Laporan Tugas Praktikum

4143106 – Software Quality Metrics

Disusun Oleh :

Nama : David Kristian silalahi

NIM : 11422010

Kelas : 43 TRPL 1

Prodi : Sarjana Teknologi Rekayasa Perangkat Lunak

FAKULTAS VOKASI
INSTITUT TEKNOLOGI DEL

Tugas

1. Pada sesi ini anda akan mengerjakan pratikum secara individu.

2. Tugas anda yaitu membuat ringkasan dari materi minggu ini yaitu Software Quality Metrics.

3. Tuliskan ringkasan yang anda buat dalam sebuah laporan.

4. Kumpulkan laporan anda pada ecourse dengan penamaan berikut:
LaporanMetriks_NimLengkap.pdf

Rangkuman Software Quality Metrics

Software Quality Metrics adalah ukuran kuantitatif yang menunjukkan seberapa kuat suatu item memiliki atribut kualitas tertentu. Hal ini juga dapat berupa fungsi yang mengambil data perangkat lunak sebagai input dan menghasilkan nilai numerik tunggal yang bisa diinterpretasikan sebagai tingkat keberadaan atribut kualitas tertentu dalam perangkat lunak tersebut.

Tentu terdapat tujuan dari pengukuran Software Quality Metrics, tujuan – tujuannya adalah :

1. Memfasilitasi pengendalian manajemen dan perencanaan intervensi manajerial yang tepat.
   Hal ini menghitung :
   a. Deviasi kinerja fungsional (kualitas) actual dari kinerja yang direncanakan.
   b. Deviasi kinerja jadwal dan anggaran aktual dari kinerja yang direncanakan.
2. Mengidentifikasi situasi yang memerlukan atau memungkinkan perbaikan proses pengembangan atau pemeliharaan.
   Hal ini melakukan akumulasi informasi metrik mengenai kinerja dari tim, unit, dan sebagainya.
3. Metrik juga digunakan untuk perbandingan data kinerja dengan indicator, seperti :
   a. Standar kualitas perangkat lunak yang ditetapkan.
   b. Target kualitas yang ditetapkan untuk organisasi atau individu.
   c. Prestasi kualitas tahun sebelumnya.
   d. Prestasi kualitas proyek sebelumnya.
   e. Tingkat kualitas rata-rata yang dicapai oleh tim lain yang menerapkan alat pengembangan yang sama dalam lingkungan pengembangan yang serupa.
   f. Prestasi kualitas rata-rata organisasi.
   g. Praktik industri untuk memenuhi persyaratan kualitas

Software Quality Metrics Requirements

General Requirements	Explanation
Relevant	Terkait dengan atribut yang sangat penting

Valid	Mengukur atribut yang diperlukan
Reliable	Menghasilkan hasil yang serupa ketika diterapkan dalam kondisi yang serupa
Comprehensive	Berlaku untuk berbagai macam implementasi dan situasi
Mutually exclusive	Tidak mengukur atribut yang diukur dengan metrik lain
Easy and simple	Implementasi pengumpulan data metrik sederhana dan dilakukan dengan sumber daya minimal
Does not require independent data collection	Pengumpulan data metrik terintegrasi dengan sistem pengumpulan data proyek lainnya: kehadiran karyawan, upah, akuntansi biaya, dll. Selain aspek efisiensinya, persyaratan ini berkontribusi pada koordinasi semua sistem informasi yang melayani organisasi
Immune to biased interventions by interested parties	Untuk menghindari hasil yang diharapkan dari analisis metrik, diharapkan bahwa orang yang tertarik akan mencoba mengubah data dan, dengan melakukan itu, meningkatkan catatan mereka. Tindakan seperti itu jelas membiaskan metrik yang relevan. Kekebalan (total atau setidaknya sebagian) dicapai terutama dengan pilihan metrik dan prosedur yang memadai

Classification of Software Quality Metric

1. **First Classification**
   - Process Metrics
   - Product Metrics
2. **Second Classification**
   - Quality
   - Timetable
   - Effectiveness

Process Metrics

Software Process Quality Metrics

- **Error density metrics**
  Software volume measures
  Beberapa metrik kepadatan menggunakan jumlah baris kode sementara yang lain menerapkan titik fungsi.
- **Error Counted measures**

- Beberapa berhubungan dengan jumlah kesalahan dan lain-lain dengan jumlah kesalahan tertimbang. Langkah-langkah berbobot yang memastikan tingkat keparahan kesalahan dianggap memberikan evaluasi yang lebih akurat dari situasi kesalahan.
- Metode umum yang diterapkan untuk mencapai ukuran ini adalah klasifikasi kesalahan yang terdeteksi ke dalam kelas keparahan, diikuti dengan pembobotan setiap kelas.

NCE dan WCE

Terdapat NCE dan WCE yang merupakan metrik yang digunakan dalam evaluasi kualitas perangkat lunak, khususnya dalam konteks pengujian perangkat lunak.

NCE (Normalized Cumulative Error) : Ini adalah metrik yang mengukur total kesalahan dalam pengujian perangkat lunak, yang dinormalisasi terhadap beberapa faktor seperti ukuran atau kompleksitas perangkat lunak. NCE dapat memberikan gambaran tentang efektivitas pengujian dalam menemukan kesalahan dalam perangkat lunak.

WCE (Weighted Cumulative Error) : Ini adalah bentuk pengembangan dari NCE yang memberikan bobot yang berbeda pada kesalahan tergantung pada tingkat kepentingannya. Dalam WCE, kesalahan yang lebih kritis atau penting diberi bobot yang lebih tinggi, sehingga menekankan pentingnya kesalahan tertentu dalam evaluasi kualitas perangkat lunak.

Error Density Metric

Error Density Metric adalah metrik yang digunakan untuk mengukur jumlah kesalahan per unit ukuran atau kompleksitas dalam perangkat lunak. Dengan kata lain, metrik ini mengukur seberapa sering kesalahan muncul dalam perangkat lunak relatif terhadap ukurannya atau kompleksitasnya.

Implementation of software quality metrics

Hal ini tentu memerlukan :

- Definisi metrik kualitas perangkat lunak, relevan dan memadai untuk tim, departemen, dll.
- Aplikasi reguler berdasarkan unit, dll.
- Analisis statistik dari data metrik yang dikumpulkan.
- Tindakan selanjutnya adalah :
  - Perubahan dalam organisasi dan metode unit pengembangan dan pemeliharaan perangkat lunak dan/atau badan lain yang mengumpulkan data metric
  - Perubahan dalam metrik dan pengumpulan data metric
  - Penerapan data dan analisis data untuk merencanakan tindakan korektif untuk semua unit terkait.

Definition of new software quality metrics

Memiliki empat tahap proses, yaitu :

1. Definisi atribut yang akan diukur : kualitas perangkat lunak, produktivitas tim pengembangan, dll.
2. Definisi metrik yang mengukur atribut yang diperlukan dan konfirmasi kecukupannya dalam memenuhi persyaratan yang tercantum.
3. Penentuan nilai target komparatif berdasarkan standar, kinerja tahun sebelumnya, dll. Nilai-nilai ini berfungsi sebagai indikator apakah unit yang diukur (tim atau individu atau bagian dari perangkat lunak) sesuai dengan karakteristik yang diminta dari atribut yang diberikan.
4. Penentuan metrik proses aplikasi.
    a. Metode pelaporan, termasuk proses pelaporan dan frekuensi pelaporan
    b. Metode pengumpulan data metrik

Application of the metrics – managerial aspects

- Menetapkan tanggung jawab untuk pelaporan dan pengumpulan data metrik.
- Instruksi tim mengenai metrik baru.
- Tindak lanjutnya meliputi :
  - Dukungan untuk memecahkan masalah aplikasi dan penyediaan informasi tambahan bila diperlukan.
  - Kontrol pelaporan metrik untuk kelengkapan dan akurasi.
  - Pembaruan dan perubahan definisi metrik bersama dengan metode pelaporan dan pengumpulan data sesuai dengan kinerja sebelumnya.

Statistical analysis of metrics data

- Statistik deskriptif, seperti mean, median dan modus serta penggunaan presentasi grafis seperti histogram, grafik distribusi kumulatif, diagram lingkaran dan diagram kontrol (menunjukkan juga nilai indikator) untuk mengilustrasikan informasi yang terkait, memungkinkan kita untuk cepat mengidentifikasi tren dalam nilai metrik.
- Statistik analitik, Untuk menentukan apakah perubahan yang diamati dalam metrik itu bermakna, apa pun arahnya, tren yang diamati harus dinilai signifikansinya. Inilah peran statistik analitik (misalnya, uji regresi, analisis varians, atau uji yang lebih mendasar seperti T-test dan Chi-square test).

Limitations of software metrics

- Kendala anggaran dalam mengalokasikan sumber daya yang diperlukan (tenaga, dana, dll.) untuk pengembangan sistem metrik kualitas dan penerapannya secara reguler.
- Faktor manusia, terutama penentangan karyawan terhadap evaluasi kegiatan mereka.
- Ketidakpastian mengenai validitas data, berakar pada pelaporan yang parsial dan bias.

Factors affecting development process parameters

- Programming style: sangat memengaruhi volume perangkat lunak, di mana pengkodean "wasteful" dapat menggandakan volume kode yang dihasilkan (KLOC).
- Komentar dokumentasi yang disertakan dalam kode: memengaruhi volume kode. Volume komentar biasanya ditentukan oleh gaya pemrograman (KLOC).

- Software complexity: modul kompleks membutuhkan lebih banyak waktu pengembangan (per baris kode) dibandingkan dengan modul sederhana. Modul kompleks juga mengalami lebih banyak cacat daripada modul sederhana dengan ukuran yang sama (KLOC, NCE).
- Persentase kode yang digunakan kembali: semakin tinggi persentase kode yang digunakan kembali yang dimasukkan ke dalam perangkat lunak yang dikembangkan, semakin besar volume kode yang dapat diproduksi per hari serta semakin rendah jumlah cacat yang terdeteksi dalam tinjauan, pengujian, dan penggunaan reguler (NDE, NCE).
- Profesionalisme dan ketelitian tim peninjau desain dan pengujian perangkat lunak: memengaruhi jumlah cacat yang terdeteksi (NCE).
- Gaya pelaporan hasil tinjauan dan pengujian: beberapa tim menghasilkan laporan singkat yang menyajikan temuan dalam sejumlah kecil item (NCE kecil), sementara yang lain menghasilkan laporan komprehensif, menunjukkan temuan yang sama untuk sejumlah besar item (NDE besar dan NCE).

Factors affecting the magnitude of the product (maintenance) parameters

- Kualitas perangkat lunak yang diinstal dan dokumentasinya (ditentukan oleh kualitas tim pengembangan serta tim peninjau dan pengujian): semakin rendah kualitas awal perangkat lunak, semakin besar kegagalan perangkat lunak yang diantisipasi diidentifikasi dan upaya pemeliharaan selanjutnya (NYF, NHYC).
- Gaya pemrograman dan volume komentar dokumentasi yang disertakan dalam kode: seperti pada tahap pengembangan, keduanya sangat memengaruhi volume perangkat lunak yang akan dipelihara, di mana pengkodean dan dokumentasi yang boros dapat menggandakan volume kode yang akan dipertahankan (KLMC).
- Kompleksitas perangkat lunak: modul kompleks memerlukan investasi lebih banyak sumber daya pemeliharaan per baris kode daripada modul sederhana, dan mengalami lebih banyak cacat yang tidak terdeteksi selama tahap pengembangan (NYF).
- Persentase kode yang digunakan kembali: semakin tinggi persentase kode yang digunakan kembali, semakin rendah jumlah cacat yang terdeteksi dalam penggunaan reguler serta semakin sedikit pemeliharaan korektif dan upaya HD (NYF) yang diperlukan.
- Jumlah penginstalan, ukuran populasi pengguna, dan tingkat aplikasi yang digunakan: memengaruhi jumlah panggilan HD serta jumlah kerusakan yang terdeteksi oleh pengguna selama penggunaan reguler (NHYC, NYF).