KONTROL KUALITAS PADA PERANGKAT LUNAK

Rini Astuti

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

riniastuti@likmi.ac.id

Abstrak

Umumnya kontrol kualitas dilakukan untuk memastikan bahwa setiap produk sudah memenuhi persyaratan yang sudah ditetapkan. Di dalam pengembangan sistem informasi, kontrol kualitas merupakan serangkaian pemeriksaan, kajian, dan pengujian yang digunakan pada keseluruhan siklus pengembangan untuk memastikan bahwa produk sistem informasi dibuat sesuai dengan kebutuhan dan ketentuan yang ditetapkan.

Identifikasi cakupan dan karakteristik dari kontrol kualitas pada pengembangan sistem informasi sangat ditentukan oleh faktor kualitas dan jenis pengujian yang dibutuhkan oleh organisasi.

Dalam tulisan ini akan diuraikan beberapa metode kontrol kualitas terhadap produk sistem informasi yang didukung teknologi informasi yaitu perangkat lunak.

Kata-kata kunci *: faktor kualitas dan kontrol kualitas*

1. PENDAHULUAN

Kontrol kualitas sangat menentukan reaksi dan umpan balik pemakai ketika mereka mendapatkan produk yang telah dipesannya. Kontrol kualitas sistem informasi adalah kumpulan prosedur yang digunakan oleh organisasi (1) untuk memastikan bahwa produk sistem informasi akan memenuhi tujuan kualitas di nilai terbaik untuk pelanggan, dan (2) untuk terus meningkatkan kemampuan organisasi dalam menghasilkan produk sistem informasinya di masa yang akan datang.

Umumnya metode kontrol kualitas mengharuskan untuk ditetapkannya faktor kualitas yang dibutuhkan dalam lingkungan organisasi penggunanya. Faktor kualitas

adalah karakteristik dari kemampuan suatu perangkat lunak baik secara fungsional dan non fungsional.

Produk sistem informasi umumnya berupa perangkat lunak yang siap dioperasikan di lingkungan pemakainya selanjutnya disebut Perangkat Lunak.

Kontrol kualitas suatu Perangkat Lunak mengacu pada persyaratan fungsional dan kebutuhan non-fungsional seperti kemampuan dukungannya, kinerja dan kegunaan[5]. Hal ini juga mengacu pada kemampuan perangkat lunak untuk melakukan baik dalam skenario yang tak terduga maupun untuk menjaga tingkat kerusakan yang relatif rendah.

Prosedur ini biasanya menjadi prosedur standar dan persyaratan yang digariskan untuk ide Verifikasi dan Validasi serta pengujian perangkat lunak. Hal ini berbeda dari jaminan kualitas perangkat lunak yang mencakup audit terhadap sistem manajemen kualitas terhadap standar. Sedangkan kontrol kualitas perangkat lunak adalah kontrol dari produk, jaminan kualitas perangkat lunak adalah kontrol dari proses.

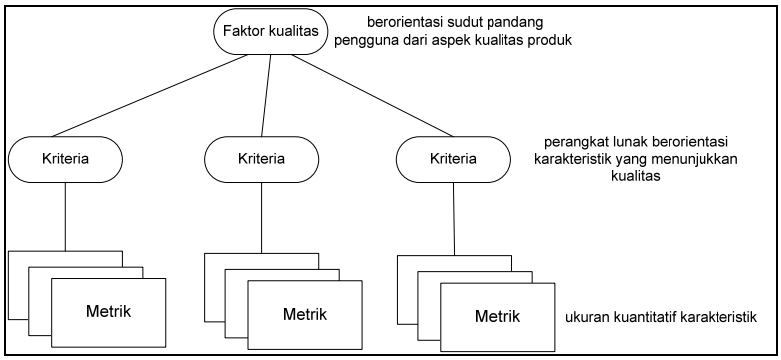
2. METODE KONTROL PERANGKAT LUNAK

Terdapat beberapa metode yang dapat digunakan untuk melakukan kontrol terhadap perangkat lunak yang sudah dibangun. Metode tersebut diantaranya adalah sebagai berikut :

1. Kerangka kualitas Perangkat Lunak *Rome laboratory*
2. Paradigma *Goal Question Metric*
3. Model Manajemen Resiko
4. Model Pengembangan Spiral

2.1 Kerangka kualitas Perangkat Lunak *Rome laboratory*

Kerangka kualiats perangkat lunak *Rome laboratory* merupakan salah satu metode yang sistematis awal untuk menentukan, memprediksi, dan mengevaluasi kualitas perangkat lunak. Metode ini menggunakan model hirarki untuk mendefinisikan dan mengukur kualitas dalam suatu produk perangkat lunak berdasakan faktor kualitas seperti dalam Gambar 1.



Gambar 1. Model kualitas Peranglat Lunak

.

Definisi faktor kualitas model *Rome Laboratory* [1] meliputi :

1. Efisiensi: relatif sejauh mana sumber daya digunakan (ruang penyimpanan waktu proses misalnya, waktu komunikasi).
2. Integritas: sejauh mana perangkat lunak akan melakukan tanpa kegagalan karena kode akses tidak sah ke kode atau data dalam suatu periode waktu tertentu.
3. Reliabilitas: sejauh mana perangkat lunak akan melakukan tanpa kegagalan apapun dalam suatu periode waktu tertentu.
4. *Survavibility:* sejauh mana perangkat lunak akan melakukan dukungan fungsi kritis tanpa kegagalan dalam periode waktu tertentu ketika sebagian dari sistem dalam beroperasi.
5. *Usability:* upaya relatif untuk menggunakan perangkat lunak (pelatihan dan operasi).
6. *Correctness*: sejauh mana perangkat lunak sesuai dengan spesifikasi dan standarnya.
7. *Maintainability*: kemudahan dalam upaya untuk menemukan dan memperbaiki kegagalan perangkat lunak dalam periode waktu tertentu.
8. *Verifiability:* upaya relatif untuk memverifikasi pengoperasian dan kinerja perangkat lunak tertentu.
9. *Expandability:* upaya relatif untuk meningkatkan kemampuan perangkat lunak tertentu atau performa dengan meningkatkan fungsi saat ini atau dengan menambahkan fungsi baru atau data.
10. *Flexibility:* kemudahan dalam upaya untuk mengubah tujuan, fungsi atau data perangkat lunak untuk memenuhi persyaratan lain.

11. *Interoperability:* upaya relatif untuk beberapa perangkat lunak dari satu sistem ke sistem lain.
12. *Portability*: upaya relatif untuk mengangkut perangkat lunak untuk digunakan di lingkungan sistem lain (konfigurasi perangkat keras dan/atau lingkungan perangkat lunak sistem)
13. *Reusability* : upaya relatif untuk mengkonversi komponen perangkat lunak untuk digunakan di aplikasi lain.

2.2 Paradigma *Goal Question Metric*

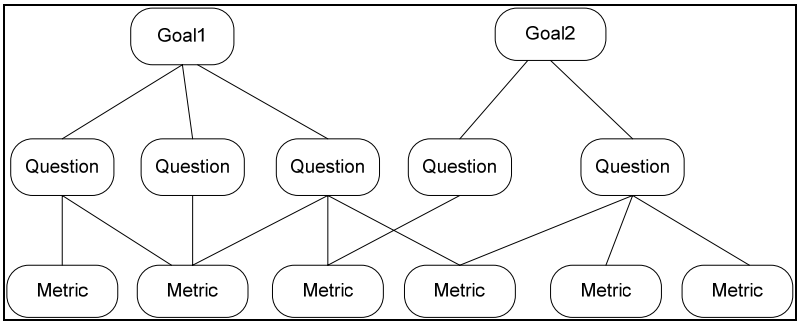
Model pendekatan ***Goal Question Metric (GQM)*** juga dapat digunakan untuk mendefinisikan seberapa baik sasaran kualitas dan monitoring yang dapat terpenuhi. Pendekatan ini didasarkan pada asumsi bahwa suatu organisasi mengukur dengan cara yang harus menentukan tujuan untuk dirinya sendiri lebih dulu dan proyeknya, kemudian harus melacak tujuan-tujuan dengan data yang dimaksudkan untuk menentukan tujuan-tujuan operasional, dan akhirnya menyediakan kerangka kerja untuk menafsirkan data sehubungan dengan tujuan lain.

Paradigma GQM terdiri dari tingkatan berikut :

1. Tingkat Konseptual (Goal/Tujuan): Tujuan adalah ditetapkan untuk objek, karena berbagai alasan, sehubungan dengan berbagai model kualitas, dari berbagai titik pandang, relatif terhadap lingkungan tertentu. Objek pengukuran adalah:
   a. Produk: berbagai model, penyerahan dan dokumen yang dihasilkan selama siklus hidup sistem; Misalnya, spesifikasi, desain, program, paket uji.
   b. Proses: Perangkat Lunak yang terkait kegiatan biasanya terkait dengan waktu; misalnya, menetapkan, merancang, pengujian, wawancara.
   c. Sumber daya: Produk yang digunakan oleh proses untuk menghasilkan output mereka; Misalnya personal, hardware, software, ruang kantor.
2. Tingkat Operasional (Question/Pertanyaan): Satu kumpulan pertanyaan yang digunakan untuk menggambarkan cara penilaian/pencapaian tujuan tertentu yang akan dilakukan berdasarkan beberapa model karakteristik. Pertanyaan ini mencoba mengkarakteristikkan objek pengukuran (produk, proses, sumber daya) sehubungan dengan masalah kualitas yang dipilih dan untuk menentukan kualitas dari sudut pandang yang dipilih.

3. Tingkat Kuantitatif (Metric): Satu set data terkait dengan setiap pertanyaan untuk menjawabnya dengan cara kuantitatif. Data dapat berupa:
   a. Tujuan: Jika mereka bergantung hanya pada obyek yang sedang diukur dan tidak pada sudut pandang dari mana mereka diambil; Misalnya, beberapa versi dari dokumen, jam staf dihabiskan untuk tugas, ukuran program.
   b. Subjektif: Jika mereka bergantung pada kedua obyek yang sedang diukur dan sudut pandang dari mana mereka diambil, misalnya pembacaan teks, tingkat kepuasan pengguna.

Sebuah model GQM adalah struktur hirarkis (lihat Gambar 2) yang dimulai dengan tujuan (menetapkan tujuan pengukuran, objek yang akan diukur, masalah untuk diukur, dan sudut pandang pengukuran). Tujuannya adalah disempurnakan dalam beberapa pertanyaan, seperti satu contoh, yang biasanya memecah masalah menjadi komponen-komponen utamanya. Masing-masing pertanyaan ini kemudian disempurnakan dalam metrik, seperti misalnya beberapa di antaranya tujuan, beberapa dari mereka subjektif. Metrik yang sama dapat digunakan untuk menjawab berbagai pertanyaan di bawah tujuan yang sama. Beberapa model GQM juga dapat memiliki pertanyaan dan metrik yang sama, yang penting ketika mengukur, yang berbeda sudut pandangnya diperhitungkan dengan benar (yaitu, metrik yang mungkin memiliki nilai yang berbeda bila diambil dari sudut pandang yang berbeda).



Gambar 2. Struktur Hirarki Model GQM

2.3 Model Manajemen Resiko

Manajemen risiko adalah suatu pendekatan sistematis untuk mengidentifikasi dan mengendalikan faktor-faktor dalam pengembangan perangkat lunak yang paling mungkin membahayakan keberhasilan pencapaian tujuan termasuk tujuan kualitas perangkat lunak.

Pendekatan manajemen resiko umumnya memiliki tahapan sebagai berikut:

1. Mengidentifikasi risiko yang terkait dengan status saat ini persyaratan produk, proses, dan sumber daya berdasarkan pengalaman umum dengan masalah dalam pengembangan perangkat lunak dan kesulitan spesifik yang muncul dalam proyek.
2. Mengevaluasi kemungkinan terjadinya resiko dan dampak terhadap biaya proyek.
3. Memprioritaskan jenis resiko berdasarkan peluang yang terjadinya tertinggi dan dapat berakibat terjadinya cacat serius pada produk.
4. Memilih teknik kontrol resiko berdasarkan prioritas resiko dan batasan proyek dan membuat rencana untuk menggunakannya.
5. Melaksanakan rencana dan memonitor kemajuan dalam mengurangi dan menyelesaikan risiko dengan menggunakan teknik seperti kajian *milestone.*
6. Setelah penilaian masing-masing, mengambil tindakan korektif jika diperlukan dan ulangi proses untuk mengevaluasi kembali dan membuat prioritas ulang terhadap resiko.

2.4 Model Pengembangan Spiral

Model spiral merupakan salah satu paradigma pengembangan perangkat lunak yang mempertimbangkan resiko. Model spiral merupakan pendekatan yang realistik untuk perangkat lunak berskala besar. Pengguna dan pembangun bisa memahami dengan baik perangkat lunak yang dibangun karena setiap kemajuan yang dicapai selama proses dapat diamati dengan baik.

Model pengembangan perangkat lunak Spiral adalah contoh dari model yang didorong resiko pembangunan perangkat lunak. Model spiral sebagian besar didasarkan pada serangkaian perkembangan prototipe, yang masing-masing diarahkan untuk memperoleh informasi yang akan mengurangi resiko perangkat lunak yang signifikan dan mempengaruhi spesifikasi sistem dan desain. Model spiral menyediakan untuk adaptasi produk dan proses pembangunan.

3. KESIMPULAN

Seperti telah dijelaskan bahwa kontrol kualitas perangkat lunak adalah, pertama, proses perencanaan, membuat, mengukur, menilai, dan mengambil tindakan yang tepat untuk mencapai kualitas yang ditetapkan oleh pelanggan, dan, kedua, kumpulan prosedur, misalnya, analisis sederhana, untuk meningkatkan proses terus-

menerus dengan mengembangkan perangkat lunak. Sehingga pengukuran merupakan dasar untuk pengendalian dan peningkatan kualitas.

Tidak mudahnya melakukan kontrol kualitas perangkat lunak mungkin disebabkan oleh hal berikut ini :

1. Adanya faktor yang sulit dipenuhi seperti kebutuhan sistem, biaya, jadwal, yang berada di luar program.
2. Manajer pengembangan lebih fokus pada biaya dan jadwal dari pada kualitas selama pengembangan perangkat lunak.
3. Memberikan hasil yang sesuai anggaran dan tepat waktu setidaknya sama penting bagi pelanggan sebagai kualitas perangkat lunak karena ada biaya tinggi terkait sehubungan dengan tidak memiliki sistem yang tersedia pada saat dibutuhkan.

4. DAFTAR PUSTAKA

1. Clapp, Judith A, *Software Quality Control, Error Analysis, and Testing*, 1995 William Andrew In.

2. Daniel Galin, Software Quality ASsurance, From theory to Implementation, Pearson, Adisson Wesley, 2004.

3. G.Gordon S., James I.Mc Manus, “Handbook of software quality assurance”, 3rd ed, Prentice Hall, NEW JERSEY.

4. Perry, William, Effective Method for Software Testing, 1995, John Wiley & Sons,inc.

5. Pressman, Roger S, PhD Software Engineering : *A Practtioner’s Approach, 2001*.

6. Rombach , H. Dieter, and Gianluigi Caldiera, Victor R. Basili, *THE GOAL QUESTION METRIC APPROACH*, 2001.

7. http://www.sqa.net/softwarequalitycontrol.html