Universitas Esa Unggul

Smart, Creative and Entrepreuerial

Konsep pemeliharaan PL

Istilah pemeliharaan perangkat lunak digunakan untuk menjabarkan aktivitas dari analis sistem (software engineering) yang terjadi pada saat hasil produk perangkat lunak sudah dipergunakan oleh pemakai (user).

Biasanya pengembangan produk perangkat lunak memerlukan waktu antara 1 sampai dengan 2 tahun, tetapi pada pase pemeliharaan perangkat lunak menghabiskan 5 sampai dengan 10 tahun. Aktivitas yang terjadi pada pase pemeliharaan antara lain:

– Penambahan atau peningkatan atau juga perbaikan untuk produk perangkat lunak
– Adaptasi produk dengan lingkungan mesin yang baru
– Pembetulan permasalahan yang timbul

“ Pemeliharaan sistem berawal begitu sistem baru menjadi operasional dan berakhir masa hidupnya ”

Jenis Pemeliharaan :

- Pemeliharaan Korektif : Pemeliharaan perangkat lunak dengan melakukan perbaikan kesalahan yang terjadi pada perangkat lunak
- Pemeliharaan Adaptif : Pemeliharaan perangkat lunak dengan melakukan penyesuaian fungsi-fungsi yang ada pada perangkat lunak sehingga lebih memudahkan user.
- Pemeliharaan Penyempurnaan : Pemeliharaan perangkat lunak dengan melakukan pengembangan / peningkatan terhadap perangkat lunak yang telah ada.
- Pemeliharaan Preventif : Pemeliharaan perangkat lunak dengan perombakan secara total atau melakukan perekayasaan kembali pada perangkat lunak yang ada.

Siklus Hidup Pemeliharaan Sistem (SMLC)

Tahapan SMLC :

- Memahami Permintaan Pemeliharaan
- Mentransformasi permintaan pemeliharaan menjadi pengubahan
- Menspesifikasi perubahan
- Mengembangkan perubahan

- Menguji perubahan
- Melatih pengguna dan melakukan test penerimaan
- Pengkonversian dan meluncurkan operasi
- Mengupdate Dokumen
- Melakukan pemeriksaan Pasca implementasi

Maintainability (Kemampuan pemeliharaan sistem)

Prosedur untuk peningkatan maintainability :

- Menerapkan SDLC dan SWDLC
- Menspesifikasi definisi data standar
- Menggunakan bahasa pemrograman standart
- Merancang modul-modul yang terstruktur dengan baik
- Mempekerjakan modul yang dapat digunakan kembali
- Mempersiapkan dokumentasi yang jelas, terbaru dan komprehensif
- Menginstall perangkat lunak, dokumentasi dan soal-soal test di dalam sentral repositor sistem CASE atau CMS (change management system)

Tiga pendekatan untuk menyusun Pemeliharaan sistem :

- Pendekatan Pemisahan -> Pemeliharaan dan Pemeliharaan
- Pendekatan Gabungan -> Menggabungkan personalia penyusun dan pemelihara menjadi sebuah kelompok utama sistem informasi
- Pendekatan Fungsional -> Variasi dari pendekatan gabungan dengan memindahkan tenaga profesional sistem dari sistem informasi dan menugasi mereka pada fungsi bisnis untuk penyusunan maupun pemeliharaan.

Ada 5 CASE Tools yang membantu pemeliharaan sistem dari sistem lama dan membantu memecahkan kemacetan timbunan sistem baru yang belum dikerjakan :

- Rekayasa Maju (Forward engineering)
- Rekayasa Mundur (Reverse engineering)
- Rekayasa Ulang (Reengineering)
- Restrukturisasi (restrukturing)
- Sistem Pakar Pemeliharaan (Maintenance expert system)

Mengelola Pemeliharaan Sistem

- Menetapkan Kegiatan Pemeliharaan Sistem

- Mengawali dan merekam kegiatan pemeliharaan sistem tidak terjadwal (Form Maintenance Work Order : Pekerjaan yang diperlukan/dilakukan, waktu yang diperkirakan dibandingkan dengan waktu yang sebenarnya, kode pemeliharaan, biaya pemeliharaan)
- Menggunakan sistem perangkat lunak helpdesk
- Mengevaluasi aktivitas pemeliharaan sistem
- Mengoptimalkan program pemeliharaan sistem

Martin & McClure (1983)

- …perubahan yang harus dilakukan terhadap program komputer setelah diserahkan kepada pelanggan / pengguna

Von Mayrhauser (1990)

- Pemeliharaan mencakup hidup sistem perangkat lunak sejak diinstalasi sampai tidak digunakan lagi

Martin & McClure (1993)

Pemeliharaan P/L dilakukan dengan maksud tertentu

- Memperbaiki kesalahan
- Memperbaiki kekurangan pada rancangan
- Interaksi dengan sistem lain
- Perluasan sistem
- Melakukan perubahan yang perlu
- Melakukan perubahan file atau basisdata
- Meningkatkan rancangan
- Mengubah program untuk dapat memanfaatkan P/K, P/L, fitur sistem, fasilitas komunikasi.

Semua sistem informasi sewaktu-waktu berubah. Pemeliharaan sistem adalah kegiatan yang membuat perubahan ini.

KEPERLUAN PEMELIHARAAN SISTEM

Sistem perlu dipelihara karena beberapa hal, yaitu :

1. Sistem memiliki kesalahan yang dulunya belum terdeteksi, sehingga kesalahan-kesalahan sistem perlu diperbaiki.

2. Sistem mengalami perubahan-perubahan karena permintaan baru dari pemakai sistem.
3. Sistem mengalami perubahan karena perubahan lingkungan luar (perubahan bisnis).
4. Sistem perlu ditingkatkan.

Biaya pemeliharaan sistem sering diabaikan. Kenyataannya biaya pemeliharaan sistem merupakan biaya yang cukup besar.

Biaya pemeliharaan perangkat lunak telah terus menerus naik selama 25 tahun terakhir.

Beberapa perusahaan membelanjakan 80% atau lebih dari anggaran sistem mereka pada pemeliharaan perangkat lunak.

JENIS PEMELIHARAAN SISTEM

Pemeliharaan sistem dapat digolongkan menjadi empat jenis :
- Pemeliharaan Korektif
- Pemeliharaan Adaptif
- Pemeliharaan Perfektif (Penyempurnaan)
- Pemeliharaan Preventif

Pemeliharaan Korektif

Pemeliharaan korektif adalah bagian pemeliharaan sistem yang tidak begitu tinggi nilainya dan lebih membebani, karena pemeliharaan ini mengkoreksi kesalahan-kesalahan yang ditemukan pada saat sistem berjalan.

Umumnya pemeliharaan korektif ini mencakup kondisi penting atau bahaya yang memerlukan tindakan segera. Kemampuan untuk mendiagnosa atau memperbaiki kesalahan atau malfungsi dengan cepat sangatlah berharga bagi perusahaan.

Pemeliharaan Adaptif

Pemeliharaan adaptif dilakukan untuk menyesuaikan perubahan dalam lingkungan data atau pemrosesan dan memenuhi persyaratan pemakai baru.

Lingkungan tempat sistem beroperasi adalah dinamik, dengan demikian, sistem harus terus merespon perubahan persyaratan pemakai. Misalnya, Undang-Undang Perpajakan yang baru mungkin memerlukan suatu perubahan dalam kalkulasi pembayaran bersih.

Umumnya pemeliharaan adatif ini baik dan tidak dapat dihindari.

Pemeliharaan Penyempurnaan

Pemeliharaan penyempurnaan mempertinggi cara kerja atau maintainabilitas (kemampuan untuk dipelihara). Tindakan ini juga memungkinkan sistem untuk memenuhi persyaratan pemakai yang sebelumnya tidak dikenal.

Ketika membuat perubahan substansial modul apapun, petugas pemeliharaan juga menggunakan kesempatan untuk mengupgrade kode, mengganti cabang-cabang yang kadaluwarsa, memperbaiki kecerobohan, dan mengembangkan dokumentasi.

Sebagai contoh, kegiatan pemeliharaan ini dapat berbentuk perekayasaan ulang atau restrukturisasi perangkat lunak, penulisan ulang dokumentasi, pengubahan format dan isi laporan, penentuan logika pemrosesan yang lebih efisien, dan pengembangan efisiensi pengoperasian perangkat.

Pemeliharaan Preventif

Pemeliharaan Preventif terdiri atas inspeksi periodik dan pemeriksaan sistem untuk mengungkap dan mengantisipasi permasalahan.

Karena personil pemeliharaan sistem bekerja dalam sistem ini, mereka seringkali menemukan cacat-cacat (bukan kesalahan yang sebenarnya) yang menandakan permasalahan potensial. Sementara tidak memerlukan tindakan segera, cacat ini bila tidak dikoreksi di tingkat awal, jelas sekali akan mempengaruhi baik fungsi sistem maupun kemampuan untuk memeliharanya dalam waktu dekat.

Daftar Pustaka

- Software Engineering Ian Sommerville
- Software Engineering Roger S.Pressman

Sistematika dokumentasi

Menurut Roger S. Pressman [1], ada tiga hal yang dapat mendefinisikan suatu perangkat lunak yaitu:
(1) program komputer yang bila dieksekusi akan memberikan fungsi dan kerja seperti yang diinginkan.

(2) struktur data yang memungkinkan program memanipulasi informasi secara proposional, dan

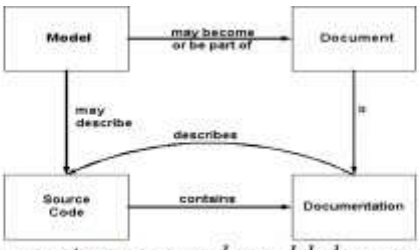
(3) dokumen yang menggambarkan operasi dan kegunaan program. Sehingga dapat dikatakan sebuah program komputer belum dapat disebut perangkat lunak tanpa disertai dengan dokumentasinya. Hal ini menunjukkan betapa pentingnya dokumentasi pada pembuatan sebuah perangkat lunak, tetapi banyak pengembang perangkat lunak yang kurang memperhatikan masalah dokumentasi.

Dokumentasi merupakan sebuah artefak yang tujuannya untuk menyampaikan informasi tentang sistem perangkat lunak yang menyertainya . Selain itu dokumentasi mempunyai fungsi sebagai berikut :
1. Bertindak sebagai media komunikasi antar anggota pengembang tim,
2. Penyimpanan sistem informasi untuk digunakan oleh maintenance engineers,
3. Membantu manajer proyek dalam merencanakan, mengatur anggaran, dan penjadwalan dalam proses pembangunan perangkat lunak,
4. Memberi penjelasan kepada pengguna bagaimana cara menggunakan dan mengelola sistem yang dibangun.

Sebagai tempat penyimpanan informasi, dokumen semestinya harus berisi informasi yang lengkap, valid, mudah dimengerti, dan up-to-date. Tapi sayangnya banyak pengembang yang membiarkan dokomen yang dibuat tidak memberikan informasi yang lengkap atau informasi yang tidak diperbaharui (out-of-date).
Beberapa software engineers berpendapat bahwa “my code is self-documenting”. Mereka beranggapan cukup dengan source code sudah merupakan dokumentasinya, sehingga tidak diperlukan dokumen tambahan Hal ini mungkin dapat berlaku jika program yang dibuat untuk dirinya sendiri. Tetapi bagaimana jika program tersebut digunakan oleh orang lain

atau program tersebut sebagai bagian dari sebuah sistem perangkat lunak yang dikerjakan oleh banyak orang ? Software engineers yang lain mungkin dapat mengerti jalannya program dengan membaca kode tersebut, tetapi tetap akan membutuhkan waktu yang lebih lama dibandingkan dengan membaca sebuah dokumen yang menjelaskan secara rinci tetang program tersebut. Scott Ambler dalam thesis Andrew Forward menjelaskan hubungan antara source code, model,
dokumen, dan dokumentasi . Ambler menjelaskan bahwa sebuah dokumentasi merupakan penjelasan dari kode yang dibuat. Sebuah model juga mungkin menjelaskan kode, dan model ini dapat menjadi dokumen atau bagian dari dokumen.

Hubungan tersebut dapat dilihat pada gambar berikut



Gambar 1: Hubungan antara *source code, model, document,* dan *documentation*

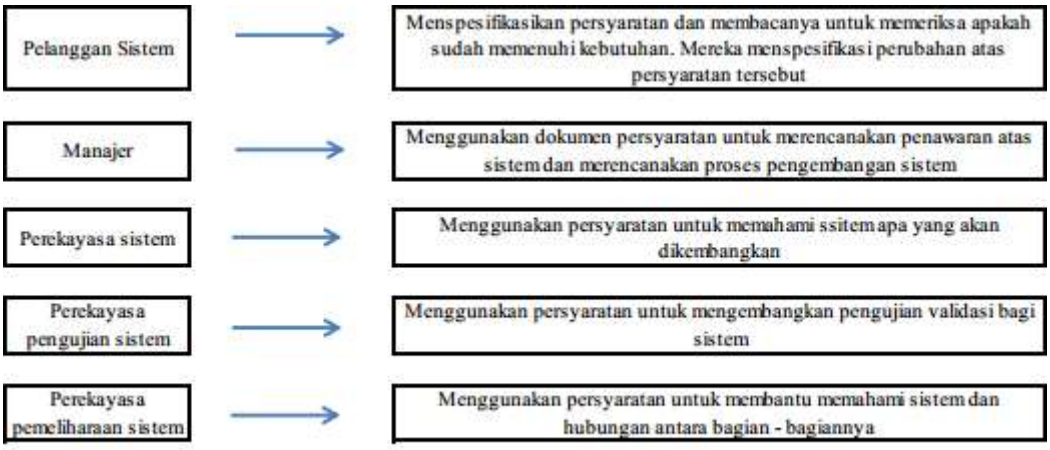
Ian Sommerville mengklasifikasi dokumentasi ke dalam dua kelas, yaitu dokumentasi proses dan dokumentasi produk .
Dokumentasi proses merupakan dokumen yang menyimpan semua proses dari pembangunan dan pemeliharaan perangkat lunak, termasuk perencanaan, penjadwalan, lembar kerja, serta memo maupun email.
Sedangkan dokumen produk yaitu dokumen yang merupakan penjelasan dari perangkat lunak yang dibangun. Dokumentasi pengguna dan dokumentasi sistem termasuk dalam dokumen produk. Dokumentasi pengguna yaitu dokumen yang menjelaskan tentang bagaimana penggunaan dari produk perangkat lunak tersebut, sedangkan dokumen sistem yaitu semua dokumen yang menjelaskan tentang sistem yang dibagun, mulai dari spesifikasi kebutuhan sampai dengan pengujian perangkat lunak.

Pada sumber lain ada yang mengklasifikasikan dokumentasi ke dalam empat bagian yaitu dokumen kebutuhan, arsitektur dan desain, dokumen teknis, dokumen end user, dan dokumen pemasaran . Dokumen kebutuhan merupakan dokumen yang menjelaskan tentang atribut, kemampuan, karakterisitik, atau kualitas dari suatu sistem yang merupakan dasar dari

pembuatan suatu perangkat lunak. Dokumen arsitektur dan disain yaitu dokumen yang menjelaskan tentang arsitektur sistem dan prinsip – prinsip konstruksi yang akan digunakan dalam desain komponen perangkat lunak. Dokumen teknis merupakan dokumentasi dari kode, algoritma dan interface. Dokumen end user merupakan dokumen manual tentang
bagaimana perangkat lunak tersebut digunakan. Dokumen pemasaran berisi bagaimana cara pemasaran dari produk dan analisis permintaan pasar.

2.1 Dokumen Persyaratan Perangkat Lunak

Dokumen persyaratan perangkat lunak (SRS/Software Requirements Specification) merupakan persyaratan resmi mengenai apa yang dituntut dari pengembang sistem [7]. Dokumen berisi persyaratan user untuk sistem dan spesifikasi secara rinci dari persyaratan sistem. Berikut ilustrasi contoh dokumen persyaratan perangkat lunak dan bagaimana pemanfaatannya [7].

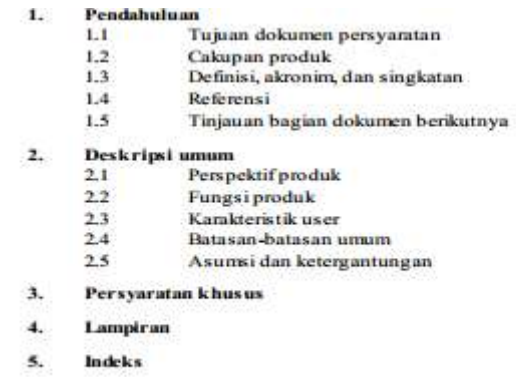


Gambar 2: Ilustrasi pemanfaatan dokumen persyaratan perangkat lunak

Heninger dalam buku Ian Sommerville [7] mengusulkan bahwa ada enam persyaratan yang harus dipenuhi oleh dokumen persyaratan perangkat lunak yaitu:

- Menspesifikasi perilaku sistem eksternal.
- Menspesifikasi batasan – batasan implementasi.
- Mudah diubah
- Berfungsi sebagai alat bantu referensi bagi pemelihara sistem.

Lembaga IEEE telah membuat standar untuk dokumen persyaratan perangkat lunak (IEEE/ANSI 830-1993). Berikut outline yang disarankan oleh IEEE untuk dokumen persyaratan perangkat lunak:



Gambar 3: Outline dokumen persyaratan perangkat lunak [7]

Persyaratan khusus mencakup persyaratan fungsional, non-fungsional dan interface yang merupakan bagian penting dari dokumen persyaratan perangkat lunak. Standar dari IEEE memberikan saran apa saja yang perlu ditulis di dokumen persyaratan perangkat lunak, tetapi pemanfaatannya tergantung dari kebutuhan pengembang dan pengguna perangkat lunak
tersebut.

2.2 Dokumentasi Desain

Dokumentasi desain berisi penjelasan rinci tentang inti teknis dari rekayasa perangkat lunak yang meliputi struktur data, arsitektur program, interface dan detail prosedural [1]. Gambar 3 menunjukan contoh outline dari dokumen desain yang diambil dari buku Pressman [1]. Berikut penjelasan perbagian dari Pressman mengenai outline tersebut:

- Bagian I berisi ruang lingkup dari kerja desain.
- Bagian II berisi desain data, struktur file eksternal dan referensi silang yang menghubungkan objek data denganfile tertentu.
- Bagian III berisi desain arsitektur.
- Bagian IV dan V, pada bagian ini berkembang pada saat desain interface dan procedural dimulai.
- Bagian VI berisi referensi silang yang bertujuan utnuk menetapkan bahwa semua persyaratan dipenuhi oleh desain
perangkat lunak dan menunjukkan modul mana yang krites terhadap implementasi persyaratan spesifik.
- Bagian VII berisi tahap pertama dari pembuatan dokumentasi pengujian.
- Bagian VIII dan IX berisi data tambahan meliputi deskripsi algoritma, prosedur alternative, data dalam bentuk
tabel, kutipan dari dokumen lain, dan informasi relevan lainnya.

2.3 Dokumentasi Pengujian

Pengujian perangkat lunak merupakan sederetan langkah yang digunakan untuk melakukan pengujian atau pengecekan terhadap unit program ataupun sistem lengkap dari perangkat lunak untuk menjamin bahwa persyaratan sistem telah dipenuhi. Pengujian memastikan bahwa program tersebut telah berfungsi sebagaimana mestinya. Rencana, hasil serta prosedur pengujian harus didokumentasikan dalam suatu dokumen pengujian. Gambar 5 menunjukkan outline dari dokumen pengujian.

2.4 Dokumentasi Pengguna

Dokumentasi pengguna merupakan dokumen yang menyertai sebuah perangkat lunak yang berisi penjelasan secara detail tentang perangkat lunak tersebut. Dokumen pengguna menjelaskan setiap feature dari perangkat lunak dan menjelaskan bagaimana cara menggunakan setiap feature tersebut. Selain itu dokumen pengguna juga dapat memberikan penjelasan terhadap setiap masalah atau error yang terjadi dan bagaimana cara menanganinya. Dokumen pengguna dapat berupa dokumen cetak, elektronik, dokumen online yang mudah diakses ataupun gabungan dari semuanya. Dengan adanya dokumen pengguna ini, pengguna dapat dimudahkan dalam menggunakan perangkat lunak tersebut.

IEEE telah mendefinisikan standar untuk dokumentasi pengguna. Pada standar tersebut, IEEE mendefinisikan komponen-komponen yang semestinya ada pada dokumentasi pengguna. Komponen yang disarankan oleh IEEE dapat dijadikan panduan untuk membuat dokumentasi pengguna.

3. KUALITAS DOKUMENTASI

Berdasarkan hasil survei yang dilakukan oleh Andrew Forward, software engineers mengungkapkan dokumen seperti apa yang dianggap berkualitas bagus, jelek dan sangat buruk [9].

1. Dokumen berkualitas bagus

- Arsitektur dan informasi dokumentasi lainnya selalu valid atau setidaknya menyediakan panduan sejarah yang dapat berguna untuk pemeliharaan perangkat lunak.
- Inline comments pada kode program cukup baik dalam memberikan informasi yang berguna untuk pemeliharaan perangkat lunak.

2. Dokumen berkualitas jelek

- Dokumentasi untuk semua jenis sering sekali tidak diperbaharui (out of date)
- Sistem mempunyai terlalu banyak dokumentasi
- Penulisan dokumentasi yang buruk
- Pengguna kesulitan menemukan isi yang berguna dalam dokumentasi
- Pembuatan dokumentasi yang memakan waktu yang tidak sebanding dengan keuntungan dari
dokumentasi tersebut

3. Dokumen berkualitas sangat buruk
- Sebuah dokumentasi yang informasinya tidak dapat dipercaya

Secara umum dokumentasi yang bagus yaitu dokumen yang ditulis dengan baik, mudah dibaca dan dimengerti serta memberikan informasi yang lengkap dan akurat. Walaupun pembuatan dokumen yang seperti ini mungkin akan menyita waktu yang lebih banyak, tetapi dengan dokumen yang baik akan sangat membantu baik itu pengembang maupun pengguna program tersebut.

Berdasarkan survei Andrew Forward [10] menunjukkan bahwa isi dokumen merupakan atribut dokumen yang paling penting dari sebuah dokumentasi perangkat lunak. Tiga atribut lainnya yang dianggap penting yaitu up-to-date, availability, use of examples. Atribut-atribut tersebut yang sangat menentukan kualitas suatu dokumen, walaupun atribut lainnya juga tidak kalah pentingnya.

4. ALAT BANTU
Ada banyak software tool yang dapat digunakan untuk membantu membuat dokumentasi perangkat lunak. Penggunaan tool dapat mempercepat dan mempermudah dalam pembuatan dokumentasi.

Teknologi lainnya yang digunakan pengembang berdasarkan survei tersebut yaitu Ar goUML, Visio, FrameMaker, AuthorIT, whiteboards dan digital cameras, JUnit dan XML editors. Word processors paling banyak digunakan karena merupakan tool yang gampang digunakan dan lebih fleksibel. Tool yang berguna lainnya yaitu software sejenis mindmap (freemind).
Software tersebut dapat membantu untuk pengembang dalam menuliskan dokumentasi terkait dengan pembuatan perangkat lunak.