DEFINISI DAN KONSEP PEMELIHARAAN 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**

Pemeliharaan perangkat lunak adalah salah satu aspek dalam siklus hidup pengembangan perangkat lunak (SDLC) yang berfokus pada pemeliharaan dan pembaruan perangkat lunak setelah pengembangannya selesai dan perangkat lunak tersebut telah diterapkan dalam user environment. Tujuan utama dari pemeliharaan perangkat lunak adalah untuk memastikan bahwa sistem tetap berjalan dengan efisien, aman, dan relevan dari waktu ke waktu, dengan selalu melakukan perubahan sesuai kebutuhan, perbaikan bug, serta evolusi teknologi. Dengan demikian, pemeliharaan perangkat lunak bukan hanya memperbaiki kesalahan, tetapi juga tentang memastikan perangkat lunak tersebut terus berkembang dan memenuhi harapan penggunanya[1].

1. DEFINISI PEMELIHARAAN PERANGKAT LUNAK

Pemeliharaan perangkat lunak merujuk pada aktivitas yang dilakukan setelah perangkat lunak diimplementasikan dan diterima oleh pengguna untuk memastikan bahwa perangkat lunak tersebut tetap berfungsi sesuai harapan, relevan dengan perubahan kebutuhan, serta aman terhadap ancaman yang muncul. Tujuan utama dari pemeliharaan perangkat lunak adalah untuk memperbaiki kesalahan (bugs), mengadaptasi perangkat lunak terhadap perubahan kondisi, dan meningkatkan fungsionalitas atau kinerja perangkat lunak agar tetap memenuhi kebutuhan pengguna seiring berjalannya waktu. Pemeliharaan perangkat lunak mencakup berbagai tindakan seperti perbaikan, pembaruan, penyesuaian, dan penghapusan bagian dari perangkat lunak yang sudah tidak sesuai lagi dengan kebutuhan atau kondisi sistem

Pemeliharaan perangkat lunak dibagi menjadi empat jenis utama:

1) **Pemeliharaan Korektif**: Memperbaiki bug atau masalah yang ditemukan setelah perangkat lunak diterapkan.
2) **Pemeliharaan Adaptif**: Menyesuaikan perangkat lunak dengan perubahan di lingkungan operasional, seperti pembaruan sistem operasi atau perangkat keras
3) **Pemeliharaan Perfective**: Menyempurnakan dan meningkatkan fungsionalitas atau kinerja perangkat lunak untuk memenuhi permintaan pengguna atau untuk meningkatkan efisiensi..
4) **Pemeliharaan Preventif**: Melakukan perbaikan dan pembaruan untuk

mencegah masalah yang dapat terjadi di masa depan

2. KONSEP PEMELIHARAAN PERANGKAT LUNAK

Pemeliharaan perangkat lunak adalah proses yang dilakukan untuk mempertahankan dan meningkatkan kualitas perangkat lunak setelah perangkat lunak tersebut selesai dikembangkan dan diterapkan. Konsep pemeliharaan perangkat lunak mencakup berbagai aktivitas yang memastikan perangkat lunak terus berfungsi sesuai dengan tujuan penggunaannya, serta menyesuaikan dengan perubahan yang terjadi, baik di lingkungan perangkat keras maupun perangkat lunak itu sendiri. Pemeliharaan ini juga mencakup pengelolaan dan penyelesaian masalah yang muncul setelah perangkat lunak diimplementasikan, serta perbaikan yang diperlukan agar perangkat lunak tetap efektif dan efisien.

Berikut ini adalah beberapa model yang digunakan dalam pemeliharaan perangkat lunak:

1) **Model Pemeliharaan Reaktif**

   Model ini berfokus pada perbaikan perangkat lunak berdasarkan masalah atau bug yang dilaporkan oleh pengguna setelah perangkat lunak digunakan. Pemeliharaan ini bersifat reaktif karena bertindak setelah masalah terjadi. Biasanya, model ini digunakan ketika perangkat lunak telah diterapkan dan masalah hanya ditangani jika terjadi kerusakan atau kesalahan pada sistem[2].

   - **Keunggulan**: Dapat lebih efisien dalam hal sumber daya, karena hanya melakukan pemeliharaan berdasarkan masalah yang teridentifikasi
   - **Kelemahan**: Pendekatan ini bisa menyebabkan perangkat lunak menjadi tidak stabil dalam jangka panjang karena masalah dibiarkan berkembang sebelum diperbaiki.

2) **Model Pemeliharaan Proaktif**

   Model ini lebih berfokus pada antisipasi dan pencegahan masalah sebelum mereka terjadi. Pemeliharaan proaktif melibatkan kegiatan seperti pemantauan kinerja, pengujian, dan perbaikan yang dilakukan sebelum masalah teridentifikasi oleh pengguna. Ini mencakup refactoring kode, pembaruan perangkat lunak, atau memperbarui dokumentasi teknis[3].

   - **Keunggulan**: Lebih dapat mencegah masalah besar di masa depan dan meningkatkan kualitas perangkat lunak.
   - **Kelemahan**: Dapat memerlukan lebih banyak sumber daya di awal, karena melibatkan investasi untuk memantau dan mengidentifikasi masalah potensial.

3) **Model Pemeliharaan Terus-Menerus (Continuous Maintenance)**

   Model ini mencakup pemeliharaan yang dilakukan secara berkelanjutan selama perangkat lunak digunakan. Pemeliharaan tidak hanya dilakukan saat masalah muncul, tetapi dilakukan secara rutin untuk memastikan perangkat lunak tetap up-to-date, aman, dan berfungsi dengan baik. Pemeliharaan ini sering kali mencakup pembaruan perangkat keras atau perangkat lunak terkait dan mengintegrasikan teknologi baru[1].

   - **Keunggulan**: Memastikan perangkat lunak selalu dalam kondisi terbaik, menyesuaikan dengan perkembangan teknologi terbaru.
   - **Kelemahan**: Proses pemeliharaan yang berkelanjutan memerlukan komitmen sumber daya yang cukup besar dalam jangka panjang

4) **Model Pemeliharaan Fungsional (Functional Maintenance)**

   Model ini mengutamakan peningkatan fungsionalitas perangkat lunak setelah pengguna memberikan umpan balik. Jika ada fitur baru yang dibutuhkan oleh pengguna atau perubahan dalam kebutuhan bisnis, model pemeliharaan fungsional akan fokus pada penambahan atau modifikasi fitur perangkat lunak untuk memenuhi permintaan tersebut [2].

   - **Keunggulan**: Membantu perangkat lunak tetap relevan dengan perkembangan kebutuhan pengguna atau bisnis.
   - **Kelemahan**: Pemeliharaan fungsional dapat menyebabkan kompleksitas yang lebih tinggi dalam pengelolaan fitur dan kode.

5) **Model Pemeliharaan Refactoring**

   Refactoring adalah proses untuk meningkatkan struktur kode sumber perangkat lunak tanpa mengubah fungsionalitas eksternal. Model pemeliharaan ini berfokus pada penyusunan ulang kode untuk meningkatkan kualitas perangkat lunak, mempermudah pemeliharaan di masa depan, atau memperbaiki masalah desain yang ada[3].

   - **Keunggulan**: Memperbaiki kualitas kode tanpa menambah kompleksitas atau merusak fungsionalitas yang sudah ada.
   - **Kelemahan**: Memerlukan pemahaman yang mendalam tentang kode yang ada dan dapat menjadi waktu yang intensif

6) **Model Pemeliharaan Berbasis Perubahan (Change-Based Maintenance)**

   Model ini menekankan perubahan yang dilakukan terhadap perangkat lunak untuk menyesuaikan dengan perubahan yang terjadi, baik di lingkungan teknologi maupun kebutuhan pengguna. Model ini berfokus pada adaptasi perangkat lunak agar tetap relevan dan sesuai dengan standar atau kebutuhan baru. Pemeliharaan berbasis perubahan mencakup modifikasi kode sumber,

pembaruan modul, atau penyesuaian arsitektur perangkat lunak[4].

- **Keunggulan**: Dapat membuat perangkat lunak lebih fleksibel dan mampu menanggapi perubahan yang terjadi di luar kontrol langsung tim pengembang.
- **Kelemahan**: Perubahan yang terlalu sering dapat menyebabkan ketidakstabilan perangkat lunak jika tidak dikelola dengan baik.

7) **Model Pemeliharaan Outsourcing**

Dalam model ini, pemeliharaan perangkat lunak dilakukan oleh pihak ketiga atau vendor luar yang memiliki keahlian khusus dalam pemeliharaan perangkat lunak. Model ini banyak digunakan oleh perusahaan yang lebih memilih untuk mengalihkan tanggung jawab pemeliharaan kepada pihak luar untuk mengurangi biaya atau meningkatkan efisiensi[5]

- **Keunggulan**: Mengurangi beban internal dan memberikan akses ke keahlian spesifik dari pihak ketiga.
- **Kelemahan**: Mengandalkan pihak ketiga bisa menimbulkan masalah dalam koordinasi, pengelolaan kualitas, atau kontrol atas proses pemeliharaan

3. DAFTAR REFERENSI

1. ISO/IEC 14764:2006, *Software Engineering - Software Life Cycle Processes - Maintenance*. International Organization for Standardization
2. Sommerville, I. (2011). *Software Engineering* (9th ed.). Addison-Wesley.
3. Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill.
4. Fowler, M. (2018). *Refactoring: Improving the Design of Existing Code*. Addison-Wesley
5. Lacity, M. C., & Willcocks, L. P. (2017). *Robotic Process Automation: The Next Transformation in Business Services*. Palgrave Macmillan.

4. Daftar Bacaan

1. Sama seperti pada daftar referensi

5. 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	1-6

	b. Komponen perangkat lunak c. Model perangkat lunak d. Fungsi dan peran dari software engineer	
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) c. Integration testing dan user testing d. Faults, Error dan Failures	Idem
11	Review Teknik Pengujian Perangkat Lunak dari proses testing	Idem
12	Pengujian unit, Pengujian integrasi, Pengujian sistem, Pengujian Penerimaan	Idem
13	a. Quality assurance pada perangkat lunak b. Keamanan data akses	Idem
14	Definisi pemeliharaan perangkat lunak dan Konsep Pemeliharaan Perangkat lunak	Idem
15	Teknik pemeliharaan perangkat lunak (Pemeliharaan korektif, pemeliharaan adaptif, pemeliharaan perfektif, pemeliharaan preventif)	Idem

16	**UAS**	