan aktivitas seperti inspeksi dan review ulang dari software yang telah diterima. Pengujian secara umum dideskripsikan sebagai sekumpulan prosedur yang dibuat untuk mengevaluasi beberapa aspek dari perangkat lunak yang dibangun. Pengujian juga digambarkan sebagai sebuah proses untuk mengungkapkan kelemahan dari software, dan untuk membangun software yang mencapai faktor-faktor yang telah dispesikasikan dan dijadikan sebagai patokan penilaian kualitas sebuah software.

Definisi lain yang ditemukan dalam beberapa literatur, mendefinisikan *testing* sebagai pengukuran kualitas *software*. Sama halnya dengan *testing*, pengertian kualitas bagi tiap praktisi dapat berbeda-beda, karena kualitas memang merupakan suatu hal yang subjektif dan abstrak. Berikut ini beberapa definisi sederhana tentang kualitas. Menurut CROSBY: Kualitas adalah pemenuhan terhadap kebutuhan. Menurut ISO-8402: Kualitas adalah keseluruhan dari fitur yang menjadikan produk dapat memuaskan atau dipakai sesuai kebutuhan dengan harga yang terjangkau. Menurut W.E. Perry: Kualitas adalah pemenuhan terhadap standar.Menurut R. Glass: Kualitas adalah tingkat kesempurnaan. Menurut J. Juran: Kualitas adalah tepat guna. Berdasarkan pemaparan di atas maka dapat disimpulkan kualitas adalah pemenuhan terhadap suatu kebutuhan yang dapat memuaskan dan memenuhi standar dari kebutuhan tersebut.

Catatan bahwa definisi pengujian cukup umum dan bisa diterima secara ilmiah. Pengujian mencakup aktivitas validasi dan verifikasi, dan meliputi daerah pengujian seperti : pemeriksaan ulang teknis, rencana pengujian, pelacakan ujicoba, desain kasus ujicoba, pengujian unit, tes integrasi, sistem pengujian, tes yang diakui, dan kegunaan pengujian. Definisi tersebut juga menggambarkan pengujian sebagai proses yang mempunyai tujuan ganda yaitu pengungkapan kelemahan dari software dan evaluasi kualitas software dilihat dari sisi reliability, security, usability, dan correctness. Sebagai catatan, pengujian dan debugging (lokalisasi kesalahan) merupakan dua aktivitas yang berbeda. Proses debugging dimulai setelah proses pengujian dilakukan dan penguji memberikan catatan bahwa software tidak berperilaku sebagaimana yang dispesifikasikan. Debugging atau lokalisasi kesalahan adalah sebuah proses memberikan lokalisasi kesalahan atau kelemahan, memperbaiki koding, dan menguji kembali koding.

Definisi *software* berkualitas adalah *software* yang bebas *error* dan *bug* secara objektif, tepat waktu, sesuai dengan kebutuhan atau keinginan dan dapat dirawat (*maintainable*). Pengertian kata objektif adalah suatu proses pembuktian yang terstruktur, terencana dan tercatat atau terdokumentasi dengan baik. Pendekatan yang objektif sangat diperlukan karena kualitas adalah suatu hal yang tidak nyata dan subjektif. Ia tergantung pada pelanggan dan hal-hal lain yang mempengaruhinya secara keseluruhan. Pelanggan pada proyek pengembangan *software* dapat meliputi pengguna akhir (*end-user*), *tester* dari pelanggan, petugas kontrak dari pelanggan, pihak manajemen dari pelanggan, pemilik saham, *reviewer* dari majalah, dan lain-lain, dimana tiap tipe pelanggan akan mempunyai sudut pandang sendiri terhadap kualitas. *Testing* membuat kualitas dapat dilihat secara objektif, karena *testing* merupakan pengukuran dari kualitas *software*. Dengan kata lain *testing* berarti pengendalian kualitas (*Quality Control* - QC), dan QC mengukur kualitas produk, sedangkan jaminan kualitas (*Quality Assurance* – QA) mengukur kualitas proses yang digunakan untuk membuat produk berkualitas. Walaupun demikian, *testing* tidak dapat memastikan kualitas *software*, namun dapat memberikan kepercayaan atau jaminan terhadap *software* dalam suatu tingkat tertentu. Karena testing merupakan pembuktian dalam suatu kondisi terkendali, dimana *software* difungsikan sebagaimana yang diharapkan pada *test case* yang digunakan. QA dan pengembangan produk adalah aktifitas yang berjalan secara paralel. QA meliputi *review* dari metode pengembangan dan standar, *review* dari semua dokumentasi (tidak hanya untuk standarisasi tapi juga verifikasi dan kejelasan isi). Secara keseluruhan QA juga meliputi validasi kode. Tugas dari QA adalah superset dari

testing. Misinya adalah untuk membantu dalam minimalisasi resiko kegagalan proyek. Tiap individu QA harus memahami penyebab kegagalan proyek dan membantu tim untuk mencegah, mendeteksi dan membenahi masalah.

Peran sebagai penguji perangkat lunak melibatkan partisipasi dalam kegiatan-kegiatan seperti menilai kualitas sebuah produk sistem informasi. Jika terlibat dalam kegiatan-kegiatan penilaian kualitas sistem informasi maka seseorang perlu memahami apa artinya kualitas dan bagaimana menghitung dan mengukurnya. Seseorang penguji harus mampu mendeteksi kelemahan dan cacat pada produk sistem dan dapat melakukan proses pengujian secara efisien dan efektif. Akhirnya perlu memahami sifat pengembangan dan pengujian proses dan bagaimana proses ini berdampak pada kualitas sistem informasi. Kualitas berkaitan dengan sejauh mana suatu sistem, komponen sistem, atau proses memenuhi persyaratan tertentu. Selain itu kualitas berkaitan dengan sejauh mana suatu sistem, komponen sistem, atau proses, bertemu pelanggan, atau pengguna, kebutuhan atau harapan.

Adapun yang dimaksud dengan kualitas produk kerja seperti sistem informasi atau sistem perangkat lunak adalah pemenuhan terhadap kebutuhan fungsional dan kinerja yang didokumentasikan secara eksplisit, pengembangan standar yang didokumentasikan secara eksplisit dan sifat-sifat implisit yang diharapkan dari sebuah sistem yang dibangun secara professional. Untuk mendapatkan produk yang berkualitas, perlu dibangun sebuah sistem pengujian yang berkualitas. Sistem pengujian yang berkualitas dibangun oleh tenaga penguji yang berkualitas. Tenaga penguji yang berkualitas hanya bisa didapatkan melalui pendidikan yang berkualitas.

Software dikatakan berkualitas jika :

a. Memenuhi kebutuhan pemakai –berarti jika software tidak dapat memenuhi kebutuhan pengguna software tersebut, maka software dapat dikatakan tidak atau kurang memenuhi kualitas.
b. Memenuhi standar pengembangan software –berarti jika cara pengembangan software tidak mengikuti metodologi standar, maka hampir dipastikan bahwa kualitas yang baik akan sulit atau bahkan tidak tercapai.
c. Memenuhi sejumlah kriteria implisit berarti jika salah satu kriteria implisit tersebut tidak dapat dipenuhi maka software yang bersangkutan tidak dapat dikatakan memenuhi kualitas yang baik.

Spesialis penguji memiliki peran penting dalam mengembangkan dan menerapkan komponen manajerial. Konsep dan alat-alat untuk membangun kemampuan uji manajemen harus diketahui dengan baik sehingga dapat membantu dalam pengembangan diri sebagai seorang spesialis penguji. Pengembangan, dokumentasi, dan pelembagaan(penetapan) tujuan dan kebijakan mempunyai hubungan yang penting bagi sebuah organisasi. Tujuan / kebijakan tersebut mungkin berhubungan dengan bisnis, teknis, ataupun politik. Oleh karena itu untuk menetapkan tujuan dan kebijakan memerlukan partisipasi dan dukungan dari manajemen bagian atas. Staf teknis dan pihak lain yang berkepentingan juga ikut berpartisipasi dalam penetapan tujuan dan pengembangan kebijakan.

Pengujian mencakup verifikasi logika dasar dari setiap program dan verifikasi untuk melihat bahwa keseluruhan system berjalan sebagaimana mestinya. Seseorang tidak dapat mendalami semua pengujian program karena terdapat jumlah jalur yang bersifat kombinatorial pada suatu program. Pengujiian program-program individual melibatkan suatu upaya untuk meyakinkan bahawa jalur yang paling mungkin terjadi telah bekerja dengan semestinya.

Namun harus dingat suatu ungkapan yang menyatakan bahwa " tidak ada program yang sepenuhnya bersih " dan merencanakan sistem sehingga kesalahan-kesalahan dapat dengan mudah diketahui dan dikoreksi. Cara pengujian program yang paling efektif adalah dengan menggunakan dua orang penguji program yang independent. Bukti-bukti juga menunjukkan bahwa bekerja melalui sebuah terminal dengan sebuah salinan program merupakan cara yang paling disukai oleh setiap penguji independent tersebut.

Software quality assurance (SQA) merupakan sebuah tim kelompok orang dengan pelatihan dan keterampilan yang diperlukan untuk memastikan bahwa semua tindakan yang perlu diambil selama proses development sehingga perangkat lunak yang dihasilkan sesuai dengan persyaratan teknis yang telah ditetapkan. Fungsi utama dari tim SQA adalah untuk memberikan informasi yang relevan dengan kualitas produk kepada pihak manajemen. Untuk mendukung peran ini, para ahli menyarankan bahwa sebuah tim SQA menyiapkan rencana SQA untuk setiap proyek yang menggambarkan audit dan tinjauan yang harus dilakukan, standar-standar yang relevan dengan proyek, tata cara pelaporan dan pelacakan kesalahan, dan dokumen-dokumen mereka akan menghasilkan. Tim juga harus menjelaskan sifat dari hubungan komunikasi antara mereka sendiri, kelompok pengembangan, dan kelompok pengujian. Jika tim SQA mengidentifikasi masalah-masalah yang berdampak pada kualitas produk dan kemudian laporan ini kepada manajemen, maka manajemen harus menangani masalah-masalah dan menyediakan sumber daya untuk mengatasinya sehingga kualitas ditingkatkan.

Pendekatan yang digunakan untuk merancang dan mengembangkan sistem perangkat lunak berdampak pada bagauimana merancangan dan desain penguji tes cocok. Ada dua pendekatan utama untuk pengembangan sistem button-up dan top-down. Sistem yang dikembangkan dengan bahasa prosedural umumnya dianggap sebagai uterdiri dari data pasif dan prosedur aktif. Tingkat abstraksi untuk dua jenis sistem juga agak berbeda. Dalam prosedural tradisional, tingkat terendah abstraksi digambarkan sebagai fungsi atau prosedur yang dilakukan beberapa tugas sederhana. Tingkat berikutnya yang lebih tinggi dari abstraksi adalah sekelompok

prosedur (atau fungsi) yang panggilan satu sama lain dan merupakan persyaratan sistem utama. Unit test, adalah komponen perangkat lunak terkecil yang mungkin diuji. Sebuah unit secara tradisional dipandang sebagai fungsi atau prosedur yang dijalankan dalam bahasa (imperatif) pemrograman prosedural. Tujuan dari unit testing adalah untuk mengisolasi setiap bagian dari program dan menunjukkan bahwa bagian-bagian individu sudah benar. Sebuah tes unit menyediakan kontrak, ketat ditulis bahwa potongan kode harus memenuhi. Akibatnya usaha dapat dibagi menjadi beberapa manfaat atau keuntungan, diantaranya :

a. Pencarian Masalah Awal

Unit test menemukan masalah di awal siklus pengembangan. Pada tes-driven (TDD), yang sering digunakan dalam kedua Extreme Programming dan Scrum, unit test dibuat sebelum kode itu sendiri ditulis. Jika tes lulus, kode yang dianggap lengkap. Unit test yang sama dijalankan terhadap fungsi yang sering sebagai basis kode yang lebih besar dikembangkan baik sebagai kode diubah atau melalui proses otomatis dengan membangun. Jika unit test gagal, dianggap bug baik dalam kode berubah atau tes sendiri. Unit test kemudian memungkinkan letak kesalahan atau kegagalan untuk dapat dengan mudah dilacak. Karena unit test mengingatkan tim pengembangan dari masalah sebelum menyerahkan kode ke penguji atau klien, masih di awal proses pembangunan.

b. Memfasilitasi Perubahan

Unit pengujian memungkinkan programmer untuk kode refactor di kemudian hari, dan pastikan modul masih bekerja dengan benar (misalnya, dalam pengujian regresi). Prosedur ini menulis kasus uji untuk semua fungsi dan metode sehingga setiap kali perubahan menyebabkan kesalahan, dapat dengan cepat diidentifikasi dan diperbaiki. Tes unit tersedia memudahkan para programmer untuk memeriksa apakah sepotong kode masih bekerja dengan benar. Dalam lingkungan pengujian unit terus menerus, melalui praktek yang melekat pemeliharaan berkelanjutan, unit test akan terus secara akurat mencerminkan tujuan penggunaan executable dan kode dalam menghadapi perubahan. Tergantung pada praktek-praktek pembangunan yang ditetapkan dan cakupan unit uji, up-to-the-detik akurasi dapat dipertahankan.

c. Menyederhanakan Integrasi

Unit pengujian dapat mengurangi ketidakpastian dalam unit sendiri dan dapat digunakan dalam pendekatan gaya bottom-up pengujian. Dengan menguji bagian dari sebuah program pertama dan kemudian menguji jumlah bagian-bagiannya, pengujian integrasi menjadi lebih mudah. Hirarki yang rumit unit test tidak pengujian integrasi sama. Integrasi dengan unit perangkat harus dimasukkan dalam tes integrasi, tapi tidak dalam unit test Integrasi pengujian biasanya masih sangat bergantung pada manusia pengujian manual;. Pengujian tingkat tinggi atau global-scope bisa sulit untuk mengotomatisasi, sehingga pengujian manual sering muncul lebih cepat dan lebih murah.

d. Dokumentasi

Unit pengujian menyediakan semacam dokumentasi hidup dari sistem. Pengembang yang ingin belajar apa fungsi disediakan oleh unit dan bagaimana menggunakannya dapat melihat unit test untuk mendapatkan pemahaman dasar API unit. Satuan kasus uji mewujudkan karakteristik yang sangat penting untuk keberhasilan unit. Karakteristik ini dapat menunjukkan penggunaan yang tepat / tidak tepat unit serta perilaku negatif yang akan terjebak oleh unit. Sebuah kasus uji unit, dalam dan dari dirinya sendiri, mendokumentasikan karakteristik kritis, meskipun lingkungan pengembangan perangkat lunak yang tidak hanya mengandalkan kode untuk mendokumentasikan produk dalam pembangunan. Sebaliknya, dokumentasi naratif biasa lebih rentan terhadap melayang dari pelaksanaan program dan dengan demikian akan menjadi usang (misalnya, perubahan desain, terobosan fitur, praktek santai dalam menjaga dokumen up-to-date).

e. Disain

Ketika perangkat lunak dikembangkan dengan menggunakan pendekatan uji-driven, tes unit bisa menggantikan desain formal. Setiap unit test dapat dilihat sebagai elemen desain menentukan kelas, metode, dan perilaku yang dapat diamati. Contoh Jawa berikut akan membantu menggambarkan hal ini. Berikut ini adalah kelas tes yang menentukan sejumlah elemen pelaksanaan. Pertama, bahwa harus ada sebuah antarmuka yang disebut Adder, dan kelas yang mengimplementasikan dengan konstruktor nol-argumen disebut AdderImpl. Ini melanjutkan dengan menegaskan bahwa antarmuka Adder harus memiliki metode yang disebut menambahkan, dengan dua parameter bilangan bulat, yang mengembalikan bilangan bulat lain. Hal ini juga menentukan perilaku dari metode ini untuk berbagai nilai-nilai kecil.

3.2.1 Prosedur

Pengujian unit pada umumnya merupakan perkembangan dari langkah pengkodean. Setelah program sumber dikembangkan, ditinjau kembali dan diverifikasi untuk sintaknya, maka perancangan test case di mulai.

Peninjauan kembali perancangan informasi akan menyediakan petunjuk untuk menentukan test case. Karena modul bukan program yang berdiri sendiri, maka driver (pengendali) dan atau suatu perangkat lunak harus dikembangkan untuk tiap-tiap pengujian unit.

3.2.2. Kelas

Unit testing menggunakan sebuah kelas (test class) yang akan menguji kelas lain. Sebuah test class hanya akan menguji satu class saja. Dalam sebuah test class terdapat beberapa test method yang digunakan untuk menguji method-method pada kelas lain. Sebuah method dapat diuji oleh lebih dari satu test method.

3.2.3. Metode sebagai unit

Dalam kegiatan pengujian perangkat lunak, terdapat 2 metode dasar yang umum digunakan, yaitu *white-box testing* dan *black-box testing*.

a. *White-box testing* adalah pengujian yang memperhitungkan mekanisme internal sistem atau komponen (IEEE, 1990). *White-box testing* juga dikenal sebagai*structural testing*,*clear-box testing*, *dan glass-box testing*.
   Konotasi *clear-box*atau *glass box* mengindikasikan bahwa kita memiliki visibilitas penuh akan kerja internal perangkat lunak, terutama pada logika dan struktur kodenya. Adapun kelebihan metode ini adalah mampu mendeteksi kesalahan logika, kesalahan ketik pada kode sumber dan ketidaksesuaian asumsi. Sementara kekurangannya adalah pada perangkat lunak yang tergolong besar, dianggap sebagai metode yang boros karena melibatkan sumber daya yang besar.
b. *Black-box testing* atau disebut juga *functional testing*merupakan pengujian yang mengabaikan mekanisme internal sebuah sistem atau komponen, dan berfokus semata-mata pada *output* yang dihasilkan dalam menanggapi *input* dan kondisi eksekusi yang dipilih (IEEE, 1990). Di dalam metode ini, penguji tidak atau tidak seharusnya memiliki akses ke kode sumber. Kode ini dianggap sebagai suatu "kotak hitam" yang dapat dimasukkan suatu informasi dan kemudian akan menghasilkan suatu keluaran yang diharapkan. Adapun kelebihan metode ini adalah dapat menguji keseluruhan fungsionalitas, dapat menemukan cacat lebih cepat, dan dapat memilih *subset test* secara efektif dan efisien sehingga dapat membantu memaksimalkan *testing investment*. Sementara itu, kekurangannya adalah penguji tidak akan pernah yakin apakan perangkat lunak yang diuji benar-benar sudah lolos uji, apakah sudah mencakup semua possible input atau belum.

Tujuan utama untuk pengujian unit adalah memastikan bahwa setiap unit perangkat lunak unit individu berfungsi sesuai dengan spesifikasinya. Sumber daya harus dialokasikan, dan kasus-kasus uji harus dikembangkan, menggunakan kedua strategi pengujian desain black box dan white box. Unit harus diuji oleh penguji indevenden (orang lain selain pengembang) dan hasil tes dan catat ditemukan harus dicatat sebagai bagian dari sejarah unit (dibuat umum). Untuk mempersiapkan tes unit pengembang/penguji harus melakukan beberapa tugas antara lain : Rencana pendekatan umum untuk pengujian unit, design uji kasus, uji prosedure (ini akan melekat pada rencana uji), Menentukan hubungan antara tes, dan menyiapkan kode tambahan yang diperlukan untuk pengujian unit. Sebuah rencana unit umum uji harus disiapkan. Ini dapat dibuat sebagai komponen dari rencana uji master atau sebagai rencana berdiri sendiri. Dokumen yang memberikan masukan bagi rencana unit adalah rencana proyek, serta persyaratan, spesifikasi, dan dokumen desain yang menggambarkan unit target. Komponen rencana unit tes dijelaskan secara rinci standar IEEE perangkat lunak unit testing standart ini kaya akan informasi dan merupakan panduan yang sangat baik untuk perencanaan uji. Dalam setiap tahap serangkaian kegiatan berbasis ditugaskan pada yang ditemukan dalam standar IEEE unit test. Tahap 1 : Pendekatan unit jelaskan test dan resiko dalam fase pengujian unit perencanaan pendekatan umum untuk unit testing diuraikan perencanaan tes sebagai berikut : mengidentifikasi resiko uji; menjelaskan teknik yang digunakan untuk merancang test case untuk unit, menjelaskan untuk teknik yang akan digunakan untuk validasi data dan pencatatan hasil uji, dan menjelaskan persyaratan untuk memanfaatkan pengujian dan perangkat lunak lain yang interface dengan unit yang akan diuji, misalnya, setiap benda harus diperlukan untuk pengujian unit berorientasi objek. Tahap 2 : Identifikasi fitur unit akan diuji, fase ini merupakan informasi dari spesifikasi desain unit dan deskripsi rinci. Perencanaan menentukan fitur dari setiap unit akan diuji jika beberapa fitur tidak akan dicakup oleh tes. Tahap 3:Tambahkan tingkat rencana, pada tahap ini perencanaan menyempurnakan rencana sebagaimana yang dihasilkan dalam dua tahap sebelumnya. Perencanaan menambahkan rincian baru ke sumber, pendekatan, dan bagian penjadwalan rencana uji unit. Tes dokumen terkait yang akan dibutuhkan untuk tugas ini, misalnya tes log dan laporan uji insiden, harus dijelaskan dan direferensikan standar untuk dokumen-dokumen yang disediakan.

Pengujian tahap perancangan bertujuan untuk menguji struktur perangkat lunak yang diturunkan dari kebutuhan**.** Kebutuhan yang bersifat umum dirinci menjadi bentuk yang lebih spesifik. Faktor-faktor pengujian yang dilakukan pada tahap perancangan : perancangan yang berkaitan dengan kebutuhan; kesesuaian perancangan dengan metodologi dan teori; portabilitas rancangan; perancangan perawatan; kebenaran rancangan berkaitan dengan fungsi dan aliran data; dan kelengkapan perancangan antar muka.

Jika kelas adalah komponen dipilih, penguji mungkin perlu menangani masalah masalah khusus yang berkaitan dengan pengujian dan pengujian ulang komponen ini.

a. Pengujian kelas-I
   Salah satu fitur yang paling menguntungkan pengembangan berorientasi obyek adalah enkapsulasi. Ini adalah teknikyang dapat digunakan untuk menyembunyikan informasi. Sebuah unit program, dalam hal ini kelas, dapat dibangu dengan antar muka publik yang jelas yang menyatakan layanan untuk kelas klient. Pelaksanaan layanan bersifat pribadi.
b. Pengujian Kelas-II
   Sangat sering menganggap bahwa sekali satu metode dalam super class telah diuji, tidak perlu di uji ulang di sub class yang mewarisi hal itu. Namun dalam beberapa kasus, metode yang digunakan dalam kontek yang berbeda adalah sub class dan akan di uji ulang.
   Tidak hanya metode lokal yang didefenisikan baru harus di uji ulang, tetapi merancang satu set baru pengujian kasus mungkin diperlukan. Hal ini dikarenakan struktural yang berbeda.

Pengujian kotak hitam sering dikenal sebagai pengujian acak. Pengujian produk sistem informasi jika menerapkan pengujian kotak hitam untuk mengungkapkan kelemahan sistem maka akan melibatkan konsumen dalam jumlah banyak. Penggunaan dari uji acak masukan dapat menghemat sebagian dari waktu dan usaha yang diperlukan. Pemilihan masukan test secara acak mempunyai kesempatan sangat kecil dalam menghasilkan satu set data pengujian yang efektif .Sudah ada banyak diskusi di dalam dunia pengujian tentang apakah pernyataan seperti itu tepat.

Efektivitas yang relatif dari pendekatan masukan pengujian acak melawan suatu pendekatan lebih terstruktur dalam menghasilkan masukan tes menjadi pokok bahasan banyak dokumen riset. Ujian jenis ini dapat sangat bermanfaat terutama di tingkatan sistem. Biasanya penguji menetapkan suatu cakupan untuk generator nilai yang acak, atau masukan pengujian dihasilkan menurut suatu distribusi statistik yang dihubungkan dengan suatu pola dari pemakaian. Namun waktu pengujian harus dibatasi dalam satu rentang tertentu dan harus ada dokumentasi untuk setiap tahap pengujian kotak hitam. Pengujian kotak hitam harus dibatasi jumlah konsumen yang terlibat dalam pengujian. Hal ini dilakukan karena mempertimbangkan lamanya masa pengujian yang akan diberikan terutama jika sistem informasi yang diuji adalah sistem yang berukuran besar. Bahkan dalam pemilihan konsumen sebagai pihak yang melakukan pengujian kotak hitam terkadang harus dipilih secara acak dan per kategori.

Kelebihan dari jenis pengujian kotak hitam adalah mampu mengungkapkan kelemahan sistem secara tidak terduga sedangkan kelemahannya adalah hasilnya kurang bisa dipertanggungjawabkan karena sifatnya yang random dan sangat membutuhkan waktu yang banyak.

Seorang analis sistem sangat dibutuhkan dalam pengujian kotak putih karena analis sistem adalah orang yang mengerti sistem secara teknis. Hal yang diharapkan dari seorang analis sistem ketika melaksanakan pengujian adalah :

a. Bekerja dengan para pemakai untuk mengenali adanya kebutuhan-kebutuhan dasar terhadap sebuah system pengolahan informasi yang baru dan mempersiapkan sebuah survey pendahuluan. Memperhatikan tugas analisis dan desain sistem maupun hubungan antara para pendesain dan para pemakai.
b. Mengembangkan sebuah pengalaman menyeluruh mengenai prosedur-prosedur pengolahan system informasi saat itu melaluo studi keputusan dan aliran-aliran informasi, teknik-teknik pengumpulan data , termasuk studi dokumen-dokumen, wawancara dan kuisiner.
c. mempersiapkan studi kelayakan bersama para pemakai, dan mengemukakan alternative untuk sebuah system informasi yang baru.
d. Selama tahap-tahap di atas, analis juga mengajar para pemakai mengenai system informasi.
e. Membantu pemakai mendesain sebuah system untuk alternative system yang terpilih
f. Mengkonversikan file dan logika system ke dalam struktur file terinci dan spsesifikasi pemrograman.
g. Bekerja dengan para pemakai untuk mengembangkan dokumentasi yang dijelaskan pada bab 18.
h. Berperan serta dalam memanajmen teknis proyek, terutama dalam pengujian program dan system.
i. Bersama para pemakai , membuat rencana pemasangan system baru.
j. Memandu dan berperan serta dalam konversi actual dari system lama ke system baru.
k. Mengadakan penilaian pasca pelaksanaan bersama dengan para pemakai.

Seorang analisis sistem harus mampu menerapkan pengetahuan tentang sistem secara terstruktur dan biasanya dilakukan secara bertahap dan per bagian. Biasanya masukan sistem disiapkan oleh analis sistem dan jumlah masukan sudah dipilih secara teknis oleh analis sistem. Kelebihan dari metode ini adalah waktu yang singkat dan pengungkapan kelemahan lebih objektif. Kelemahannya adalah kesulitan dalam mencari tenaga analis sistem yang mengerti kerja sistem dengan baik. Penerapan pengujian kotak putih dan hitam harus dilakukan pada setiap pengujian unit dan modul gabungan.Selama kegiatan desain sistem, modul-modul program dasar diuji oleh masing-masing pemrogram yang ditulis. Pada tahap ini para pemrogram biasanya menata data pengujian mereka sendiri atau menggunakan sebuah program yang menghasilkan data pengujian. Bila seseorang mengikuti pendekatan-pendekatan pemrograman atas bawah maka perlu membuat driver program untuk menguji

modul modul pada tingkat yang lebih rendah. Dengan pemrograman atas bawah dapat digunakan modul dummi untuk memanggil program program tingkat rendah yang belum dikoding.

Pengujian unit dengan data dari pemrograman memang diperlukan , tetapi itu belum cukup. Pemrograman individual dan pendekatan tim dapat bermanfaat karena dapat memberikan input daripada pemrogram lainnya yang mungkin juga memikirkan adanya logika yang terlewat tersebut, Pertemuan-pertemuan peninjauan terstruktur juga dapat menemukan berbagai kelalaian. Sedangkan pada pengujian modul gabungan harus diterapkan secara keseluruhan. Setelah modul-modul lolos dari pengujian unit, unit-unit tersebut kemudian digabungkan untuk diuji . Dalam pengujian modul gabungan sekali lagi ditekankan perlunya pembangkitan data pengujian independent. Hal yang penting adalah melibatkan para pemakai dalam membuat data pengujian supaya data tersebut lebih sedikit dibuat dan lebih realistis.

Pengujian yang dijelaskan di atas mencurahkan perhatiannya kepada program-program. Kegiatan utama lainnya yang terjadi sekaligus bersama pengembangan program-program adalah desain dan pengujian prosedur-prosedur manual. Prosedur-prosedur manual kadang-kadang diabaikan, padahal mereka dapat menentukan keberhasilan suatu system. Adalah penting untuk mendesain beberapa uji percobaan sebelum mereka menjadi percobaan –percobaan yang nyata. Juga sangat penting bagi para pemakai untuk menetapkan data yang aneh atau tidak biasa untuk menguji keistimewaan pengeditan dan pemeriksa kesalahaan system tersebut.

Dalam banyak kasus uji adalah memungkinkan untuk melakukan pengujian-pengujian pararel selama masa konversi, yakni melakukan konversi dari system lama ke system baru sambil meneruskan metode pengolahan lama secara pararel. Hasil-hasil dari dua system tersebuit dibandingkan sebagai pemeriksaan lanjutan bagi system baru sebelum sistem yang lama dihentikan. Pengujian pararel menawarkan banyak keuntungan tetapi mereka memberikan beban berat kepada pamakai untuk menjalankan dua system sekaligus. Pengujian pararel hanya dapat diterima untuk periode waktu yang singkat. Dalam suatu konversi harus mengusahakan suatu transisi setahap demi setahap menuju sistem baru.

Sasaran utama dari usaha melibatkan para pemakai yang memakan waktu dan membutuhkan ketelitian tersebut adalah untuk mempermudah jalan mencapai tahap terakhir ini. Pertama, peran serta membantu pemakai menguasai beberapa pengendalian pengolahan. Ini berarti mengurangi jumlah kekuatan yang berpindah ke departemen computer. Peran serta juga dapat membuat seseorang menjadi terbiasa dengan system dan memberikan konstribusi untuk latihan. Karena peran serta, maka lebih banyak cirri-ciri yang penting dapat dicakup di dalam sebuah system baru dan hal ini akan akan memberikan manfaat yang lebih banyak kepada para pemakai. Seorang analis system dan departemen computer bertindak sebagai narasumber untuk membantu suatu tim desain dan mengatur bagian-bagian usaha desain yang lebih teknis.

Pengujian blackbox merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program. Ciri-Ciri pengujian blackbox adalah (1) Pengujian black box berfokus pada kebutuhan fungsional pada software; (2) berdasarkan pada spesifikasi kebutuhan dari software; (3) Pengujian black box bukan teknik alternatif daripada pengujian white box. Lebih daripada itu, ia merupakan pendekatan pelengkap dalam mencakup error dengan kelas yang berbeda dari metode pengujian white box; (4) Pengujian black box melakukan pengujian tanpa pengetahuan detil struktur internal dari sistem atau komponen yang dites. juga disebut sebagai behavioral testing, specification-based testing, input/output testing atau functional testing.

Kategori error yang akan diketahui melalui pengujian black box yaitu (1) Fungsi yang hilang atau tak benar, (2) Error dari antar-muka; (3) Error dari struktur data atau akses eksternal database; (4) Error dari kinerja atau tingkah laku; (5) Error dari inisialisasi dan terminasi. Tidak seperti metode whitebox yang dilaksanakan di awal proses, uji coba blackbox diaplikasikan dalam beberapa tahapan berikutnya.karena uji coba blackbox dengan sengaja mengabaikan struktur control, sehingga perhatiannya difokuskan pada informasi domain.Pada pengujian blackbox terdapat jenis teknik desain tes yang dapat dipilih berdasarkan pada tipe testing yang akan digunakan, yang diantaranya : (1)Equivalence Class Partitioning yang merupakan metode pengujian black box yang membagi domain masukan dari suatu program ke dalam kelas-kelas data, dimana test cases dapat diturunkan [BCS97a]. Equivalence partitioning berdasarkan pada premis masukan dan keluaran dari suatu komponen yang dipartisi ke dalam kelas-kelas, menurut spesifikasi dari komponen tersebut, yang akan diperlakukan sama (ekuivalen) oleh komponen tersebut. Dapat juga diasumsikan bahwa masukan yang sama akan menghasilkan respon yang sama pula. Nilai tunggal pada suatu partisi ekuivalensi diasumsikan sebagai representasi dari semua nilai dalam partisi.

Analisa partisi pada Equivalence Partitioning Black Box adalah Tester menyediakan suatu model komponen yang dites yang merupakan partisi dari nilai masukan dan keluaran komponen, lalu masukan dan keluaran dibuat dari spesifikasi dari tingkah laku komponen. Setelah itu partisi adalah sekumpulan nilai, yang dipilih dengan suatu cara dimana semua nilai di dalam partisi, diharapkan untuk diperlakukan dengan cara yang sama oleh komponen (seperti mempunyai proses yang sama). Partisi untuk nilai valid dan tidak valid harus ditentukan. Aplikasikan paduan untuk derivasi dari kelas – kelas yang akuivalen, kasus uji untuk setiap domain input data item dapat dibentuk dan dieksekusi. Kasus uji dipilih sehingga sejumlah atribut dari equivalence class

dieksekusi sekali saja. Sejumlah besar kesalahan cenderung terjadi dalam batasan domain input daripada nilai tengah. Untuk alasan ini Boundary value alaysis dibuat sebagai ujicoba. BVA mengarahkan pada pemilihan kasus uji yang melatih nilai – nilai batas. BVA merupakan desain teknik kasus uji yang melengkapi equivalence partitioning. Dari pada memfokuskan hanya pada kondisi input, BVA juga menghasilkan kasus uji dari domain output. Panduan untuk BVA hampir sama pada beberapa bagian seperti yang disediakan untuk equivalence partitioning :

a. Jika kondisi input menspesifikasikan kisaran yang dibatasi oleh nilai a dan b, kasus uji harus dibuat dengan nilai a dan b, sedikit diatas dan sedikit dibawah nilai a dan b.
b. Jika kondisi input menspesifikasikan sejumlah nilai, kasus uji harus dibuat dengan melatih nilai maksimum dan minimum,juga nilai – nilai sedikit diatas dan dibawah nilai maksimum dan minimum tersebut.
c. Aplikasikan paduan 1 dan 2 untuk kondisi output. Sebagai contoh asumsikan table temperature Vs table tekana sebagai output yang menghsilkan nilai maksimum dan minimum yang mungkin untuk tabel masukan
d. Jika struktur data program internal telah mendskiripsikan batasan (misalkan array ditetapkan 100), maka desain kasus uji yang akan melatih struktur data pada batasan tersebut.

Kebanyaka pengembangan software secara intuitif melakukan BVA pada beberapa tingkatan. Dengan mengaplikasikan paduan diatas, uji coba batasan akan lebih lengkap, selain itu memiliki kemungkinan pendeteksian kesalahan yang lebih tinggi.

Cause-Effect Graphing merupakan desain teknik kasus ujicoba yang menyediakan representasi singkat mengenai kondisi logikal dan aksi yang berhubungan. Tekniknya mengikuti 4 tahapan berikut

a. Causes dan effects didaftarkan untuk modul dan identifier yang ditunjukan untuk masing – masing.
b. Causes – effects graph
c. Graph dikonversikan kedalam tabel keputusan
d. Aturan tabel keputusan dikonversikan kedalam kasus uji.

Dalam beberapa situasi dimana keandalan suatu software amat kritis, beberapa aplikasi sering menggunakan software dan hardware ganda. Ketika software redundant dibuat, tim pengembangan software lainnya membangun versi independen dari aplikasi menggunakan spesifikasi yang sama. Setiap versi dapat diuji dengan data uji yang sama untuk memastikan seluruhnya menyedikan output yang sama. Kemudian seluruh versi dieksekusi secara paralel dengan perbandingan hasil real time untuk memastikan konsistensi.
Dianjuran bahwa versi independen suatu software untuk aplikasi yang amat kritis harus buat, walaupun nantinya hanya satu versi saja yang akan digunakan dalam system. Versi independen ini merupakan basis dari teknik black box testing yang disebut Comparison testing atau back to back testing.

Ketika multiple implementasi dari spesifikasi yang sama telah diproduksi. Kasus uji didesain dengan menggunakan teknik black box yang lain (misalnya equivalence partitioning) disediakan sebagai input untuk setiap versi dari software. Jika setiap outputnya sama, diasumsikan implemetasinya benar, jika tidak, setiap versi diperiksa untuk menentukan jika kerusakan terdapat pada satu atau lebih versi yang akan bertanggung jawab atas perbedaan tersebut. Jika setiap spesifikasi dari seluruh versi telah dibuat dalam kesalahan, maka seluruh versi akan mereflesikan kesalahan. Sebagai tambahan, jika setiap versi independent memberikan hasil identik , tetapi salah, uji coba hasil dan kondisi akan gagal untuk mendeteksi kesalahan.

Pada melakukan pengujian kotak hitam dan kotak putih, maka yang harus dilakukan adalah melakukan dokumentasi setiap kelemahan sistemyang ditemukan, lokasi kelemahan sistem dan solusi-solusi dalam mengatasi kelemahan tersebut.

Pekerjaan terakhir adalah mengeksekusi rencana pengujian dan memasikan bahwa bagian pengujian dapat menangani masalah yang timbul dengan bantuan beberapa pedoman dari ahli pengujian. Pekerjaan sebagai manajer pengujian sangat penting ketika bergerak ke tahap implementasian rencana pengujian dan mengumpulkan data yang berhubungan dengan rencana tersebut.

Ketika memulai sebuah eksekusi, kebanyakan pendekatan terorganisir tidak menyelamatkan proyek yang sedang berjalan. Pada awalnya sebuah rencana pengujian : pengujian kasus, pengujian tools, pengujian arsitektur maupun mengukur pengujian yang tersembunyi dan semua komponen program yang telah dikembangkan adalah proaktif dan ketika sempurna relative menjadi objek yang statil. Pengujian merupakan resiko manajemen dan tidak dapat mengatur resiko tanpa data. Pengujian terhadap data mentah cenderung tidak berbentuk, membingungkan dan sangat sulit untuk dikategorikan.

Pengujian/testing pada akhir pengembangan jadwal, dan setiap kesalahan penundaan dan kenaikan dapat dibuktikan sendiri dalam proses pengujian, berkonsentrasi selama perjalanan mencari bug yang merusak dan menyebabkan kebingungan serta kehilangan data.

4. KESIMPULAN

Setelah melakukan analisis penggunaan metode pengujian kotak hitam dan kotak hitam maka dapat disimpulkan bahwa metode pengujian kotak hitam dan kotak putih ibarat dua sisi uang logam. Metode pengujian kotak hitam dan kotak putih harus dikerjakan bersisian dan saling melengkapi untuk mengungkapkan kelemahan sebuah sistem informasi. Pengujian kotak hitam dan kotak putih sudah harus direncanakan pada saat sebuah sistem informasi mulai direncanakan untuk dikembangkan. Hal ini berarti pihak-pihak yang terlibat dalam pengujian juga sudah harus disiapkan dari awal.

Membangun sebuah sistem informasi dengan menitikberatkan langkah pengujian serta melibatkan semua pihak yang terlibat dengan produk tersebut menunjukkan tingkat keseriusan seorang pengembang dalam membangun dan menghasilkan produk sistem informasi yang berkualitas. Kualitas sistem informasi mengacu pada sejauh mana hasil dari sistem informasi memenuhi kebutuhan pengguna. Kebutuhan pengguna sistem informasi adalah kegunaan dan keamanan. Kegunaan dan keamanan sistem informasi dapat dicapai jika dalam melakukan pengujian kotak hitam dan kotak putih, penguji memperhatikan verifikasi dan validasi, Proses verifikasi dan validasi memegang kunci penting dalam penentuan kualitas sistem.

Langkah-langkah pengujian khususnya pengujian kotak putih dan kotak hitam adalah melibatkan semua pihak sehingga diharapkan produk sistem informasi yang dihasilkan adalah produk yang memang sesuai kebutuhan. Proses merupakan entitas aktif yang harus dikelola dengan baik sehingga tujuan pengujian dapat dicapai dengan utuh. Selain itu kualitas memegang peranan penting dalam menentukan bagaimana pengujian dilakukan. Pencapaian kualitas adalah langkah untuk menghindari biaya kerugian yang lebih besar setelah produk sistem informasi dilemparkan ke pasar.

5. SARAN

Pengujian adalah sebuah proses yang memiliki aspek ekonomis, teknis dan manajerial. Aspek ekonomis menunjukkan kenyataan bahwa sumber daya dan waktu harus tersedia dalam usaha pengujian yang dilakukan. Pada kenyataannya, pengujian yang lengkap dalam banyak kasus tidak bisa dilakukan karena tekanan ekonomis. Sebuah organisasi harus menstrukturkan dengan jelas proses pengujian yang akan dilakukan sehingga proses pengujian dalam dilakukan dengan anggaran yang jelas dan tepat waktu, serta dapat memenuhi kebutuhan user.

Sedangkan aspek teknis dari pengujian menunjukkan teknik-teknik, metode-metode, ukuran-ukuran dan perangkat-perangkat digunakan untuk menjamin bahwa software yang dites adalah bebas dari kelemahan-kelemahah dan handal untuk dijalankan dalam kondisi dan tekanan yang dimungkinkan ketika dioperasikan. Sehingga diharapkan dalam membangun sistem informasi selain penggunaan kotak hitam dan kotak putih maka harus diterapkan juga jenis-jenis pengujian yang lain. Kegiatan dari pengujian harus seratus persen direncanakan.

Setiap pengujian berangkat dari prinsip bahwa tidak ada produk sistem informasi yang sempurna dan selalu ada kelemahan sistem yang bisa ditemukan. Sehingga diharapkan bahwa dalam menguji produk sistem informasi harus menyertakan metode pengujian lain selain metode pengujian kotak hitam dan kotak hitam.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada segenap civitas akademika Sekolah Tinggi Manajemen Informatika dan Komputer Widya Dharma yang telah memberikan dukungan baik data maupun moril untuk penelitian ini.

DAFTAR PUSTAKA

[1] Al Fatta, Hanif. (2007). *Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern*. Andi. Yogyakarta.

[2] Hall, James, A dan Singleton, Tommie.(2007). *Audit Teknologi Informasi dan Assurance.* Salemba Empat. Jakarta

[3] Effy oz.(2009).Management Information Systems Sixth Edition.Cengage Learning.United State of America.

[4] Humdiana dan Indrayani, Evi. (2006). *Sistem Informasi Manajemen*. Graha Ilmu. Yogyakarta.

[5] Irwanto, Djon. (2007). *Membangun Object Oriented Software dengan Java dan Object Database*. P.T. Elex Media Komputindo. Jakarta.

[6] Jogiyanto, H. M (2009). *Sistem Teknologi Informasi : Pendekatan Terintegrasi: Konsep Dasar, Teknologi, Aplikasi, Pengembangan, dan Pengelolaan*. Andi Offset. Yogyakarta.

[7] Laudon, Kenneth C., dan Jane P. Laudon, 2011, Sistem Informasi Manajemen (Judul asli : *Management Information System*), Ed.10, Penerjemah : Sungkono, Chriswan dan Eka P, Machmudin. Salemba Empat, Jakarta.

[8] Kadir, Abdul, 2008, Pengenalan Sistem Informasi, Ed.1, Cetakan 5, Andi, Yogyakarta.

[9] Burstain, Ilene, 2002, Practical Software Testing, Springer, United States of America.

[10] Ladjamudin, Al Bahra Bin. (2005). *Analisis dan Desain Sistem Informasi.* Graha Ilmu. Yogyakarta.

[11] Kendall, Kenneth E dan Kendall, Julie E. (2003). *Systems Analysis and Design*. Prentice Hall, Inc, Upper Saddle River.

[12] Sutabri, Tata. (2004). *Analisa Sistem Informasi*. Andi, Yogyakarta.