BAB 8

JAMINAN KUALITAS PERANGKAT LUNAK
Software Quality Assurance [SQA]

Jaminan kualitas perangkat lunak adalah aktivitas pelindung yang diaplikasikan pada seluruh proses perangkat lunak.

SQA meliputi :

1. pendekatan manajemen kualitas
2. teknologi rekayasa perangkat lunak yang efektif (metode dan peranti)
3. kajian teknik formal yang diaplikasikan pada keseluruhan proses perangkat lunak
4. strategi pengujian *multitiered* (deret bertingkat)
5. kontrol dokumentasi perangkat lunak dan perubahan
6. prosedur untuk menjamin kesesuaian dengan standar pengembangan perangkat lunak
7. mekanisme pengukuran dan pelaporan.

Kontrol Kualitas

Kontrol kualitas merupakan serangkaian pemeriksaan, kajian, dan pengujian yang digunakan pada keseluruhan siklus pengembangan untuk memastikan bahwa setiap produk memenuhi persyaratan yang ditetapkan.

Konsep kunci kualitas kontrol adalah bahwa semua produk kerja memiliki spesifikasi yang telah ditentukan dan dapat diukur dimana kita dapat membandingkan output dari setiap proses.
Kalang (loop) menjadi penting untuk meminimalkan cacat yang dihasilkan.

Jaminan Kualitas

Jaminan kualitas terdiri atas fungsi auditing dan pelaporan manajemen.

Tujuan jaminan kualitas adalah :
untuk memberikan data yang diperlukan oleh manajemen untuk menginformasikan masalah kualitas produk, sehingga dapat memberikan kepastian & konfidensi bahwa kulitas produk dapat memenuhi sasaran.

Biaya Kualitas

Biaya kualitas menyangkut semua biaya yang diadakan untuk mengejar kualitas atau untuk menampilkan kualitas yang berhubungan dengan aktivitas.

Studi tentang biaya kualitas dilakukan untuk memberikan garis dasar bagi biaya kualitas yang sedang digunakan, untuk mengidentifikasi kemungkinan pengurangan biaya kualitas serta memberikan basis perbandingan yang ternormalisasi.

Biaya kualitas dapat dibagi ke dalam biaya-biaya yang dihubungkan dengan :

a. pencegahan
b. penilaian
c. kegagalan.

a) Biaya pencegahan meliputi :

- Perencanaan
- Kajian teknis formal
- Perlengkapan pengujian
- Pelatihan

b) Biaya penilaian meliputi :

- Inspeksi in-proses dan interproses
- Pemeliharaan dan kalibrasi peralatan
- Pengujian

c) Biaya kegagalan

Biaya kegagalan adalah biaya yang akan hilang bila tidak ada cacat yang muncul sebelum produk disampaikan kepada pelanggan.

Biaya ***kegagalan internal*** adalah
biaya yang diadakan bila kita mendeteksi suatu kesalahan dalam produk sebelum produk dipasarkan.

Biaya kegagalan internal meliputi:

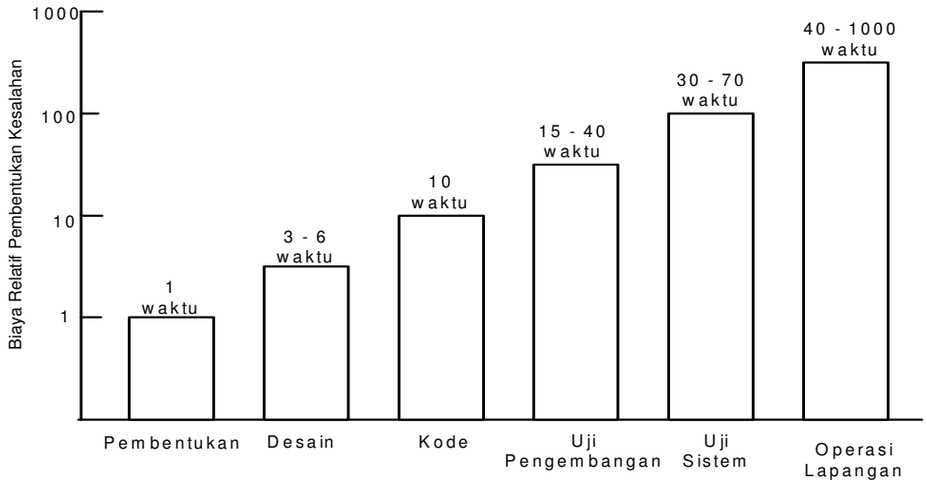
- Pengerjaan kembali
- Perbaikan
- Analisis mode kegagalan

Biaya ***kegagalan eksternal*** adalah
biaya yang berhubungan dengan cacat yang ditemukan setelah produk disampaikan kepada pelanggan.

Biaya kegagalan eksternal meliputi:

- Resolusi keluhan
- Penggantian dan pengembalian produk
- Dukungan *help line*
- Kerja jaminan

Biaya relatif mendapatkan dan membetulkan cacat bertambah secara dramatis pada saat kita melangkah dari pencegahan ke pendeteksian dan dari kegagalan internal ke kegagalan eksternal.



Gambar 8.1 *Biaya Relatif pembetulan kesalahan*

JAMINAN KUALITAS PERANGKAT LUNAK

Kualitas perangkat lunak didefinisikan sebagai:
Konformansi terhadap kebutuhan fungsional dan kinerja yang dinyatakan secara eksplisit, standar perkembangan yang didokumentasikan secara eksplisit, dan karakteristik implisit yang diharapkan bagi semua perangkat lunak dikembangkan secara profesional.

definisi tersebut berfungsi untuk menekankan tiga hal penting, yaitu:

1. Kebutuhan perangkat lunak merupakan fondasi yang melaluinya *kualitas* diukur.

2. Standar yang telah ditentukan menetapkan serangkaian kriteria pengembangan yang menuntun cara perangkat lunak direkayasa.

3. Ada serangkaian *kebutuhan implisit* yang sering dicantumkan (misalnya kebutuhan akan kemampuan pemeliharaan yang baik).

Kelompok SQA berfungsi sebagai perwakilan *in-house* pelanggan, yaitu orang yang akan melakukan SQA harus memperhatikan perangkat lunak dari sudut pandang pelanggan.

Kelompok SQA harus dapat menjawab pertanyaan-pertanyaan dibawah ini untuk memastikan bahwa kualitas perangkat lunak benar-benar terjaga.

- Apakah perangkat lunak cukup memenuhi faktor kualitas
- Sudahkah pengembangan perangkat lunak dilakukan sesuai dengan standar yang telah ditetapkan sebelumnya?
- Sudahkah disiplin teknik dengan tepat memainkan perannya sebagi bagian dari aktivitas SQA?

Aktivitas SQA

Jaminan kualitas perangkat lunak terdiri dari berbagai tugas yang berhubungan dengan dua konstituen yang berbeda :

- perekayasa perangkat lunak yang mengerjakan kerja teknis

- kelompok SQA yang bertanggung jawab terhadap perencanaan jaminan kualitas, kesalahan, penyimpanan rekaman, analisis, dan pelaporan.

Tugas kelompok SQA adalah

membantu tim rekayasa perangkat lunak dalam pencapaian produk akhir yang berkualitas tinggi.
Aktivitas yang dilakukan (atau difasilitasi) oleh kelompok SQA yang independen:

- *Menyiapkan rencana SQA untuk suatu proyek*. Rencana tersebut mengindentifikasikan hal-hal berikut:
  - Evaluasi yang dilakukan
  - Audit dan kajian yang dilakukan
  - Standar yang dapat diaplikasikan pada proyek
  - Prosedur untuk pelaporan & penelusuran kesalahan
  - Dokumen yang dihsilkan oleh kelompok SQA
  - Jumlah umpan balik yang diberikan pada tim proyek perangkat lunak

- *Berpartisipasi dalam pengembangan deskripsi proses pengembangan proyek*.

- *Mengkaji aktivitas rekayasa perangkat lunak untuk memverifikasi pemenuhan proses perangkat lunak yang sudah ditentukan*.

- *Mengaudit produk kerja perangkat lunak yang ditentukan untuk membuktikan kesesuaian dengan produk kerja yang ditentukan tersebut sebagai bagian dari proses perangkat lunak*.

- *Memastikan bahwa deviasi pada kerja dan produk perangkat lunak didokumentasikan & di- tangani sesuai dgn rosedur pendokuementasian*.

- *Mencatat ketidak-sesuaian dan melaporkannya kepada manajemen senior*.

- *Mengkoordinasi kontrol dan manajemen perubahan, dan membantu mengumpulkan dan menganalisis metrik perangkat lunak*.

KAJIAN PERANGKAT LUNAK

Kajian perangkat lunak merupakan salah satu aktivitas SQA yang terpenting.

Kajian perangkat lunak adalah suatu filter bagi proses rekayasa perangkat lunak, yaitu kajian yg diterapkan pd berbagai titik selama pengembangan PL & berfungsi untuk mencari kesalahan yg kemudian akan dihilangkan.

Kajian perangkat lunak berfungsi untuk “memurnikan” produk kerja perangkat lunak yang terjadi sebagai hasil dari analisis, desain, dan pengkodean.

KAJIAN TEKNIK FORMAL
(Formal Technic Review - FTR)

FTR adalah aktivitas jaminan kualitas perangkat lunak yang dilakukan oleh perekayasa perangkat lunak.

Kajian teknik formal atau walktrough adalah pertemuan kajian yang disesuaikan dengan kebutuhan yang terbukti sangat efektif untuk menemukan kesalahan.

Keuntungan utama kajian teknis formal adalah penemuan kesalahan sejak awal sehingga tidak berlanjut ke langkah selanjutnya dalam proses perangkat lunak.

Tujuan FTR adalah

1. Menemukan kesalahan dlm fungsi, logika, / implementasinya dlm berbagai representasi PL;
2. Membuktikan bahwa perangkat lunak di bawah kajian memenuhi syarat;
3. Memastikan bahwa PL disajikan sesuai dgn standar yg sudah ditentukan sebelumnya;
4. Mencapai perangkat lunak yg dikembangkan dengan cara yang seragam;
5. Membuat proyek lebih dapat dikelola.

FTR berfungsi sebagai dasar pelatihan yang memungkinkan perekayasa yunior mengamati berbagai pendekatan yang berbeda terhadap analisis perangkat lunak, desain, dan implementasi.
FTR juga berfungsi untuk mengembangkan backup dan kontinuitas karena sejumlah orang mengenal baik bagian-bagian perangkat lunak yang tidak mereka ketahui sebelumnya.

Masing-masing FTR dilakukan sebagai suatu pertemuan dan akan berhasil hanya bila direncanakan, dikontrol dan dihadirkan dengan tepat. Dalam paragraf berikut, panduan yang mirip dengan *walktrough* disajikan sebagai kajian teknis formal representatif.

TABEL 8.1 Perbandingan Biaya Pengembangan

Kesalahan yang ditemukan	Jumlah	Unit Biaya	Total
		Kajian dilakukan	
Selama desain	22	1.5	33
Sebelum pengujian	36	6.5	234
Selama pengujian	15	15	315
Setelah peluncuran	3	67	201
			783

		Kajian tidak dilakukan	
Sebelum pengujian	22	6.5	143
Selama pengujian	82	15	1230
Setelah peluncuran	12	67	804
			2177

Pertemuan Kajian

Tanpa memperhatikan format FTR yang dipilih, setiap pertemuan kajian harus mematuhi batasan-batasan berikut ini :

- Antara 3 & 5 orang (khususnya) harus dilibatkan dalam kajian;
- Persiapan awal harus dilakukan, tetapi waktu yang dibutuhkan harus tidak lebih dari 2 jam dari kerja bagi setiap person
- Durasi pertemuan kajian harus kurang dari 2 jam

Pertemuan kajian dihadiri oleh pimpinan kajian, pengkaji, dan prosedur. Salah satu dari pengkaji berperan sebagai pencatat, yaitu seseorang yang mencatat semua masalah penting yang muncul selama pengkajian.

FTR dimulai dengan pengenalan agenda dan pendahuluan dari prosedur. Bila ada masalah kesalahan ditemukan akan dicatat.

Pada akhir kajian, semua peserta FTR yang hadir harus memutuskan apakah akan

1. menerima produk kerja tanpa modifikasi lebih lanjut,
2. menolak produk kerja sehubungan dengan kesalahan yangada (sekali dbetulkan, kajiann lain harus dilakukan), atau

3. menerima produk kerja secara sementara (kesalahan minor telah terjadi & harus dikoreksi, tetapi kajian tambahan akan diperlukan).

Keputusan kemudian dibuat.

Semua peserta FTR melengkapinya dengan *tanda tangan* yang menunjukkan partisipasi mereka dalam kajian serta persetujuan mereka terhadap pertemuan tim kajian.

Pelaporan Kajian dan Penyimpanan Rekaman

Selama FTR, seorang pengkaji (pencatat) secara aktif mencatat semua masalah yang sudah dimunculkan, yang kemudian dirangkum pada akhir pertemuan sehingga dihasilkan *daftar masalah kajian*. Sebagai tambahan, *laporan rangkuman kajian* yang sederhana telah diselesaikan di mana rangkuman kajian merupakan jawaban dari tiga pertanyaan berikut:

1. Apa yang dikaji ?
2. Siapa yang melakukan?
3. penemuan apa yang dihasilkan dan apa kesimpulannya?

Daftar masalah kajian mempunyai dua tujuan:

1. Mengidentifikasi area masalah pada produk,
2. Daftar *item kegiatan* yang menjadi petunjuk bagi prosedur saat koreksi dilakukan.

Daftar masalah biasanya dilampirkan pada laporan.

Pedoman Kajian

Pedoman untuk melakukan kajian teknis formal harus dilakukan sebelumnya, didistribusikan kepada semua

pengkaji, disetujui, dan kemudian dilaksanakan. Kajian yang tidak terkontrol sering dapat menjadi lebih buruk daripada bila tidak ada kajian sama sekali.

Berikut ini serangkaian pedoman minimum untuk kajian teknis formal:

1. *Kajian produk, bukan produser*.
2. *Menetapkan agenda dan menjaganya*.
3. *Membatasi perdebatan dan bantahan*.
4. *Menetapkan area masalah, tetapi tidak tergoda untuk menyelesaikannya setiap masalah yang dicatat*.
5. *Mengambil catatan tertulis*.
6. *Membatasi jumlah peserta dan mewajibkan persiapan awal*.
7. *Mengembangkan daftar bagi masing-masing produk kerja yang akan dikaji.*
8. *Mengalokasikan sumber-sumber daya dan jadwal waktu untuk FTR.*
9. *Melakukan pelatihan bagi semua pengkaji.*
10. *Mengkaji kajian awal Anda.*

PENDEKATAN FORMAL TERHADAP SQA

Kualitas perangat lunak merupakan tugas setiap orang & kualitas dapat dicapai melalui analisis, desain, pengkodean, dan pengujian yang baik serta aplikasi standar pengembangan perangkat lunak yang diterima.

Pada lebuh dari dua dekade, segmen komunitas rekayasa perangkat lunak yang kecil tetapi vokal telah memperlihatkan bahwa dibutuhkan suatu pendekatan yang lebih formal terhadap jaminan kualitas perangkat lunak.

Pembuktian matematis terhadap kebenarannya dapat diaplikasikan untuk menunjukkan bahwa program menyesuaikan diri secara tepat dengan spesifikasinya.

JAMINAN KUALITAS STATISTIK (SQA)

Jaminan kualitas statistik mencerminkan trend yang sedang tumbuh di seluruh industri untuk menjadi lebih kuantitatif terhadap kualitas.
Pada perangkat lunak, jaminan kualitas statistik mengimplikasikan langkah-langkah berikut ini:

1. Informasi tentang cacat perangkat lunak dikumpulkan dan dipilah-pilahkan.
2. Melakukan suatu usaha untuk menelusuri masing-masing cacat sampai ke penyebab pokoknya.
3. Dengan menggunakan prinsip Pareto (80 persen cacat dapat ditelusuri sampai 20 persen dari semua kemungkinan penyebab), mengisolasi yang 20 persen tersebut (*vital few*)
4. Sekali penyebab *vital few* telah diidentifikasi, beralih untuk membetulkan maslah yang menyebabkan cacat.

Banyak kesalahan ditemukan pada waktu perangkat lunak sedang dalam proses pengembangan. Cacat yang lain ditemukan setelah perangkat lunak diluncurkan kepada pemakai akhir. Meskipun ratusan kesalahan yang berbeda diluncurkan, semuanya dapat ditelusuri dari satu (atau lebih) penyebab berikut ini :

- Spesifikasi yang tidak lengkap atau keliru (IES)
- Kesalahan interpretasi komunikasi pelanggan (MMC)
- Deviasi intersioanl dari spesifikasi (IDS)
- Pelanggaran standar pemrograman (VPS)
- Kesalahan dalam representasi data (EDRIMI)

- Kesalahan dalam logika desain (EDL)
- Interface modul yang tidak konsisten (IMI)
- Pengujian yang tidak lengkap atau keliru (IET)
- Dokumentasi yang tidak lengkap atau tidak akurat (IID)
- Kesalahan dalam penerjemahan bahasa pemrograman desain (PLT)
- Antarmuka manusia dengan komputer yang tidak konsisten atau mengandung ambiguitas (HCI)
- Dan msih banyak lagi (MIS)

RELIABILITAS PERANGKAT LUNAK

Reliabilitas perangkat lunak, tidak seperti faktor kualitas yang lain, dapat diukur, diarahan, dan diestimasi dengan menggunakan data pengembangan historis. Reliabilitas perangkat lunak didefinisikan dalam bentuk statistik sebagai “kemungkinan operasi program komputer bebas kegagalan di dalam suatu lingkungan tertentu dan waktu tertentu”.

Kapan saja reliabilitas perangkat lunak dibicarakan, selalu muncul pertanyaan yang sangat penting : Apa yang dimaksudkan dengan bentuk “kegagalan?” dalam konteks dan banyak diskusi mengenai kualitas dan reliabilitas perangkat lunak, kegagalann adalah ketidaksesuaian dengan kebutuhan perangkat lunak.

Kegagalan hanya akan mengganggu atau bahkan merupakan bencana. Satu kegagalan dapat diperbaiki dalam beberapa detik sementara kesalahan yang lain mungkin membutuhkan waktu pembetulan berminggu-minggu atau bahkan berbulan-bulan.

Pembetulan satu kegagalan kenyataannya dapat menghasilkan kesalahan lain yang baru yang mungkin akan membawa lagi kesalahan yang lain lagi.

Pengukuran Reliabilitas dan Availabilitas

Kerja awal dalam reliabilitas perangkat lunak berusaha mengekstrapolasi matematika teori reliabitas perangkat keras. Sebagian besar model reliabilitas yang berhubungan dengan perangkat keras didasarkan pada kegagalan sehubungan dengan *keusangan (wear)*, bukan kesalahan karena cacat desain. Dalam perangkat keras, kegagalan sehubungan dengan keusangan fisik (misalnya pengaruh suhu, korosi, kejutan) lebih banyak terjadi daripada kegagalan karena isu. Akan tetapi, yang terjadi pada perangkat lunak adalah hal yang sebaliknya. Kenyataannya, semua kegagalan perangkat lunak dapat ditelusuri ke dalam desain atau masalah implementasi; keusangan tidak tercakup.

Masih ada perdebatan yang terjadi di seputar hubunan antara konsep kunci dalam reliabilitas perangkat keras dan kemampuan aplikasinya terhadap perangkat lunak.

Meskipun ada hubungan yang tidak dapat dibantah, namun sangat penting untuk memprtimbangkan beberapa konsep sederhana yang berlaku untuk kedua sistem elemen tersebut.

Bila kita andaikan suatu sistem yang berbasis komputer, pengukuran reliabilitas secara sederhana adalah berupa mean time *between failure* (MTBF), dimana :

MTBF = MTTF + MTTR
(Akronim MTTF adalah mean time to failure dan MTR berarti *mean time to repair.*)

Banyak peneliti berpendapat bahwa MTBF merupakan pengukuran yang jauh lebih berguna daripada

pengukuran cacat/KLOC. Secara sederhana dapat dikatakan bahwa seorang pemakai akhir lebih memperhatikan kegagalan, bukan jumlah cacat. Karena masing-masing cacat yangada pada sebuah program tidak memiliki tingkat kegagalan yang sama, maka penghitungan cacat total hanya memberikan sedikit indikasi tentang reliabilitas sistem.

Contohnya adalah sebuah program yang telah beroperasi selama 14 bulan. Banyak cacat mungkin tidak terdeteksi dalam jumlah waktu yang lama sampai pada akhirnya cacat itu ditemukan. MTBF dari cacat yang tidak jelas seperti itu dapat berlangsung sampai 50, bahkan 100 tahun. Cacat yang lain, yang juga belum ditemukan, dapat memiliki tingkat kegagalan 18 atau 24 bulan. Meskipun setiap kategori pertama cacat (yang memiliki MTBF panjang) dihilangkan, pengaruhnya pada reliabilitas perangkat lunak tidak dapat diabaikan.

Availabilitas perangkat lunak adalah kemungkinan sebuah program beroperasi sesuai dengan kebutuhan pada suatu titik yang diberikan pada suatu waktu dan didefinisikan sebagai :

Availabilitas = MTTF / (MTTF + MTTR) x 100 %

Pengukuran reliabilitas MTBF sama sensitifnya dengan MTTF dan MTTR. Pengukuran availabilitas jauh lebih sensitif daripada MTTR, yang merupakan pengukuran tidak langsung terhadap kemampuan pemeliharaan perangkat lunak.

Keamanan Perangkat Lunak dan Analisis Risiko

Leveson membicarakan pengaruh perangkat lunak dalam sistem kritis keamanan ketika menulis :

Sebelum perangkat lunak digunakan di dalam sistem kritis keamanan, perangkat lunak itu sering dikontrol oleh alat mekanik konvensional (tidak dapat diprogram) dan elektronik. Teknik keamanan sistem didesain untuk mengatasi kegagalan acak dalam sistem-sistem tersebut. Kesalahan perancangan oleh manusia dapat sepenuhnya dihindari atau dihilangkan sebelum perangkat lunak tersebut diluncurkan dan dioperasikan.

Ketika perangkat lunak diguanakn sebagai bagian dari sistem kontrol, kompleksitasnya dapat bertambah dengan satu urutan besaran atau lebih. Kesalahan desain yang tidak kentara yang disebabknan oleh kesalahan manusia – sesuatu yang dapat diunkapkan dan dikurangi dalam kontrol konvensional berbasis perangkat keras – menjadi lebih sulit ditemukan pada waktu perangkat lunak digunakan.

Keamanan perangkat lunak dan analisis risiko adalah aktivitas jaminan kualitas perangkat lunak yang berfokus pada identifikasi dan penilaian risiko potensial yang mungkin berpengaruh negatif terhadap perangkat lunak dan menyebabkan seluruh sistem menjadi gagal. Jika risiko dapat diidentifikasi pada awal proses rekayasa perangkat lunak, maka ciri-ciri desain perangkat lunak dapat ditetapkan sehingga akan mengeliminasi atau mengontrol risiko potensial.

Proses analisis dan modeling dilakukan sebagai bagian dari keamanan perangkat lunak. Awalnya, risiko diidentifikasi dan dipilah-pilahkan berdasarkan kekritisan dan risiko. Sebagai contoh, beberapa risiko yang berkaitan dengan kontrol peluncuran berbasis komputer untuk automobil mungkin:

- Menyebabkan percepatan yang tidak terkontrol tidak dapat dihentikan
- Tidak lepas ketika pedal rem ditekan
- Tidak nyambung ketika skalar diaktifkan
- Perlahan-lahan kehilangan atau menambah kecepatan

Setelah risiko tingkat sistem diidentifikasi, maka digunakan teknik analisis untuk menandai kehebatan dan probabilitas event. Supaya efektif, perangkat lunak harus dianalisis dalam konteks keseluruhan sistem. Sebagai contoh, kesalahan input pemakai yang tidak kentara (manusia *sebagai* komponen sistem) dapat diperbesar oleh kesalahan perangkat lunak, sehingga menghasilkan data kontrol yang memposisikan sebuah perangkat lunak, sehingga menghasilkan data kontrol yang memposisikan sebuah perangkat mekanik secara tidak tepat. Jika ada serangkaian kondisi lingkungan eksternal (dan hanya jika mereka ditemui), maka posisi perangkat mekanik yang tidak tepat dapat menyebabkan kegagalan fatal. Teknik analisis seperti analisis pohon kegagalan, logika real-time , atau model Petrinet , dapat digunakan untuk memprediksi rantai event yang dapat mengakibatkan risiko dan kemungkinan di mana setiap event akan terjadi untuk menciptakan rantai.

Analisis pohon kesalahan membangun model grafis dan kombinasi event yang konkuren dan berurutan yang dapat menyebabkan suatu event atau sistem yang penuh risiko. Dengan menggunakan pohon kesalahan yang dikembangkan dengan baik, maka dimungkinkan untuk meneliti kosekuensi urutan kegagalan yang terinterelasi yang terjadi pada komponen sistem yang berbeda. *Logika real-time* (RTL) membangun sebuah model sistem dengan menentukan event dan aksi yang sesuai. Model *event-*

action dapat dianalisis dengan menggunakan operasi logika untuk menguji tuntutan keamanan seputar komponen sistem dan *timing*-nya. Model *Petrinet* dapat digunakan untuk menentukan kesalahan yang paling berisiko.

Sekali risiko diidentifikasi dan dianalisis, maka keamanan yang berhubungan dengan kebutuhan untuk perangkat lunak dapat ditetapkan. Spesifikasi dapat berupa sederetan event yang tidak diinginkan dan sistem yang diinginkan merespon event tersebut. Peran perangkat lunak dalam mengatur event yang tidak diinginkan kemudian diindikasi.

Meskipun reliabilitas perangkat lunak berhubungan erat satu sama lain dengan lainnya, namun sangat penting untuk memahami perbedaan tipis yang ada di antara mereka. Reliabilitas perangkat lunak menggunakan analisis statistik untuk menentukan kemungkinan terjadinya kegagalan perangkat lunak. Tetapi kegagalan tidak perlu menghasilkan risiko atau kecelakaan. Kemanan perangkat lunak mengamati bagaimana kegagalan menimbulkan keadaan yang dapat menyebabkan kecelakaan. Kegagalan tidak perlu dipertimbangkan di dalam ruang hampa, tetapi dievaluasi dalam konteks keseluruhan sistem berbasis komputer.

Diskusi komprehensif tentang analisis risiko dan keamanan perangkat lunak tidak masuk dalam ruang lingkup buku ini. Pembaca yang tertarik untuk mengetahui lebih jauh tentang hal tersebut sebaiknya membaca buku yang ditulis oleh Leveson .

RENCANA SQA

SQA plan menjadi peta jalan untuk membangun jaminan kualitas perangkat lunak. Dikembangkan oleh kelompok SQA dan tim proyek, rencana itu berfungsi sebagai *template* bagi aktifitas SQA yang dibangun untuk setiap proyek perangkat lunak.

Gambar 8.5 memperlihatkan sebuah outline untuk rencana SQA yang disetujui oleh IEEE . Bagian awal menggambarkan tujuan dan ruang lingkup dokumen dan menunjukkan aktivitas proses perangkat lunak yang diungkap oleh jaminan kualitas. Semua dokumen yang dicatat oleh rencana SQA didaftar dan semua standar yang dapat diapliksikan dicatat. Bagian *Manajemen* dari rencana tersebut menggambarkan tempat SQA pada struktur organisasi; tugas-tugas dan aktivitas SQA dan penempatannya di seluruh proses perangkat lunak; dan peran organisasional serta tanggung jawab relatif terhadap kualitas produk.

Bagian *Dokumentasi* menggambarkan (dengan refernsi) masing-masing produk kerja yang dihasilkan sebagai bagian dari proses perangkat lunak; mencakup hal-hal berikut :

- Dokumen proyek (misalnya, rencana proyek)
- Model (misalnya, hirarki kelas ERD)
- Dokumen teknis (misalnya, spesifikasi, rencana pengujian)
- Dokumen pemakai (misalnya file0file help)

Sebagai tambahan, bagian ini menentukan serangkaian produk kerja minmum yang dapat diterima untuk mencapai kualitas yang tinggi.

Standar, Praktik dan Konversi mencatat semua standar/praktik yang diterapkan selama proses perangkat lunak (misalnya, standar dokumen, stadar pengkodean, dan pedoman kajian). Semua proyek, proses, dan metrik produk yang dikumpulkan sebagai bagian dari usaha rekayasa perangkat lunak juga harus dicatat.

Bagian *Kajian dan Audit* dari rencana mengidentifiaksi kajian dan audit yang akan dilakukan oleh tim rekayasa perangkat lunak, kelompok SQA, dan pelanggan. Bagian ini memberikan gambaran yang luas terhadap pendekatan bagi masing-masig kajian dan audit.

I. Tujuan Rencana
II. Referensi
III. Manajemen
    1. Organisasi
    2. Tugas
    3. Tanggung jawab
IV. Dokumentasi
    1. Tujuan
    2. Dokuen rekayasa perangkat lunak yang diperlukan
    3. Dokumen-dokumen lain
V. Standar, Praktis dan Konversi
    1. Tujuan
    2. Konvensi
VI. Tinjauan dan Audit
    1. Tujuan
    2. Tinjauan
        a. Kebutuhan perangkat lunak
        b. Desain
        c. Verifikasi dan validasi perangkat lunak
        d. Audit fungsional
        e. Audit fisik
        f. Audit *in-process*
        g. Manajemen
VII. Pengujian
VIII. Pelaporan Masalah dan Tindakan Koreksi
IX. Peranti, Teknik, dan Metodologi
X. Kontrol Kode
XI. Kontrol Media
XII. Kontrol Pemasok

XIII. Pengumpulan, Pemeliharaan, dan Penyimpanan Catatan
XIV. Pelatihan
XV. Manajemen Risiko

Gambar 8.5 *Rencana kualitas jaminan perangkat lunak standar ANSI/IEEE 730 – 1984 dan 983-1986*

Bagian *pengujian* merujuk rencana dan prosedur pengujian perangkat lunak (Bab17). Bagian ini juga menentukkan kebutuhan penyimpanan rekaman pengujian. *Pelaporan Masalah dan Tindakan Korektif* menentukan prosedur untuk pelaporan, pelacakan, dan pembetulan kesalahan serta cacat, juga mengidentifikasi tanggung jawab organisaional untuk akyivitas-aktivitas tersebut.

Bagian akhir rencana SQA adalah mengidentifikasi peranti dan metode yang mengandung aktifitas dan tugas-tugas SQA; merujuk manajemen konfigurasi perangkat lunak untuk mengontrol perubahan; menetapkan pendekatan manajemen kontrak; membangun metode perakitan, perlindungan dan pemeliharaan semua catatan; mengidentifikasi pelatihan yang dibutuhkan untuk memenuhi kebutuhan rencana, serta menetapkan metode-metode untuk mengidentifikasi, menilai, memonitor, dan mengontrol risiko.

STANDAR KUALITAS ISO 9000

Sistem jaminan kualitas dapat didefinisikan sebagai strukutr, tanggung jawab, prosedur, proses dan sumber-sumber daya organisasi untuk mengimplementasi manajemen kualitas. ISO 9000 menjelaskan elemen jaminan kualitas dalam bentuk yang umum yang dapat diaplikasikan pada berbagai bisnis tanpa memandang produk dan jasa yang ditawarkan.

Agar terdaftar dalam satu model sistem jaminan kualitas yang ada pada ISO 9000, sistem kualitas dan operasi perusahaan diperiksa oleh auditor bagian ketiga untuk memeriksa kesesuaiannya dengan standar dan operasi efektif. Bila registrasi itu berhasil, perusahaan diberi sertifikat dari badan registrasi yang diwakili oleh auditor. Audit pengawasan tegah tahuan terus dilakukan untuk memastikan kesesuaiannya dengan standar yang sudah ditetapkan.

Pendekatan ISO terhadap Sistem Jaminan Kualitas

Model jaminan kualitas ISO 9000 memperlakukan perusahaan sebagai jaringan proses yang saling terhubung (interkoneksi). Suatu sistem kualitas, supaya sesuai dengan ISO, proses-prosesnya harus menekankan pada area yang telah diidentifikasi pada standar ISO, dan harus didokumentasi dan dipraktikan sebagimana dikelaskan. Pendokuemnatsian proses membantu organisasi untuk memahami, mengontrol, dan mengembangkan jaringan proses yang mungkin dapat mendatangkan keuntunagn terbesar bagi organisasi yang merancang dan mengimplementasikan kualitas yang sesuai dengan ISO.

ISO 9000 menggambarkan elemen sebuah sistem jaminan kualitas secara umum. Elemen-elemen tersebut mencakup struktur, prosedur, proses, organisasi, dan sumber day ayang dibutuhkan untuk mehimplementasi rencana kualitas, kontrol kualitas, jaminan, kualitas, dan pengembangan kualiats. Tetapi ISO 9000 tidak menggambarkan bagaimanan organisasi seharusnya mengimpelemnatsi elemen-elemen kualitas tersebut. Sebagai konsekuensi, ada tantangan dalam mendesain dan mengimplementasi suatu sistem jaminan kualitas yang memenuhi standar dengan produk, layanan dan budaya perusahaan.

Standar ISO 9001

ISO 9001 adalah standar kualitas yang berkalu untuk rekayasa perangkat lunak. Standar tersebut berisi 20 syarat yang harus ada untuk mencapai sistem jaminan kualitas yangefektif. Karena standar ISO 9001 dapat diaplikasikan pada semua disiplin rekayasa / engineering, maka dikembangkan sekumpulan khusus pedoman ISO untuk membantu menginterpretsi standar untuk digunakan pada proses perangkat lunak.

Dua puluh syarat yang digambarkan oleh ISO 9001 menekankan topik-topik berikut :

1. Tanggung jawab manajemen
2. Sistem kualitas
3. Kajian kontrak
4. Kontrol desain
5. Kontrol data dan dokumen
6. Pembelian
7. Kontrol terhadap produk yang disuplai oleh pelanggan
8. Identifikasi dan kemampuan penelusuran produk
9. Kontrol proses
10. Pemeriksaan dan pengujian
11. Kontrol pemeriksaan, pengukuran, dan perlengkapan pengujian
12. Pemeriksaan dan status pengujian
13. Kontrol ketudaksesuaian produk
14. Tindakan preventif dan korektif
15. Penanganan, penyimpanan, pengepakan, preservasi, dan penyampaian
16. Kontrol terhadap catatan kualitas
17. Audit kualitas internal
18. Pelatihan

19. Pelayanan
20. Teknik statistik

Untuk dapat didaftar dalam ISO 9001, organisasi perangkat lunak harus membuat kebijakan dan prosedur yang memberi tekanan pada masing-masing syarat tersebut dan kemudian dapat menunjukkan bahwa prosedur dan fungsi itu telah diikuti. Untuk penjelsan leih lanjt, pembaca yang tertarik dengan informasi mengnenai ISO 9001.