FAULTS, ERROR DAN FAILURES DALAM PERANCANGAN PERANGKAT LUNAK

Mata Kuliah: Software Engineering

DOSEN: Yudhi Fajar Saputra, S.Kom., M.Sc

Pertemuan ke-

Topik Bahasan ke-

SEMESTER : 3/ TA. 2024-2025

KODE MK/SKS: MKP001/3 SKS

PRODI INFORMATIKA/ILMU KOMPUTER

UNIVERSITAS WIDYA GAMA MAHAKAM SAMARINDA

Nama Mata Kuliah	: Software Engineeing/Rekayasa Perangkat Lunak
Kode Mata Kuliah/SKS	: **MKP _____/3 SKS**
Dosen	: **Yudhi Fajar Saputra,**
Semester	: **3/ 2024**
Hari Pertemuan / Jam	: -
Tempat Pertemuan	: **Ruang Kelas A.06**

Dalam pengembangan perangkat lunak, pemahaman yang jelas mengenai **faults**, **errors**, dan **failures** sangat penting untuk memastikan kualitas dan keandalan sistem yang dikembangkan. **Faults** adalah cacat atau kesalahan dalam kode atau desain yang dapat memicu masalah di waktu yang akan datang, sementara **errors** adalah manifestasi atau bentuk dari faults yang terjadi selama eksekusi program. Ketika errors tidak terdeteksi atau tidak ditangani dengan benar, maka akan berkembang menjadi **failures**, yaitu kegagalan perangkat lunak dalam memenuhi harapan atau persyaratan pengguna. Memahami perbedaan dan hubungan antara ketiga konsep ini membantu pengembang dalam mengidentifikasi, mendiagnosis, dan mengatasi masalah yang timbul dalam siklus hidup perangkat lunak.

1. FAULTS

Faults dalam pengembangan perangkat lunak adalah cacat atau kesalahan dalam kode atau desain sistem yang dapat menyebabkan perangkat lunak tidak bekerja seperti yang diharapkan. Istilah **fault** sering disebut juga sebagai **bug** atau **defek**, dan merupakan kesalahan yang ada dalam produk perangkat lunak sebelum dijalankan atau digunakan oleh pengguna. Faults menjadi sumber utama yang memicu **errors** saat perangkat lunak dieksekusi, yang pada gilirannya dapat menyebabkan **failures** dalam sistem.

1) Penyebab Utama Faults

   beberapa penyebab utama Faults adalah

   a) **Kesalahan Koding**: Pengembang mungkin salah menulis kode, baik karena kelalaian atau kesalahan logika, sehingga menyebabkan cacat dalam kode.
   b) **Desain yang Tidak Sesuai**: Kesalahan dalam tahap desain atau perancangan perangkat lunak dapat mengakibatkan fault. Misalnya, desain yang tidak mempertimbangkan skenario tertentu.
   c) **Pemahaman Persyaratan yang Salah**: Jika pengembang salah menginterpretasikan spesifikasi atau persyaratan sistem, hal ini bisa menyebabkan adanya fitur yang tidak sesuai dengan kebutuhan pengguna

2) Contoh Faults

   Misalnya, dalam aplikasi spada, jika ada kode dalam penilaian Ujian yang tidak

memperhitungkan kalkulasi nilai secara benar, maka hasil akhirnya adalah total nilai yang salah. Ini merupakan fault, karena cacat pada kode dapat memicu error saat dijalankan.

3) Proses Deteksi dan Penanganan Faults

Dalam pengembangan perangkat lunak, fault biasanya ditemukan dan diperbaiki melalui proses **debugging** dan **pengujian**. Pengujian perangkat lunak dilakukan pada berbagai tingkatan, seperti unit testing, integration testing, dan system testing, untuk menemukan dan memperbaiki fault sebelum perangkat lunak dirilis kepada pengguna

2. ERROR

Errors dalam pengembangan perangkat lunak adalah kesalahan atau masalah yang muncul selama eksekusi program, yang menyebabkan perangkat lunak tidak berfungsi sesuai dengan yang diharapkan. Errors merupakan dampak langsung dari **faults** yang ada dalam kode atau desain, dan terjadi ketika perangkat lunak dijalankan dan mengalami kesalahan operasional atau logika. Dalam siklus pengembangan perangkat lunak, error dapat muncul pada berbagai tahap pengujian, seperti unit testing, integration testing, atau saat penggunaan oleh pengguna akhir.

1) Penyebab Utama Errors:

beberapa penyebab utama errors adalah

a) **Eksekusi Faults**: Errors terjadi saat perangkat lunak menjalankan bagian kode yang mengandung faults. Misalnya, kesalahan logika dalam algoritma yang menyebabkan perhitungan yang tidak tepat.
b) **Input Data yang Tidak Valid**: Jika perangkat lunak tidak mampu menangani input yang tidak valid atau tidak sesuai, hal ini dapat menyebabkan error selama eksekusi.
c) **Konfigurasi Sistem yang Salah**: Konfigurasi perangkat lunak atau sistem pendukung yang tidak tepat juga dapat memicu error selama perangkat lunak berjalan.

2) Contoh Errors

Contoh error umum adalah perhitungan nilai yang salah pada kalkulasi nilai ujian yang menghasilkan total nilai salah.

3) Proses Deteksi dan Penanganan Errors

Errors dalam perangkat lunak biasanya ditangani melalui proses debugging, di mana pengembang melacak sumber errors, menganalisis dampaknya, dan memperbaiki kesalahan yang menyebabkan error. Pengujian secara menyeluruh, termasuk **unit testing**, **integration testing**, dan **user acceptance testing** (UAT),

juga dilakukan untuk mendeteksi errors dan memastikan bahwa aplikasi bekerja sesuai dengan spesifikasi.

3. FAILURE

Failure dalam pengembangan perangkat lunak adalah kondisi ketika perangkat lunak gagal memenuhi fungsionalitas yang diharapkan, sehingga tidak dapat menjalankan tugasnya sesuai spesifikasi atau kebutuhan pengguna. Failure adalah hasil akhir dari rangkaian kesalahan, dimulai dari **faults** dalam kode atau desain yang menyebabkan **errors** selama eksekusi. Ketika error tidak tertangani atau tidak terdeteksi, ini bisa mengarah pada failure, di mana perangkat lunak tidak mampu memberikan hasil yang diinginkan atau berhenti bekerja.

1) Penyebab Utama Failure:

   beberapa penyebab utama failure adalah

   a) **Kesalahan Desain atau Koding**: Kesalahan yang ada pada desain atau kode perangkat lunak yang tidak diperbaiki dapat menyebabkan failure ketika sistem tidak dapat menyesuaikan diri dengan kondisi tertentu.
   b) **External Environment**: Kondisi eksternal seperti ketidaksesuaian konfigurasi atau interaksi dengan perangkat keras dan perangkat lunak lain bisa menyebabkan failure.
   c) **Skenario Tak Terduga**: Ketika perangkat lunak menemukan skenario atau input yang tidak diperkirakan oleh pengembang, hal ini bisa menyebabkan perangkat lunak tidak dapat merespons secara benar, mengakibatkan kegagalan.

2) Contoh Failure

   Contoh failure dalam perangkat lunak adalah aplikasi Spada yang gagal memproses ujian yang menghitung total nilai dengan salah, mengakibatkan pengguna tidak menerima nilai yang tepat.

3) Proses Deteksi dan Penanganan Failure

   Perbaikan failures biasanya menjadi prioritas karena dampaknya langsung terhadap pengalaman pengguna. Tim pengembang sering menerapkan **post-mortem analysis** untuk menganalisis failures besar dan membuat perbaikan serta pencegahan untuk pengembangan ke depan.

4. KESIMPULAN

Perbandingan Faults, Errors, dan Failures

Aspek	Fault	Error	Failure
Definisi	Cacat dalam	Manifestasi dari fault	Kegagalan perangkat lunak

	perangkat lunak (bug/defek)	yang terjadi saat eksekusi	untuk memenuhi tujuan yang diinginkan
Penyebab	Kesalahan dalam kode atau desain	Hasil dari fault yang ada dalam kode	Terjadi akibat error yang tidak ditangani atau diperbaiki
Waktu	Terjadi selama pengembangan	Terjadi saat perangkat lunak dieksekusi	Terjadi saat perangkat lunak digunakan oleh pengguna
Pelaku	Developer/QA	Software tester	End User

Contoh Siklus Fault, Error, dan Failure:

1. **Fault**: Seorang pengembang menulis kode untuk menghitung nilai ujian, namun ada kesalahan logika dalam perhitungan total nilai.
2. **Error**: Ketika perangkat lunak dijalankan, kode yang mengandung fault menghasilkan hasil yang salah (misalnya, nilai yang dihitung lebih besar atau lebih kecil dari yang diharapkan).
3. **Failure**: Pengguna tidak dapat nilai, atau aplikasi berhenti sebelum selesai, menyebabkan kegagalan dalam ujian.

Perbedaan dalam Pengujian Perangkat Lunak

- **Pengujian untuk Fault**: Bertujuan untuk menemukan cacat dalam perangkat lunak, baik dalam kode atau desain, sebelum perangkat lunak diimplementasikan atau diuji lebih lanjut.
- **Pengujian untuk Error**: Bertujuan untuk menangkap error selama eksekusi, biasanya dilakukan melalui unit testing, integration testing, atau dynamic testing.
- **Pengujian untuk Failure**: Bertujuan untuk memastikan bahwa perangkat lunak dapat berfungsi dengan baik di dunia nyata dan tidak mengalami kegagalan yang mengganggu pengguna, biasanya melalui user acceptance testing (UAT) atau system testing

5. DAFTAR REFERENSI

1. Myers, G. J., Sandler, C., & Badgett, T. (2011). *The Art of Software Testing*. Wiley.
2. ISTQB Foundation Level Syllabus, 2018
3. Kaner, Falk, & Nguyen, 1999, *Testing Computer Software*
4. *Nielsen, J. (1994). Usability Engineering. Academic Press*
5. *ISO 9241-11:2018 - Ergonomics of human-system interaction - Part 11: Usability: Definitions and concepts.*
6. *Rubin, J. (2012). Handbook of Usability Testing: How to Plan, Design, and Conduct*

Effective Tests. Wiley

7. *Kohavi, R., Longbotham, R., & Tang, D. (2009). Testing for Usability in Online Systems. ACM Conference on Human Factors in Computing Systems.*

6. Daftar Bacaan

1. Sama seperti pada daftar referensi

7. JADWAL PERKULIAHAN DAN TOPIK BAHASAN

Pertemuan Ke-	TOPIK BAHASAN	BACAAN
1	a. Kontrak Perkuliahan, Perkenalan dan Penjelasan b. Pengenalan Rekayasa Perangkat Lunak	Kontrak Perkuliahan
2	a. Karakteristik perangkat lunak b. Komponen perangkat lunak c. Model perangkat lunak d. Fungsi dan peran dari software engineer	1-6
3	a. Definisi SDLC b. Jenis-jenis SDLC	Idem
4	a. Observasi dan estimasi dalam perencanaan proyek b. Tujuan perencanaan proyek c. Manajemen proyek perangkat lunak yang efektif	Idem
5	a. Proses analisis kebutuhan b. Metode analisis kebutuhan c. Spesifikasi dan validasi kebutuhan	Idem
6	a. Perangkat bantu proses analisis kebutuhan b. Konsep dasar, Konteks, Proses, dan Prinsip Perancangan Perangkat Lunak; c. Isu mendasar dalam perancangan perangkat lunak	Idem
7	a. Alat bantu perancangan (DFD dan UML) b. Macam-macam diagram yang terdapat pada UML (Class Diagram, Use Case Diagram, Activity Diagram, Sequence Diagram)	Idem
8	**UTS**	
9	a. Konsep dalam User Interface b. Prinsip Desain Antarmuka (user experience, user guidance, user diversity)	Idem
10	a. Perencanaan dalam pengujian b. Proses testing: (black box testing, white box testing)	Idem

	c. Integration testing dan user testing d. Faults, Error dan Failures	
11	Review Teknik Pengujian Perangkat Lunak dari proses testing	Idem
12	a. Pengujian unit b. Pengujian integrasi c. Pengujian sistem d. Debugging dan quality assurance	Idem
13	a. Quality assurance pada perangkat lunak b. Keamanan data akses	Idem
14	a. Definisi pemeliharaan perangkat lunak. b. Konsep Pemeliharaan Perangkat lunak	Idem
15	Teknik pemeliharaan perangkat lunak (Pemeliharaan korektif, pemeliharaan adaptif, pemeliharaan perfektif, pemeliharaan preventif)	Idem
16	**UAS**	