UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Pengenalan RPL

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Teknik Informatika	01	87011	Tim Dosen

Abstrak

Modul ini menjelaskan macam dari perangkat lunak dengan komponen-komponennya. Aplikasi yang dihasilkan dengan perangkat lunak.

Kompetensi

Mampu memahami kontrak perkuliahan dan pengertian RPL.

1. Perbedaan PL & Ilmu Komputer

Perbedaan perangkat lunak dengan ilmu komputer. **Ilmu komputer** seringkali didiskripsikan sebagai suatu studi sistematis pada proses-proses algoritma yang menjelaskan dan mentransformasikan informasi seperti halnya di sini adalah teori, analisis, disain, efisiensi, penerapan dan aplikasinya. Sedangkan **perangkat lunak** merupakan data elektronik yang disimpan sedemikian rupa oleh komputer itu sendiri, data yang disimpan ini dapat berupa program atau instruksi yang akan dijalankan oleh perintah, maupun catatan-catatan yang diperlukan oleh komputer untuk menjalankan perintah yang dijalankannya. Jadi perangkat lunak itu dapat berupa program atau prosedur. Perbedaan antara RPL dengan ilmu komputer adalah intinya, ilmu komputer **berhubungan dengan teori dan metode yang mendasari sistem komputer dan perangkat lunak**, sedangkan RPL **berhubungan dengan praktek dalam memproduksi perangkat lunak**.

2. Perbedaan RPL & Rekayasa Sistem

Rekayasa system mempunyai pengertian suatu sistem yang mampu memilih alat bantu yang baik dalam perencanaan maupun dalam penerapan perangkat lunak dan memiliki teknik yang baik untuk menilai kualitas dari perangkat lunak yang dihasilkan, serta mampu mengkoordinasikan, mengontrol, dan mengatur pelaksanaan pekerjaan pembuatan perangkat lunak. Sedangkan **rekayasa perangkat lunak** itu adalah aplikasi dari ilmu komputer yang membangun system perangkat lunak yang nantinya perangkat lunak itu akan dipilih kualitas dan tekniknya oleh rekayasa sistem. Rekayasa sistem berkaitan dengan semua aspek dalam pembangunan sistem berbasis komputer termasuk hardware, rekayasa PL dan proses. RPL adalah bagian dari rekayasa sistem yang meliputi pembangunan PL, infrasktruktur, kontrol, aplikasi dan database pada sistem. Perbedaan RPL dengan Rekayasa Sistem intinya Rekayasa sistem **berkaitan dengan semua aspek dalam pembangunan sistem berbasis komputer termasuk hardware, rekayasa PL dan proses**. RPL adalah **bagian dari rekayasa sistem yang meliputi pembangunan PL, infrasktruktur, kontrol, aplikasi dan database pada sistem**.

- **Pengembangan Perangkat Lunak** : Perangkat lunak yang memenuhi spesifikasi harus di produksi
- **Validasi Perangkat Lunak** : Perangkat lunak harus divalidasi untuk menjamin bahwa perangkat lunak melakukan apa yang diinginkan oleh pelanggan.
- **Evolusi Perangkat Lunak** : Perangkat lunak harus berkembang untuk memenuhi kebutuhan pelanggan.

Ada empat fase utama pada proses rekayasa persyaratan:

1. Studi kelayakan. Dibuat perkiraan apakah user yang diidentifikasi puas menggunakan perangkat lunak dan teknologi perangkat keras yang dipakai pada saat ini. Studi ini akan memutuskan apakah sistem yang diusulkan efektif dalam hal biaya dari sudut pandang bisnis dan apakah sistem dapat dikembangkan dengan keterbatasan anggaran yang tersedia. Hasil dari studi kelayakan ini adalah informasi keputusan apakah kita akan terus dengan analisis yang lebih rinci atau tidak.
2. Elisitasi dan analisis persyaratan. Ini merupakan proses penurunan persyaratan sistem melalui observasi sistem yang ada, diskusi dengan user yang akan memakai dan yang mengadakan, analisis pekerjaan, dll. Proses ini bisa melibatkan pengembangan satu atau lebih model dan prototipe sistem. Hasil fase ini akan membantu analis memahami sistem yang akan dispesifikasi.
3. Spesifikasi persyaratan. Merupakan kegiatan menerjemahkan informasi yang dikumpulkan pada kegiatan analisis menjadi dokumen yang mendefinisikan serangkaian persyaratan. Dua jenis persyaratan bisa dicakup pada dokumen ini. Persyaratan user merupakan pernyataan abstrak persyaratan sistem untuk pelanggan dan end user sistem; persyaratan sistem merupakan deskripsi yang lebih rinci mengenai fungsionalitas yang akan diberikan.
4. Validasi persyaratan. Kegiatan ini memeriksa apakah persyaratan dapat direalisasikan, konsisten, dan lengkap. Pada proses ini kesalahan pada dokumen persyaratan pada akhirnya akan ditemukan. Kesalahan ini kemudian dimodifikasi untuk menyelesaikan masalahnya.

SOFTWARE ENGINEERING

(REKAYASA PIRANTI LUNAK)

ada bagian ini penekanan pembahasannya adalah terhadap pertanyaan-pertanyaan berikut :

- Pendahuluan
- Bagaimana definisi / apa itu perangkat lunak komputer ?
- Mengapa kita berusaha keras membangun system berbasis computer yang berkualitas tinggi ?
- Bagaimana kita dapat mengkategorisasi domain aplikasi untuk perangkat lunak computer ?
- Mitos-mitos apa yang masih ada mengenai perangkat lunak ?
- Apa itu “proses perangkat lunak” ?
- Apakah ada cara generik untuk menilai kualitas proses ?
- Model-model proses apa yang dapat diterapkan untuk pengembangan perangkat lunak ?
- Apa perbedaan model proses linier dan model proses iteratif ?
- Apa kekuatan dan kelemahannya ?
- Model proses canggih apa yang telah diusulkan untuk rekayasa perangkat lunak ?

Definisi

- Suatu proses evolusi dan pemanfaatan alat dan teknik untuk pengembangan software.
- Penetapan dan penggunaan prinsip – prinsip rekayasa dalam rangka mendapatkan software yang ekonomis yaitu software yang terpercaya dan bekerja efisien pada mesin (komputer)

- Terkandung komponen nilai dari SE yang berbentuk program dengan tugas - tugas :
- Membuat suatu desain aplikasi yang ada dilingkungan tugas atau pekerjaan.
- Buat deadline : pendahuluan, perumusan masalah, analisis, desain, dan kesimpulan.

I. DASAR-DASAR PENGERTIAN SOFTWARE

a. Perilaku dinamis dari program komputer
b. Program adalah ekspresi intelektual dalam
c. Program terdiri dari algoritma
d. Bahasa yang digunakan adalah tingkatan ultra rendah (Binary)
e. Program diterjemahkan (kompilasi, interprestasi, Assembly) untuk menjalankan.
f. Transformasi atas dasar angka suatu pernyataan program lebih mudah.
g. Sistem elemen software bersifat logika bukan fisik.
h. Penggunaan komputer untuk tingkat yang lebih tinggi (Aplikasi Program)

- SE berkaitan dengan pembangunan produk program
- SE adalah disiplin rekayasa (Engineering)
- Ilmuwan membangun dalam usaha untuk belajar
- Engineering belajar dalam usaha untuk membangun

- Kegiatan software engineering :
  - Analisa kebutuhan dan spesifikasi
  - Estimasi “Feasibility” dan sumber daya
  - Desain solusi perangkat lunak berbasis komputer

- Implementasi desain berupa program
- Pengukuran kwalitas hasil akhir berupa software

III. KARAKTERISTIK SOFTWARE

1. Sistem elemen software bersifat logika bukan fisik.
2. Software dikembangkan atau rekayasa.
3. Software tidak rusak.

IV. TUJUAN SOFTWARE ENGINEERING

- Biaya produksi rendah
- Kinerja yang tinggi
- Biaya perawatan yang rendah
- Keandalan yang tinggi
- Penyerahan tepat waktu

V. KOMPONEN BIAYA PENGEMBANGAN

- Biaya relatif pentahapan pengembangan perangkat lunak

VI. KINERJA PROGRAM

Dipengaruhi oleh :

- Keandalan perangkat keras
- Kebutuhan penggunanya yang ingin lebih baik

VII. PORTABILITAS

- Kemampuan transfer perangkat lunak dari suatu jenis komputer ke lainnya dengan biaya usaha minimum
- Mengurangi ketergantungan hanya pada suatu pemasok
- Lebih bersifat non-teknis/ politis dari pada teknis

VIII. PERAWATAN

- Perbaikan membutuhkan waktu
- Perubahan perangkat lunak juga butuh waktu
- Perawatan berusaha membutuhkan usaha yang membutuhkan perhatian cukup besar.
- Biaya relatif pentahapan pengembangan perangkat lunak

IX. KEANDALAN SYSTEM

- Dibutuhkan saat kegiatan system yang berkelanjutan
- Aplikasi yang berbeda membutuhkan tingkat keandalan yang berbeda.
- Keandalan perangkat lunak perangkat keras

X. APLIKASI – APLIKASI SOFTWARE

1. Sistem Software.
    - Compilers
    - Editors
    - Operating System Components
    - Telecommunication Processors
    - File Management Utilities
    - Operating System Components
2. Real – time Software
    - Software yang mengukur / menganalisis / mengontrol kejadian – kejadian yang sesungguhnya (sedang berlangsung)
3. Business Software
    - Payroll (penggajian)
    - Account Receivable (Piutang Usaha)
    - Account Payable (Hutang)
    - Inventory (Persediaan)
4. Engineering and Scientific Software
    - Computer Aided Design (CAD)
    - System Simulation
5. Embedded Software
    - Keypad pada microwave oven
    - Fungsi – fungsi digital pada mobil : Fuel Control, Dashboard Displays, Breaking Systems

6. Personal Computer / PCI Software
    - Word Processing
    - Spread Sheets
    - Computer Graphics
    - Database Management
    - Personal and Business Financial Appliations
7. Artifical Intelligent Software
    - Extern Systems (Knowledge – Pased Systems)
    - Pattern Recognition (Image and Voice)
    - Thasram Proving
    - Game Playing

XI. METODE :

Metode adalah cara bagaimana kita secara teknis menyediakan pembangunan software, yang terdiri dari :

1. Perencanaan proyek dan estimasi
2. Analisis kebutuhan sistem dan software
3. Rancangan struktur data
4. Arsitektur program
5. Algoritma Prosedur
6. Pengkodean
7. Testing
8. Pemeliharaan

XII. ALAT BANTU :

- Menyediakan dukungan otomatis / semi otomatis untuk metode COMPUTER AIDED SOFTWARE ENGINEERING (CASE)
- Fungsi →Mengkombinasikan software, hardware, dan software engineering database.

XIII. PROSEDUR :

1. Penggabungan metode dan alat bantu.
2. Prosedur mendefinisikan urutan (sequence) metode.
3. Prosedur mendefinisikan keluaran : dokumen, laporan, dan formulir yang dibutuhkan.
4. Prosedur mendefinisikan kontrol yang membantu keyakinan kualitas dan perubahan koordinasi.
5. Prosedur mendefinisikan 'milestone' yang memungkinkan manager memperkirakan kemajuan.

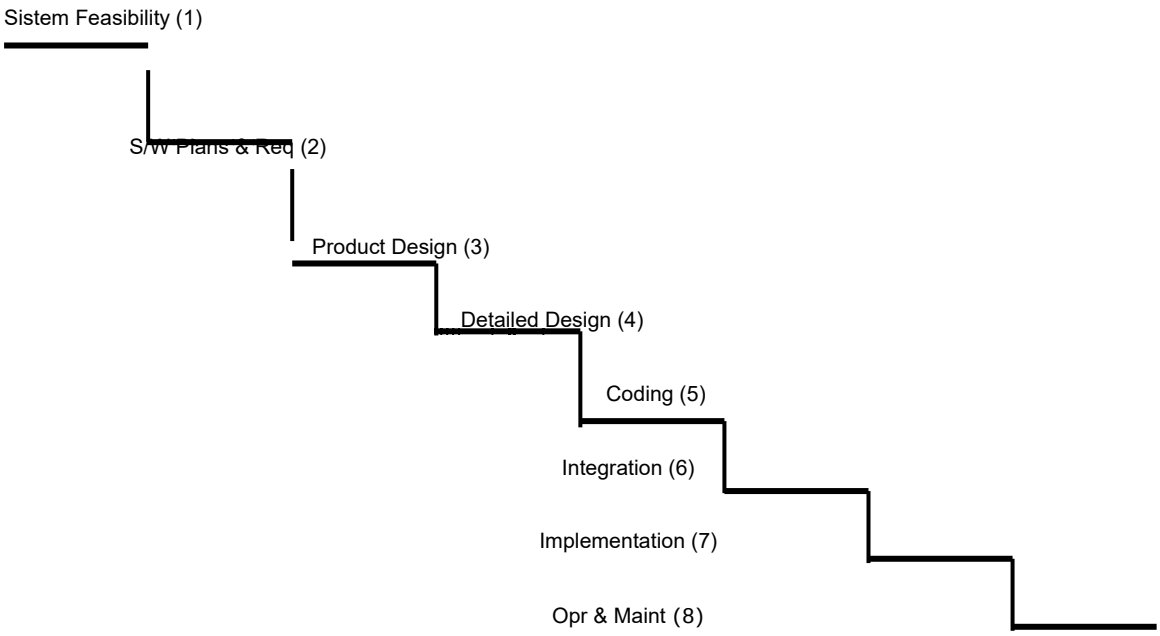
XIV. PARADIGMA REKAYASA SOFTWARE DIPILIH BERDASARKAN

1. Sifat dari proyek dan aplikasi
2. Metode dan alat bantu yang digunakan
3. Kontrol dan keluaran yang dibutuhkan

XV. DAUR HIDUP REKAYASA SOFTWARE (Software Engineering Life Cycle)

Pendekatan sistematis dan berurutan untuk pengembangan software.

Gambar the classic cycle (waterfall model) dapat dilihat pada halaman berikut :



Adapun personil yang terlibat dalam setiap tahapan adalah :

1. SA, Users dan Finance
2. SA dan Users
3. SA
4. SA
5. Programmers, Testers
6. SA, Programmers, Testers dan System Administrator
7. System Administrator, Testers dan Users
8. Maintenance staf

SA memiliki tanggung jawab untuk mengidentifikasi kebutuhan sistem dan merancang produk PL yang semestinya sedangkan Syst Adm. bertanggung jawab dalam implementasi perangkat lunaknya.

XVI. Aktivitas

1. Rekayasa sistem
2. Analisa kebutuhan software
3. Rancangan (design)
    i. Struktur data
    ii. Arsitektur software
    iii. Prosedur detil
4. Pengkodean (coding)
5. Testing

    Fokus pada logical internal dari software
6. Pemeliharaan (*maintenance*)
    i. Akomodasi terhadap perubahan lingkungan luar
    ii. Pemakaian menginginkan peningkatan fungsi atau kinerja

XVII. Prototyping

Suatu proses yang memungkinkan pembangun software menciptakan sebuah model dari software yang akan dibangun. Prototyping dapat dibangun dalam beberapa bentuk, baik dalam ukuran kecil (miniature rumah dalam pameran real estate, miniatur pesawat) maupun dalam ukuran sebenarnya seperti prototype Boeing 767 yang diujicobakan terbang dan mendarat ke berbagai negara dan prototype mobil Toyota Rush yang juga diujicobakan dalam kelaikan jalan di daerah bebatuan, menanjak, berkelok, menurun dsb, untuk menguji ketahanan dan kelayakan perangkat pendukung yang ada seperti rem, suspensi, jarak tempuh, dll. Hal ini bertujuan agar bila ada kekurangan dalam uji coba tersebut, dapat segera diperbaiki sebelum diproduksi dalam jumlah yang banyak.

XVIII. KESIMPULAN

a. Pengulangan pernyataan bahwa biaya perangkat lunak lebih mahal dari pada perangkat keras.

b. Tumbuhnya jumlah ketersediaan paket standar perangkat lunak.

c. Propaganda yang menyatakan bahwa membuat program semakin mudah.

d. Komponen Biaya Pengembangan : biaya relatif pertahapan pengembangan perangkat lunak.

e. Jumlah relatif kesalahan yang terjadi saat pertahapan pengembangan Perangkat Lunak.

UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Konsep Dasar RPL

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Informatika	02	87011	Diky Firdaus, S.Kom., MM.

Abstrak

Modul ini menjelaskan mengenai arti dan definisi dari Perangkat Lunak, jenis-jenis Perangkat Lunak dan pentingnya RPL.

Kompetensi

Mampu memahami konsep dasar RPL.

1. Tujuan Rekayasa Perangkat Lunak

A. Menghasilkan sebuah perangkat lunak yang berkualitas. Yang dimaksud dengan berkualitas dapat dilihat dari tiga sisi, sisi sponsor (individu atau organisasi yang telah mengeluarkan biaya dalam pembangunan perangkat lunak), sisi pemakai (siapapun yang menggunakan perangkat lunak tersebut), sisi *maintainer / modifier* (yang memelihara dan memodifikasi perangkat lunak tersebut). Untuk lebih jelasnya lihat gambar 2.1.

Sisi Sponsor :

Tujuan utama sponsor adalah menghasilkan dan atau menghemat uang. Sponsor ingin menggunakan perangkat lunak tersebut untuk meningkatkan produktivitas organisasi. Sponsor mengharapkan untuk dapat menghasilkan sebuah layanan dengan biaya yang rendah tetapi masuk akal. Karena itu sistem yang dibuat harus handal, fleksibel dan efisien. Selain itu biaya dari pemeliharaan, modifikasi dan peningkatan dari sistem tersebut harus serendah mungkin.

Sisi Pemakai :

Bagi pemakai perangkat lunak adalah alat untuk membantu menyelesaikan tugas-tugasnya. Karena itu perangkat lunak harus menyediakan fungsi-fungsi yang dibutuhkan oleh pemakai. Perangkat lunak juga harus handal dan efisien, perangkat lunak harus dapat menghasilkan *output* yang konsisten. Selain itu pemakai harus merasa perangkat lunak yang dibuat mudah untuk dipelajari, mudah digunakan dan mudah untuk diingat.

Sisi Maintainer/modifier :

Yang diinginkan oleh *maintainer/modifier* adalah perangkat lunak tersebut memiliki sangat sedikit *error* pada saat peng*install*an pertama (catatan : sangat kecil kemungkinannya untuk menghasilkan perangkat lunak yang 100 % bebas dari *bug*). Selain itu perangkat lunak tersebut harus terdokumentasi dengan baik. *Source code* juga harus mudah dibaca, terstruktur dan dirancang dengan baik dan bersifat modular.

2. Masalah-masalah RPL?

- Perangkat lunak telah diselesaikan dan diserahkan (*delivered*) tetapi tidak pernah digunakan (47%).
- Pemakai (*user*) sudah membayar untuk perangkat lunak tetapi tidak pernah jadi dan diserahkan (29,7%).
- Perangkat lunak digunakan setelah dilakukan modifikasi (3%).
- Perangkat lunak digunakan sebagaimana mestinya (2%).

Selain itu faktor pendukung kehadiran rekayasa perangkat lunak adalah :

- Ketidak mampuan untuk memprediksi waktu, usaha dan biaya pada pengembangan perangkat lunak.
- Kualitas perangkat lunak yang kurang baik.
- Perubahan perbandingan (rasio) harga perangkat keras dan perangkat lunak.
- Kemajuan teknologi perangkat keras.
- Kemajuan teknik perangkat lunak.
- Kebutuhan yang meningkat terhadap perangkat lunak.
- Kebutuhan akan perangkat lunak yang lebih besar dan kompleks.

3. PRODUK DAN PROSES

3.1. MENGEMBANGKAN PERAN PERANGKAT LUNAK

- 1980 an belum dikenal peranannya, bahkan menjadi bahan tertawaan. Hal ini disebabkan karena pada awal diciptakan, kecepatan perangkat lunak masih sangat lambat, sehingga masih “kalah” jika dibandingkan dengan kecepatan pengolahan angka dengan menggunakan kalkulator ataupun sempoa.

- Sekarang menjadi mesin yang mengendalikan pengambilan keputusan dan sangat erat kaitannya dengan berbagai bidang antara lain :
  - Transportasi
  - Medis
  - Telekomunikasi
  - Militer
  - Proses industri
  - Hiburan

Ada 2 peran yang dimiliki :

a. Sebagai Produk ➔ Perangkat Lunak mengantarkan potensi perhitungan yang dibangun oleh perangkat lunak computer. Tidak peduli apakah perangkat lunak ada di dalam sebuah telepon seluler atau beroperasi dalam sebuah mainframe computer, perangkat lunak ini merupakan transformer informasi yang memproduksi, mengatur, memperoleh, memodifikasi, menampilkan atau memancarkan informasi.
b. Sebagai kendaraan Produk itu sendiri -> Sebagai dasar untuk kontrol komputer seperti system operasi, jaringan untuk komunikasi informasi dan penciptaan serta control dari program-program lain.

Peran Perangkat Lunak computer mengalami perubahan penting dan dramatis selama paruh abad ke 20 seperti unjuk kerja perangkat keras, perubahan-perubahan besar dalam arsitektur computer, pertambahan yang pesat pada memori dan kapasitas penyimpanan, serta variasi pilihan input dan output yang luas

Era ketiga ditandai dengan munculnya :

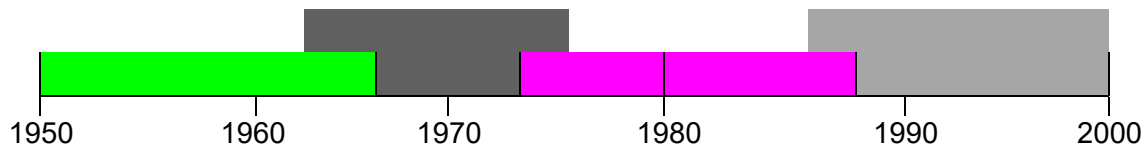
- System terdistribusi – multicomputer
- Masing-masing melakukan fungsi secara konkuren dan berkomunikasi satu dengan yang lain
- Menambah kompleksitas system berbasis computer.
- Permintaan akan pengembangan perangkat lunak dibidang jaringan local maupun global, jaringan komunikasi digital dan bandwidth yang tinggi meningkat
- Ditandai dengan kehadiran dan penyebaran pemakaian microprocessor

Era ke empat :

- User semakin jauh dari computer individual dan pemrograman
- Menuju kepada pengaruh kolektif dari computer dan perangkat lunak
- Mesin desktop yang kuat
- Sistem operasi lebih canggih
- Tersedianya jaringan local dan global
- Arsitektur perhitungan berubah cepat dari lingkungan mainframe terpusat ke lingkungan client/server yang desentralisasi

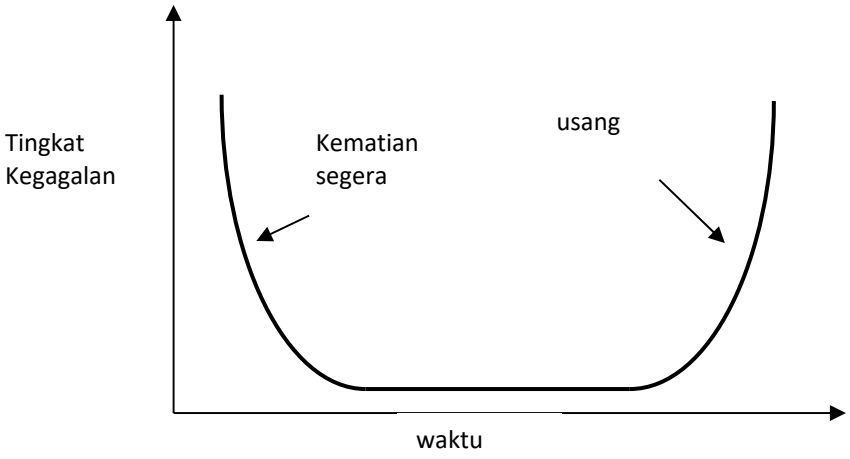
Rangkuman perkembangan yang terjadi, secara ringkas, terlihat dalam tabel sbb :

Tahun-tahun Awal	Era Kedua	Era Ketiga	Era Keempat
• Orientasi Batch • Distribusi terbatas • Perangkat Lunak Kustomisasi	• Multiuser • Real-Time • Database • Perangkat Lunak Produk	• Sistem Terdistribusi • Embedeed Intelligence • Perangkat keras biaya rendah	• Sistem Desktop bertenaga kuat • Teknologi berorientasi objek • Sistem Pakar • Jaringan syaraf tiruan • Komputasi paralel • Komputer jaringan

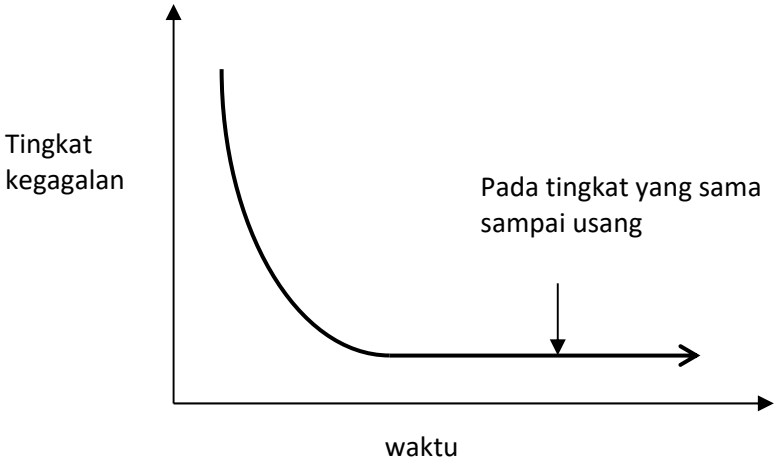


Gambar 1.1. Evolusi Perangkat Lunak

a. Embedded Software ➔ Dlm otomotif untuk kontrol bahan bakar, tampilan dashboard, sistem rem, dll.
b. Perangkat Lunak Personal Komputer ➔ Worksheet, Simple Database, dll.
c. Perangkat Lunak Kecerdasan Buatan ➔ Game, pembuktian teorema



Gambar 1.2. Kurva Kegagalan perangkat keras



Gambar 1.3. Kurva Kegagalan pada perangkat lunak

UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Siklus Hidup dan Proses Perangkat Lunak

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Informatika	03	87011	Tim Dosen

Abstrak

Modul ini menjelaskan mengenai Siklus Hidup Perangkat Lunak.

Kompetensi

Mampu memahami Siklus Hidup Perangkat Lunak. Mampu menjelaskan macam dari Siklus Hidup PL.

1. PANDANGAN UMUM TENTANG RPL

Rekayasa merupakan :

- Analisis
- Desain
- Konstruksi
- Verifikasi
- Manajemen Kesatuan Teknik (atau sosial)

Tanpa mempedulikan kesatuan yang dikembangkan, pertanyaan2 dibawah ini harus dimunculkan dan dijawab :

a. Masalah apa yang harus dipecahkan ?
b. Karakteristik kesatuan apa yg dipakai untuk menyelesaikan masalah tersebut ?
c. Bagaimana kesatuan (dan pemecahan) tersebut diadakan ?
d. Pendekatan apa yang dipakai untuk menemukan kesalahan-2 yang dibuat dalam desain dan konstruksi dari kesatuan tersebut ?
e. Bagaimana kesatuan tsb ditopang selama adaptasi dan selama perbaikan ?

"Dari IEEE (*IEEE Std 1016 – 1998 Recommended Practice for Software Design Description*) Dari standar IEEE 1016, ditekankan bahwa siklus hidup adalah segala sesuatu yang lebih berdasar kepada urutan waktu dibandingkan proses yang terjadi. Proses perangkat lunak ialah kegiatan yang tercakup dalam upaya memproduksi dan mengembangkan sistem perangkat lunak. Proses disajikan dalam model proses perangkat lunak. Kegiatannya secara umum terdiri dari penetapan spesifikasi, desain dan implementasi, validasi dan evolusi. Rekayasa kebutuhan (requirement engineering) adalah proses pengembangan spesifikasi perangkat lunak. Proses disain dan implementasi mengubah spesifikasi untuk sebuah sistem Bahwa siklus hidup perangkat lunak merupakan urutan hidup sebuah perangkat lunak berdasarkan perkembangan perangkat lunak yang ditentukan oleh pengembang perangkat lunak itu sendiri.

Proses perangkat lunak sangat rumit dan,seperti semua proses intelektual. Bergantung pada penilaian manusia. Karena dibutuhkan penilaian dan kreatifitas. keberhasilan usaha untuk mengotomasi proses perangkat lunak menjadi terbatas.

Tahapan proses perangkat lunak, yaitu :

- ✓ Rekayasa
- ✓ Analisa dan perancangan

- ✓ Pengembangan perangkat lunak
- ✓ Pasca produksi

Definisi Proses

Analisis dan defenisi persyaratan. Pelayanan, batasan, dan tujuan sistem ditentukan melalui konsultasi dengan user sistem. Perancangan sistem dan perangkat lunak. Proses perancangan sistem membagi persyaratan dalam sistem perangkat keras atau perangkat lunak. Kegiatan ini menentukan arsitektur sistem secara keseluruhan. Implementasi dan pengujian unit. Pada tahap ini, perancangan perangkat lunak direalisasikan sebagai serangkaian program atau unit program.

Integrasi dan pengujian sistem. Unit program atau program individual diintegrasikan dan diuji sebagai sistem yang lengkap untuk menjamin bahwa persyaratan sistem

Kesimpulannya:

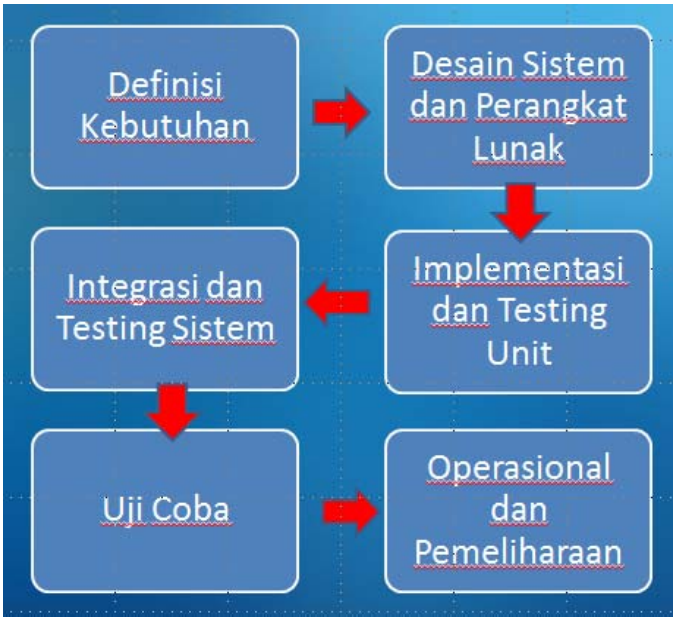
Bahwa sebuah proses perangkat lunak merupakan sekumpulan aktifitas maupun metode yang digunakan pengembang perangkat lunak.

1. Teori Dalam Siklus Hidup - Waterfall

Siklus hidup yang paling terkenal dalam dunia RPL adalah ***waterfall model*** . ***Waterfall model*** diciptakan pertama kali oleh William Royce pada tahun 1970 dan mulai terkenal karena logika fase (tahapan,tingkatan,masa).

Waterfall sendiri memiliki definisi bahwa sebuah proses hidup perangkat lunak memiliki proses yang linier dan sequensial.

Waterfall Life Cycle



2. Rapid Aplication Development (RAD)

   Merupakan metodelogi perancangan system dengan menggunakan frekwensi atau rentang waktu yang ditetapkan jaraknya sesuai dengan perangkat lunak yang dikembangkan

3. Prototype

   Metodelogi perancangan system yang evolusioner dimana dalam proses pengembangannya membutuhkan kemampuan tingkat mahir atau yang telah memahami permasalahan dalam kegiatan perancangan system dengan pengalaman dan keilmuan yang lebih baik.

4. Spiral

   Metodelogi ini pun merupakan kategori evolusioner yang membutuhkan kemampuan menganalisa permasalahan yang lebih komplek secara battom up.

UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Perencanaan Proyek Perangkat Lunak

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Informatika	04	87011	Tim Dosen

Abstrak

Modul ini menjelaskan mengenai Perencanaan Proyek Perangkat Lunak.

Kompetensi

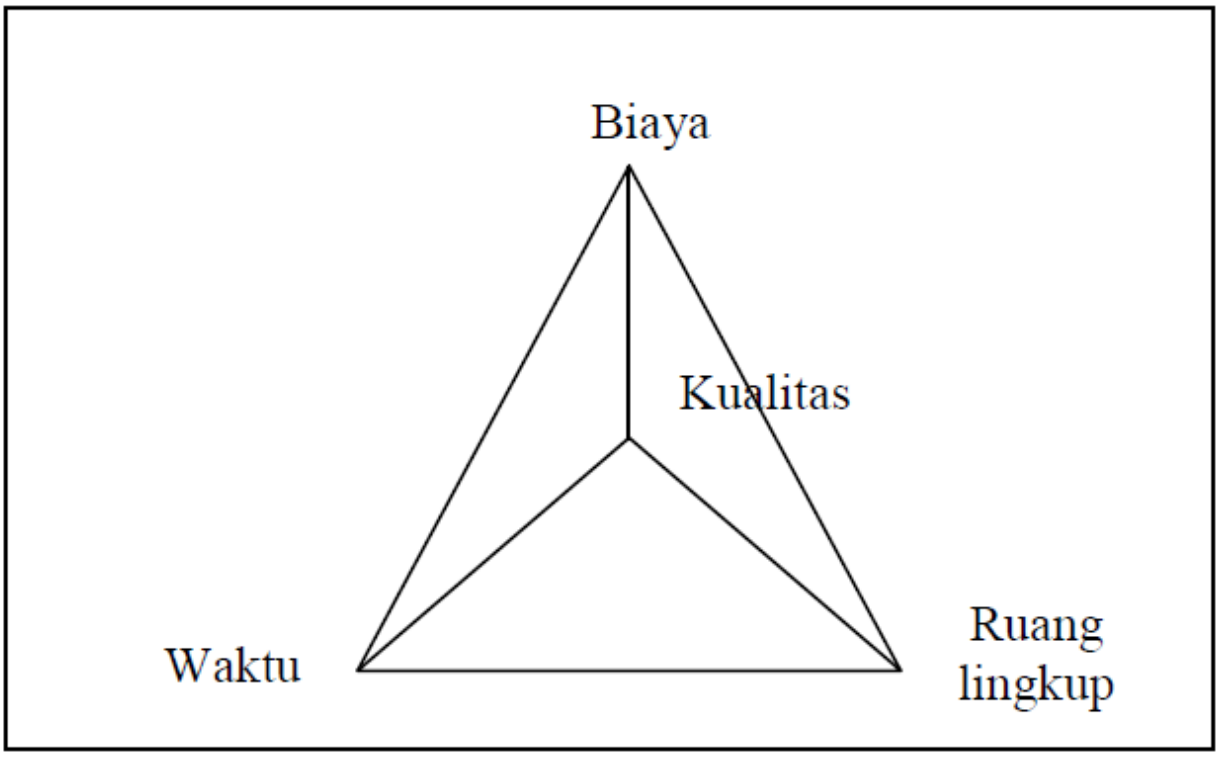
Mampu menentukan tujuan perencanaan proyek, ruang lingkup proyek PL, sumber daya yang dibutuhkan, dan estimasi proyek perangkat lunak.

“Tujuan perencanaan proyek perangkat lunak adalah untuk menyediakan sebuah kerangka kerja yang memungkinkan manajer membuat estimasi yang dapat dipertanggungjawabkan mengenai sumber daya, biaya dan jadwal”. (Roger S. Pressman, 2002)

“Proses manajemen proyek perangkat lunak dimulai dengan serangkaian aktivitas yang secara kolektif disebut project planing, yang pertama dari aktivitas ini adalah estimasi (perkiraan).” (Roger Pressman, 2002).

PENDAHULUAN PERENCANAAN PROYEK
(THE PRELIMINARY PROJECT PLAN / PPP)

- **Pendahuluan Perencanaan Proyek adalah langkah awal, sumber** daya, biaya dan jadwal yang dibutuhkan untuk menyelesaikan proyek. PPP adalah dokumen internal, tidak perlu ditunjukkan ke user, terutama user luar.



Gambar 4.1 Empat komponen proyek yang saling berpengaruh

Kunci berbagai rencana adalah memecah kegiatan yang diperlukan ke dalam sebuah bagian yang lebih kecil lagi. Rincian struktur kerja (WBS) diawali dengan menyusun komponen-komponen utama proyek. Hal ini merupakan Level 1 dari WBS (Level 0 adalah judul proyek).

Estimasi

Dalam aktifitas utama proyek yaitu perencanaan, dilakukan estimasi :

- Sumber daya manusia (ukuran orang/bulan)
- Jangka waktu kronologis (Ukuran waktu kalender)
- Biaya (Ukuran uang Rp)

Analisa Resiko

Analisis resiko sangat penting dalam manajemen proyek perangkat lunak. Beberapa hal yang harus diperhatikan berkaitan dengan resiko adalah :

- Masa yang akan datang : resiko apa yang mempengaruhi trend (kecenderungan) proyek perangkat lunak
- Perubahan : Bagaimana perkembangan dunia mempengaruhi keawetan dan kesuksesan perangkat lunak
- Pilihan : metode apa yang dipakai, berapa orang diperlukan, seberapa tinggi kualitas perangkat lunak dan sebagainya

Analisis resiko merupakan serangkaian langkah untuk menyiasati resiko, yaitu :

- Identifikasi Resiko

  Identifikasi resiko mencatat semua resiko sesuai dengan kategori (secara makro) sebagai berikut :

  1. Resiko proyek : masalah pembiayaan, penjadwalan, personil, sumber daya, pelanggan dan kebutuhan dikaitkan dengan akibatnya terhadap pelanggan.
  2. Resiko teknis : masalah desain, implementasi, antarmuka, verifikasi dan pemeliharaan.
  3. Resiko bisnis : termasuk di dalamnya adalah resiko pasar, resiko manajemen, dan resiko pembiayaan.

- Strategi manajemen resiko
- Putusan (Resolution) resiko
- Dan Pemantauan resiko

Penjadwalan

Langkah-langkah yang dilakukan dalam penjadwalan:

- Identifikasi sekumpulan tugas
- Pastikan keterkaitan antar tugas
- Estimasi usaha untuk tiap-tiap tugas
- Tentukan pekerja dan sumber-sumber lainnya
- Buat jaringan tugas
- Buat jadwal kerja berdasarkan waktu

Penelusuran dan Pengendalian

Penelusuran dan pengendalian dilakukan setelah ada penjadwalan yang pasti, yaitu memeriksa apakah tugas telah dilaksanakan sesuai dengan jadwal.

Satuan Ukuran Produktivitas dan Kualitas Perangkat Lunak

Pengukuran perangkat lunak dilakukan untuk :

- Indikasi kualitas produk
- Perkiraan produktivitas orang-orang yang menghasilkan produk
- Perkiraan manfaat dari penerapan metode dan tools
- Membentuk dasar dari estimasi
- Menegaskan (*Justify*) permintaan tools baru dan pelatihan

UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Analisis Kebutuhan Perangkat Lunak

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Informatika	05	87011	Tim Dosen

Abstrak

Modul ini menjelaskan mengenai Analisis Kebutuhan Perangkat Lunak.

Kompetensi

Mampu mengetahui tentang analisa kebutuhan PL, teknik komunikasi dan prinsip-prinsip analisis, pembuatan model prototype PL.

Pandangan umum tentang Rekayasa erangkat Lunak, merupakan :

- Analisis
- Desain
- Konstruksi
- Verifikasi
- Manajemen Kesatuan Teknik (atau sosial)

Tanpa mempedulikan kesatuan yang dikembangkan, pertanyaan2 dibawah ini harus dimunculkan dan dijawab :

a. Masalah apa yang harus dipecahkan ?
b. Karakteristik kesatuan apa yg dipakai untuk menyelesaikan masalah tersebut ?
c. Bagaimana kesatuan (dan pemecahan) tersebut diadakan ?
d. Pendekatan apa yang dipakai untuk menemukan kesalahan-2 yang dibuat dalam desain dan konstruksi dari kesatuan tersebut ?
e. Bagaimana kesatuan tsb ditopang selama adaptasi dan selama perbaikan ?

Usaha yang berhubungan dengan RPL dapat dikatagorikan menjadi 3 fase umum :

a. Fase Definisi (definition phase)
   Berfokus pada “apa” (what). Pengembang harus mampu mengidentifikasi :
   - Apa yang akan diproses
   - Fungsi dan unjuk kerja apa yang dibutuhkan
   - Tingkah laku system seperti apa yang diharapkan
   - Interface apa yang akan dibangun
   - Batasan desain apa yang ada
   - Kriteria validasi apa yang dibutuhkan untuk mendefinisikan system yang sukses
b. Fase Pengembangan (development phase)
   Berfokus pada “bagaimana” (how). Selama masa pengembangan :
   - Teknisi harus mendefinisikan bagaimana data dikonstruksikan

- Seorang manajer yang gagal melakukan komunikasi dengan pelanggan yang komprehensif di awal evolusi sebuah proyek, beresiko membangun pemecahan yang elegan untuk masalah-masalah yang salah.
- Akhirnya manajer yang tidak begitu memperhatikan proses, beresiko menyisipkan metode teknik yang cakap serta piranti ke dalam sebuah ruang hampa.

c. Fase Solusi (Decision Making)

Manajer melakukan proses pengambilan alternative terbaik untuk memutuskan proyek yang akan dilakukan.

2.1. PEOPLE

Faktor Manusia, menjadi sangat penting, sehingga Software Engineering Institute telah mengembangkan sebuah model kematangan kemampuan manajemen manusia (PM-CMM ➔ *People Management Capability Maturity Model*) yang fungsinya untuk mempertinggi kesiapan organisasi PL untuk mengerjakan aplikasi yang semakin kompleks dengan membantu menarik, menumbuhkan, memotivasi, menyebarkan dan memelihara bakat yang dibutuhkan untuk mengembangkan kemampuan per-kembangan perangkat lunak mereka.

PM-CMM membatasi area praktek bagi masyarakat PL dalam hal :

a. Rekrutmen
b. Seleksi
c. Unjuk Kerja
d. Pelatihan
e. Kompensasi
f. Perkembangan Karir
g. Desain Kerja dan Organisasi
h. Perkembangan Tim / Kultur

2.2. MASALAH (PROBLEM)

Sebelum proyek dimulai :

a. Objektivitas harus ditetapkan
b. Ruang Lingkup harus ditetapkan
c. Pemecahan Alternatif harus dipertimbangkan
d. Teknik harus didefinisikan
e. Batasan harus didefinisikan

Kelima hal di atas diperlukan untuk menentukan Estimasi biaya yang akurat

1. Apa yang disebut Kebutuhan (Requirement)

Menurut arti kamus, kebutuhan adalah sesuatu yang diminta, sesuatu yang dibutuhkan. Sedangkan menurut IEEE (*The Institute of Electrical and Electronics Engineers*) kebutuhan adalah :

- Kondisi atau kemampuan yang diperlukan pemakai untuk menyelesaikan suatu persoalan, atau untuk mencapai sebuah objek.
- Kondisi atau kemampuan yang harus dipenuhi oleh sistem, dalam arti memenuhi kontrak, standar, spesifikasi atau dokumen formal lain yang diinginkan.

Tahap kebutuhan akan perangkat lunak dimulai dengan :

a. Dikenalinya adanya sebuah permasalahan yang membutuhkan sebuah penyelesaian. Identifikasi sebuah permasalahan mungkin dapat dilakukan dengan berorientasi pada aplikasi, berorientasi pada bisnis, atau berorientasi pada kenaikan produktivitas (*product improvement oriented*).

b. Munculnya ide untuk membuat sebuah perangkat lunak baru (sebagai sebuah kemajuan).

Ada dua jenis kebutuhan :

1. Behavioral
   - Apa yang dilakukan oleh sistem (input dan output dari dan ke sistem).
   - Hubungan informasi antara input dan output sehingga menghasilkan sebuah fungsi transformasi.

2. Non-behavioral

   Mendefinisikan atribut sistem yang terkait untuk membentuk pekerjaan tersebut. Termasuk deskripsi lengkap tentang efisiensi, keamanan (*security*), *rehability maintenability* (bagaimana perawatan untuk sistem), dan *portability* (bisa dipindahkan dari satu perangkat keras ke perangkat keras lainnya).

- Jika dapat dideteksi, dilakukan perbaikan pada setiap tahap proses.
- Jika tidak dapat dideteksi, kesalahan baru kelihatan setelah produk selesai dibuat.

2. Tahap Analisis Kebutuhan Perangkat Lunak

Tahap pekerjaan analisis kebutuhan perangkat lunak pada dasarnya terdiri dari urutan aktivitas :

1. Menentukan kebutuhan (*requirement*)
   Lebih banyak berhubungan dengan pemakai. Hasil belum terstruktur.

   - Data atau informasi apa yang akan diproses
   - Fungsi apa yang diinginkan
   - Kelakuan sistem apa yang diharapkan
   - Antarmuka apa yang tersedia (*user interfaces, hardware interfaces, software interface,* dan *communications interfaces*)

2. Sintesis
   Mengubah kebutuhan yang belum terstruktur menjadi model atau gambar dengan memanfaatkan teknik dan metodeanalisis tertentu.

3. Membuat dokumen *Software Requirements Spesification* (SRS).
   Sudah merupakan analisis yang lebih rinci, sebagai tahap awal perancangan.

3. Metode Analisis

Metode atau teknik untuk melakukan analisis kebutuhan perangkat lunak dikelompokkan berdasarkan pendekatan yang diambil pada saat melakukan aktivitas tersebut.

Berorientasi Aliran Data (*Data Flow Oriented* atau *Functional Oriented*) dengan teknik top down atau bottom up.

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Spesifikasi Kebutuhan Perangkat Lunak

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Informatika	06	87011	Tim Dosen

Abstrak

Modul ini menjelaskan mengenai Spesifikasi Kebutuhan Perangkat Lunak.

Kompetensi

Mampu memahami SRS.

Pada proses kegiatan analisi system perencanaan Proyek (Project Planning) merupakan awal dari serangkaian aktivitas secara kolektif dari sebuah proses Manajemen Proyek Perangkat Lunak

- Yang pertama dalam proses perencanaan ini adalah Estimation (perkiraan)
- Bertujuan untuk melihat kondisi umum ke depan
- Siap menerima kondisi yang berada dalam tingkat ketidak pastian.
- Estimasi menjadi dasar bagian semua aktivitas perencanaan proyek yang lain, dan
- Perencanaan proyek memberikan sebuah peta jalan bagi suksesnya Rekayasa Perangkat Lunak

Untuk mendapatkan data kita perlu melakukan beberapa kegiatan (Perancangan Sistem Informasi, Tata Sutabri, 2005);

1. Observasi
2. Interview
3. Kuisioner
4. Korespondensi

OBSERVASI PADA ESTIMASI

Harapan Ekesekutif akan persyaratan seorang calon manajer proyek :

- Memiliki kemampuan untuk mengetahui ketidakberesan apa yang akan terjadi sebelum hal itu benar-benar terjadi
- Memiliki keberanian untuk memperkirakan kapan mendung akan datang

1. Yang Harus Dihindari

Hindari hal-hal berikut saat pembentukan SRS

1. Over specification (penjelasan berlebih dan berulang-ulang sehingga menjadi tidak jelas)
2. Tindakan unconcistency
3. Ambiguity dalam kata atau kalimat
4. Menuliskan “mimpi-mimpi” , yaitu hal-hal yang tidak bisa dilakukan

2. Aspek Dalam SRS

Dalam Suatu SRS ada 2 aspek yang harus bisa dilihat :

1. Fungsi

   Menjelaskan fungsi dari perangkat lunak (digunakan untuk apa keperluan apa), sifat lunak dan datanya.

2. Non-Fungsi

   a. Dependability
      i. reliability
      ii. maintainbility
      iii. security
      iv. integrity
   b. Ergonomic
   c. Performance
   d. Contraint

3. Atribut Suatu SRS

1. Benar (correct)

   Jika salah (incorrect), artinya spesifikasi yang ditulis adalah bukan yang diinginkan.

2. Tepat (precise)

   Berpengaruh pada hasil perancangan dan pembuatan software requirements design (SRD).

4. Yang Terlibat Dalam Pembuatan SRS

Ada 9 macam orang yang terlibat dalam pembuatan SRS :

1. Pemakai (user)

   Yang mengoperasikan / menggunakan produk final dari perangkat lunak yang dibuat.

2. Client

   Orang atau perusahaan yang mau membuat sistem (yang menentukan).

3. Sistem analyst (sistem engineer)

Yang biasa melakukan kontak teknik pertama dengan client. Bertugas menganalisis persoalan, menerima requirement dan menulis requirement.

4. Software engineer

   Yang bekerja setelah kebutuhan perangkat lunak dibuat (bekerja sama dengan sistem engineer berdasarkan SRS)

5. Programmer

   Menerima spesifikasi perancangan perangkat lunak, membuat kode dalam bentuk modul, menguji dan memeriksa (tes) modul.

6. Test integration group

   Kumpulan orang yang melakukan tes dan mengintegrasi modul.

7. Maintenance group

   Memantau dan merawat performansi sistem perangkat lunak yang dibuat selama pelaksanaan dan pada saat modifikasi muncul (80% dari pekerjaan).

8. Technical Support

   Orang-orang yang mengelola (manage) pengembang perangkat lunak, termasuk konsultan atau orang yang mempunyai kepandaian lebih tinggi.

9. Staff dan Clerical Work

   Bertugas mengetik, memasukkan data dan membuat dokumen.

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Penjadwalan dan Penelusuran Proyek PL

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	07	Kode MK?	Tim Dosen

Abstract

Menjelaskan mengenai tahapan penjadwalan membangun perangkat lunak

Kompetensi

Mahasiswa memahami proses menyusun jadwal membangun perangkat lunak

PENJADUALAN DAN PENELUSURAN PROYEK
(*PROJECT SCHEDULING and TRACKING*)

MENGAPA SEBUAH PROYEK BISA TERLAMBAT ?

Meskipun ada banyak alasan mengapa perangkat lunak dikirim terlambat, sebagian besar dapat ditelusuri sampai pada satu akar penyebab atau lebih berikut ini :

- Suatu batas waktu yang tidak realistis yang dibangun oleh orang diluar kelompok rekayasa perangkat lunak yang memaksa manajer dan pelaksana dalam kelompok itu
- Perubahan kebutuhan pelanggan yang tidak dicerminkan dalam perubahan jadual
- Memandang rendah jumlah usaha dan atau jumlah sumber-sumber daya yang akan dibutuhkan untuk melakukan pekerjaan itu.
- Resiko yang dapat diramalkan dan atau tidak dapat diramalkan yang tidak dipertim-bangkan pada saat proyek dimulai
- Kesulitan teknis yang tidak dapat dilihat sebelumnya
- Kesulitan manusia yang tidak dapat dilihat sebelumnya
- Kesalahan komunikasi diantara staf proyek yang mengakibatkan penundaan
- Kegagalan manajer proyek untuk mengetahui bahwa proyek ketinggalan dari jadual yang ada dan kurangnya tindakan untuk memecahkan masalah tersebut.

PRINSIP PENJADUALAN

Sejumlah prinsip dasar yang bisa menuntun penjadualan proyek perangkat lunak adalah :

- PEMBAGIAN : proyek harus dibagi-bagi kedalam sejumlah tugas dan aktivitas yang dapat dikendalikan
- SALING KETERGANTUNGAN : Saling ketergantungan dari setiap tugas dan aktivitas yang dibagi-bagi harus ditentukan
- ALOKASI WAKTU : Setiap tugas yang akan dijadualkan harus dialokasikan dalam sejumlah satuan kerja
- VALIDASI KERJA : Setiap proyek memiliki sejumlah staf tertentu. Pada saat alokasi waktu dilakukan, manajer proyek harus memastikan bahwa tidak akan ada kelebihan alokasi jumlah manusia pada suatu saat tertentu.

- BATASAN TANGGUNG JAWAB : Setiap tugas yang dijadualkan harus ditugaskan kepada satu anggota tim tertentu.
- BATASAN KELUARAN : Setiap tugas yang dijadualkan harus memiliki keluaran tertentu. Untuk proyek perangkat lunak, keluaran biasanya dalam bentuk hasil kerja (seperti rancangan modul) atau sebagian dari hasil kerja
- KEJADIAN PENTING YANG DITENTUKAN : Setiap tugas atau kelompok tugas harus dihubungkan dengan kejadian penting proyek

HUBUNGAN ANTARA MANUSIA DAN (USAHA) KERJA

Dalam proyek pengembangan perangkat lunak berskala kecil, seseorang dapat menganalisis kebutuhan, melakukan perancangan, generalisasi kode dan melakukan pengujian. Ketika ukuran proyek bertambah, jumlah manusia yang terlibat menjadi lebih banyak. Tetapi harus diingat bahwa jumlah orang yang terlibat dalam sebuah proyek dan produktivitas TIDAK LINIER.

Contoh (1) :

- Misal ada empat orang software engineer, yang mana masing-masingnya memiliki kemampuan menghasilkan 5000 baris program per tahun
- Asumsikan bahwa setiap komunikasi antar mereka akan mengurangi produktivitas sebanyak 250 baris program per tahun
- Oleh karena itu :

  Jumlah bagian komunikasi adalah = 4! / (2!2!) = 24 / (2 * 2) = 24 / 4 = 6

  Dengan demikian, produktivitas tim akan menjadi :

  = (4 * 5000) – (250 * 6)

  = 20000 – 1500 = 18500 LOC / tahun

  Atau setara dengan 7½ % lebih kecil dari pada yang kita harapkan

Example (2)

With 2 months remaining, 2 additional people are added

Therefore, the number of communication path: 6!/(2!4!) = 15

productivity of 2 new staffs = 2 * (5000/12 month) * 2 month = 1680 LOC

So,

team productivity: 20,000 + 1680 – 250*15 ≈ less than 18,500 LOC/year

TIPE-TIPE PROYEK

Meskipun mudah untuk mengembangkan system klasifikasi yang luas, kebanyakan organisasi perangkat lunak menemui proyek dengan tipe-tipe sebagai berikut :

I. Concept Development Project : diinisiasi untuk mencari konsep bisnis yang baru atau aplikasi beberapa teknologi baru
II. New Application Development Project : dilakukan sebagai konsekuensi permintaan pelanggan yang khusus
III. Application Enhancement Project : terjadi ketika perangkat lunak yang ada mengalami modifikasi utama pada fungsi, kinerja atau interface yang dapat diamati oleh pemakai akhir
IV. Application Maintenance Project : dilakukan untuk membetulkan, menyesuaikan atau memperluas perangkat lunak yang ada dengan cara yang tidak begitu jelas bagi pemakai akhir.
V. Reengineering Project : proyek yang dikerjakan dengan maksud membangun system (warisan) yang ada secara keseluruhan atau sebagian.

MENENTUKAN KRITERIA ADAPTASI :

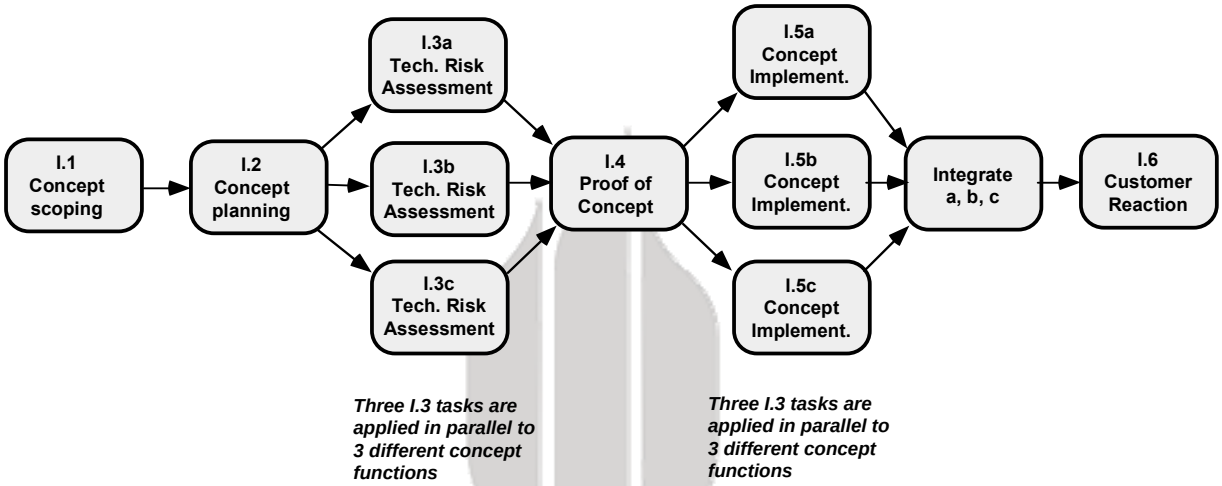
Criteria adaptasi digunakan untuk menentukan derajat kekakuan yang direkomendasikan dimana proses perangkat lunak akan diaplikasikan. Sebelas criteria adaptasi didefinisikan untuk proyek perangkat lunak yaitu :

- Size of the project
- Number of potential users
- Mission criticality
- Application longevity
- Stability of requirements
- Ease of customer/developer communication
- Maturity of applicable technology
- Performance constraints
- Embedded and non-embedded characteristics
- Project staff
- Reengineering factors

JARINGAN KERJA

- A graphic representation of the task flow for the project
- Sometimes used as a mechanism for inputting task sequence and dependencies to an automated tools
- The concurrent nature of the tasks may lead to critical path, that is, tasks that must be completed on schedule

MENDEFINISIKAN JARINGAN KERJA



PENJADUALAN

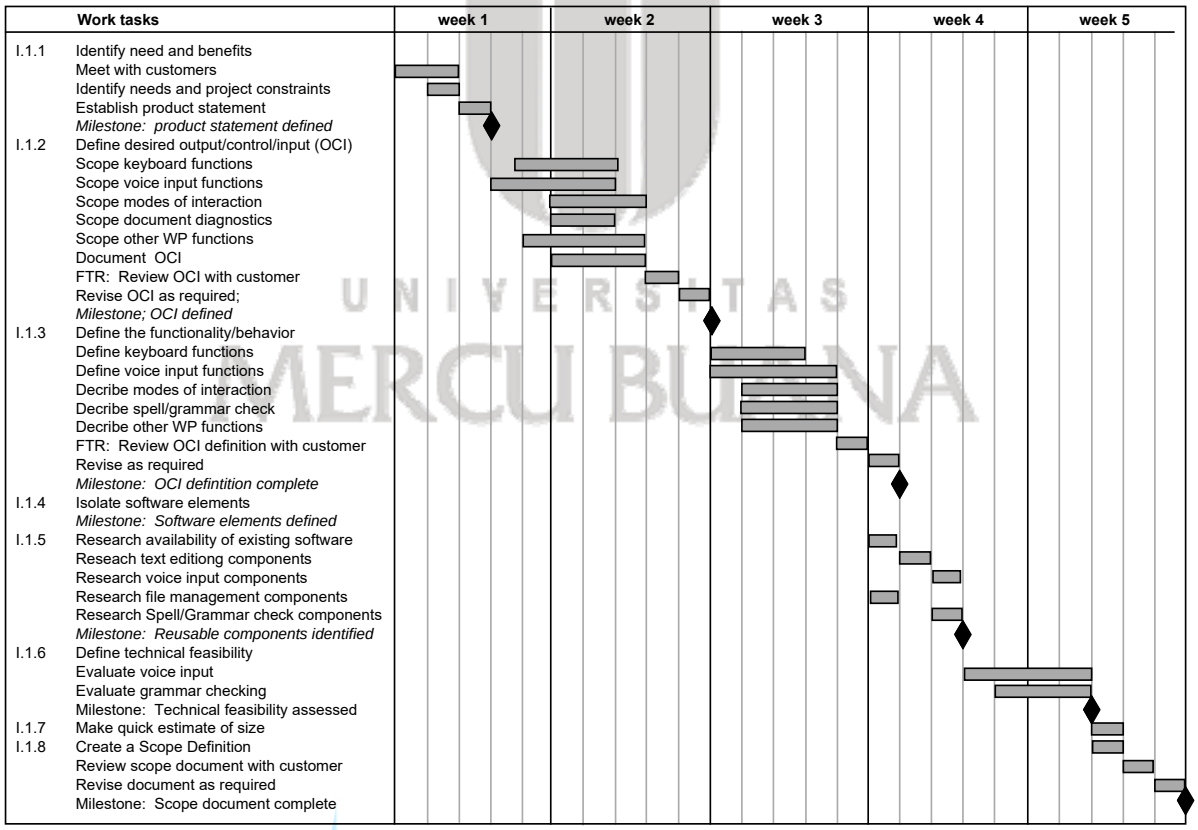
- Metode PERT (Program Evaluation and Review Technique) & CPM (Critical Path Method)
- PERT & CPM digunakan untuk :
    1. Determine critical path
    2. Establish most likely time estimates for individual task
    3. Calculate “boundary times” that define a time “window” for particular task
- Task, terkadang disebut Work Breakdown Structure (WBS)

(akan dibahas lengkap pada mata kuliah manajemen proyek perangkat lunak)

ALOKASI KERJA (USAHA)

- Aktivitas “front end” (40 – 50% usaha)
  - Komunikasi pelanggan
  - Analisa
  - Desain
  - Kajian ulang dan modifikasi
- Aktivitas konstruksi (15 – 20% usaha)
  - Coding dan pembuatan kode
- Uji coba dan instalasi (30 – 40% usaha)
  - unit, integrasi
  - white-box, black box
  - Regresi

Menggunakan alat bantu otomatis untuk memperoleh bagan timeline



Gambar …Time Schedule

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

JAMINAN KUALITAS PERANGKAT LUNAK

(*SOFTWARE QUALITY ASSURANCE*)

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	08		Tim Dosen

Abstract

Diisidengan abstract

Kompetensi

Mahasiswa mampu memahami quality anserance yang harus diwujudkan untuk meningkatkan kualitas

PERTEMUAN-8

JAMINAN KUALITAS PERANGKAT LUNAK
(*SOFTWARE QUALITY ASSURANCE*)

Pendekatan rekayasa perangkat lunak yang digambarkan bekerja kearah tujuan tunggal : menghasilkan perangkat lunak berkualitas tinggi.

Apa sebenarnya kualitas perangkat lunak itu ?

Menurut Philip Crosby, kualitas perangkat lunak : …… masalah manajemen kualitas bukanlah suatu hal yang tidak diketahui. Masalahnya adalah apa yang mereka pikir mereka tahu……

Banyak pengembang perangkat lunak terus percaya bahwa kualitas perangkat lunak merupakan sesuatu yang mulai anda khawatirkan setelah kode-kode dilahirkan. Tidak ada yang dapat melebihi suatu kebenaran.

Jaminan kualitas perangkat lunak (Software Quality Assurance [SQA]) adalah aktivitas pelindung yang diaplikasikan pada seluruh proses perangkat lunak.

SQA meliputi :

1. Pendekatan manajemen kualitas
2. Teknologi rekayasa perangkat lunak yang efektif
3. Kajian teknik formal yang diaplikasikan pada keseluruhan proses perangkat lunak
4. Strategi pengujian multitiered (deret bertingkat)
5. Kontrol dokumentasi perangkat lunak dan perubahan yang dibuat untuknya
6. Prosedur untuk menjamin kesesuaian dengan standar pengembangan perangkat lunak
7. Mekanisme pengukuran dan pelaporan

PENDAHULUAN

Bila ada salju yang jatuh, maka tidak aka nada dua butir salju yang sama. Adalah sangat sulit untuk membayangkan bahwa butiran-butiran salju itu berbeda bentuk sama satu lain. Belum lagi bahwa setiap butir memiliki struktur yang unik. Untuk mengamati perbedaan diantara dua butir salju , kita harus mengamati butiran-butiran itu secara lebih dekat. Mungkin kita dapat melakukannya dengan menggunakan kaca pembesar. Kenyataannya, semakin dekat kita melihat, semakin besar perbedaan yang dapat kita amati.

Fenomena tersebut, variasi antar sampel, sangat sesuai dengan produk manusia seperti halnya ciptaan alam. Sebagai contoh, bila ada dua papan sirkuit yang identik diamati secara cukup dekat, kita dapat mengamati bahwa jalur kecil tembaga pada papan itu sangat berbeda secara geometri, penempatan dan ketebalannya, lokasi dan diameternya sangat bervariasi.

Semua bagian yang direkayasa dan dihasilkan menunjukkan keragaman. Keragaman antar sampel tidak akan nyata tanpa bantuan peralatan yang teliti untuk mengukurnya secara geometri , karakter listrik atau atribut bagian. Tetapi dengan peralatan yang sangat sensitive, kita dapat sampai pada kesimpulan bahwa tidak ada dua sampel yang benar-benar sama.

Apakah prinsip ini sama untuk perangkat lunak dan masalah-masalah fisik ?. Bayangkan lah sebuah program yang pada beberapa titik selama eksekusinya, penyeleksian rekaman dengan urutan naik berdasarkan medan kunci yang ditentukan. Sifat rekaman rekaman tidak dipentingkan. Sifat itu dapat berupa rekaman karyawan, database pelanggan, koordinat peta untuk system control penerbangan real time atau yang lainnnya.

Pemrogram yang menciptakan rutin sorting (atau memilihnya dari sebuah pustaka komponen reuseable) memilih menggunakan quicksort untuk memecahkan masalah yang ada. Dapatkah seorang pengamat produk akhir membedakan perangkat lunak dari sebuah produk yang sangat identik yang menggunakan, misalnya buble sort. Mungkin dapat, tetapi mungkin kita membutuhkan lebih banyak informasi dan peralatan yang sangat sensitive untuk membedakan antara kedua system tersebut.

Kontrol variasi merupakan inti dari control kualitas. Pemanufaktur ingin meminimalkan variasi antara produk yang mereka buat, bahkan dalam hal yang kecil sekalipun, seperti menduplikasi disket. Kita ingin meminimalkan variasi diantara pasangan disket yang identik. Hal ini tentu bukan masalah, menduplikasi disket merupakan operasi produksi yang sangat mudah, dan kita dapat menjamin bahwa duplikat perangkat lunak yang tepat selalu dapat dibuat.

Mesin duplikasi disket dapat benar-benar menggunakan dan berjalan dalam batas toleransi, sehingga pada proses yang sederhana sekalipun, seperti duplikas, masalah sehubungan dengan variasi diantara sampel dapat terjadi.

Bagaimana organisasi pengembang perangkat lunak melakukan control variasi ?. Dari satu proyek ke proyek lain, kita perlu meminimalkan perbedaan diantara sumber-sumber daya actual yang digunakan, termasuk penataan staf, perlengkapan dan waktu kalender.

Secara umum kita perlu memastikan bahwa program pengujian kita sudah mencakup persentase perangkat lunak yang diketahui dari satu peluncuran ke peluncuran lain.

KUALITAS

American Heritage Dictionary mendefinisikan kata kualitas sebagai “sebuah karakteristik atau atribut dari sesuatu”. Sebagai atribut dari sesuatu, kualitas mengacu kepada Karakteristik yang dapat diukur- sesuatu yang dapat dibandingkan dengan standar yang sudah diketahui, seperti panjang, warna, sifat kelistrikan, kelunakan dan sebagainya. Tetapi perangkat lunak, yang sebagian besar merupakan entitas intelektual, lebih menantang untuk di karakterisasi dari pada objek fisik.

KONSEP KUALITAS

- general objective: reduce the “variation between samples” ... but how does this apply to software?
- quality control: a series of inspections, reviews, tests
- quality assurance: analysis, auditing and reporting activities
- cost of quality
    - appraisal costs
    - failure costs
    - external failure costs

ADA 2 JENIS KUALITAS PERANGKAT LUNAK

1. Quality of design:
   the characteristics that the designers specify for an item.
   Eg, requirements, specifications, and design
2. Quality of conformance:
   the degree to which the design specifications are followed during manufacturing
   Eg, implementation

PENGAWASAN KUALITAS (QUALITY CONTROL)

- Involves the series of inspections, reviews, and tests used throughout the software process to ensure each work product meets the requirements
- Includes a feedback loop to the process
- May be manual, automated, or both

- Measurement + feedback

JAMINAN KUALITAS (QUALITY ASSURANCE)

- Consists of the auditing and reporting functions of management
- Tujuan :
  - to provide management with the data necessary to be informed about product quality,
  - thereby gaining insight and confidence that product quality is meeting its goal

BIAYA KUALITAS (QUALITY COSTS)

1. Prevention costs:
   - quality planning, formal technical reviews, test equipment, training
2. Appraisal costs
   - include activities to gain insight into product condition the "first time through" each process: in-process & inter-process inspection, equipment calibration & maintenance, testing
3. Failure costs:
   - Internal Failure costs: rework, repair, failure mode analysis
   - External Failure costs: complaint resolution, product return & replacement, help line support, warranty work

TUGAS MAHASISWA ;

CARI DAN KUMPULKAN KLIPING DARI KORAN, MAJALAH ATAUPUN INTERNET MASING-MASING 5 (LIMA) CONTOH UNTUK SETIAP BIAYA KUALITAS TSB DI ATAS.

KUALITAS PERANGKAT LUNAK

Conformance to

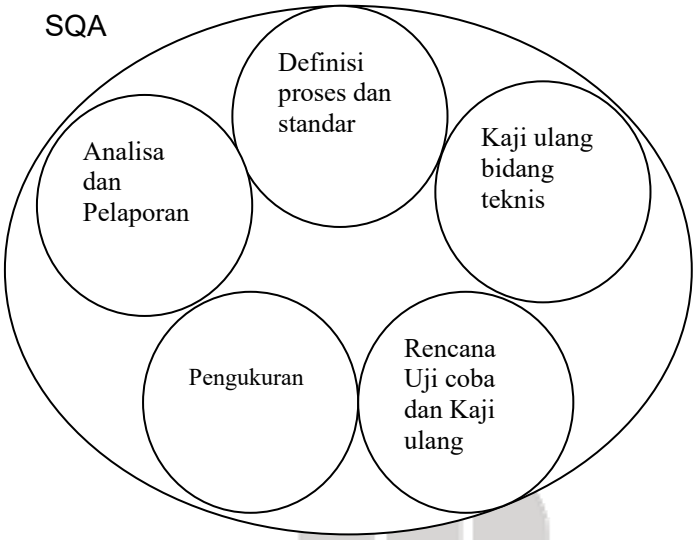
- explicitly stated functional and performance requirements,
- explicitly documented development standards, and
- implicit characteristics that are expected of all professionally developed software

Tiga poin penting dari definisi

1. Software requirements are the foundation from which quality is measured

2. Specified standards define a set of development criteria that guide the engineering
3. A set of implicit requirements often goes unmentioned (eg. Ease of use, maintainability)

JAMINAN KUALITAS PERANGKAT LUNAK



AKTIVITAS SQA

- Software engineers address quality by
  - applying solid technical methods & measures,
  - conducting formal technical reviews, and
  - performing well-planned testing
- SQA group assists the software team in achieving a high quality product by addressing
  - quality assurance planning,
  - oversight,
  - record keeping,
  - analysis, &
  - reporting

KAJI ULANG DAN INSPEKSI

Jerry Weinberg menyebutkan bahwa :

THERE IS NO PARTICULAR REASON WHY YOUR FRIEND AND COLLEAGUE CANNOT ALSO BE YOUR STERNEST CRITIC.

(Tidak ada alasan khusus mengapa teman dan kolega anda tidak dapat sebagai pengkritik yang tajam (terhadap anda))

APA YANG DIMAKSUD DENGAN KAJI ULANG (REVIEW)

- a meeting conducted by technical people for technical people
- a technical assessment of a work product created during the software engineering process
- a software quality assurance mechanism
- a training ground

BATASAN

- ✓ Between 3 and 5 people
- ✓ Advance preparation should occur but should require no more than 2 hours of work for each person
- ✓ Duration of review meeting should be less than 2 hours

YANG TIDAK TERMASUK DALAM POIN KAJIAN

- Rangkuman anggaran proyek
- Penilaian jadual
- Laporan perkembangan proyek secara menyeluruh
- Intrik-intrik politik atau mekanisme untuk balas dendam

MEMIMPIN KAJIAN

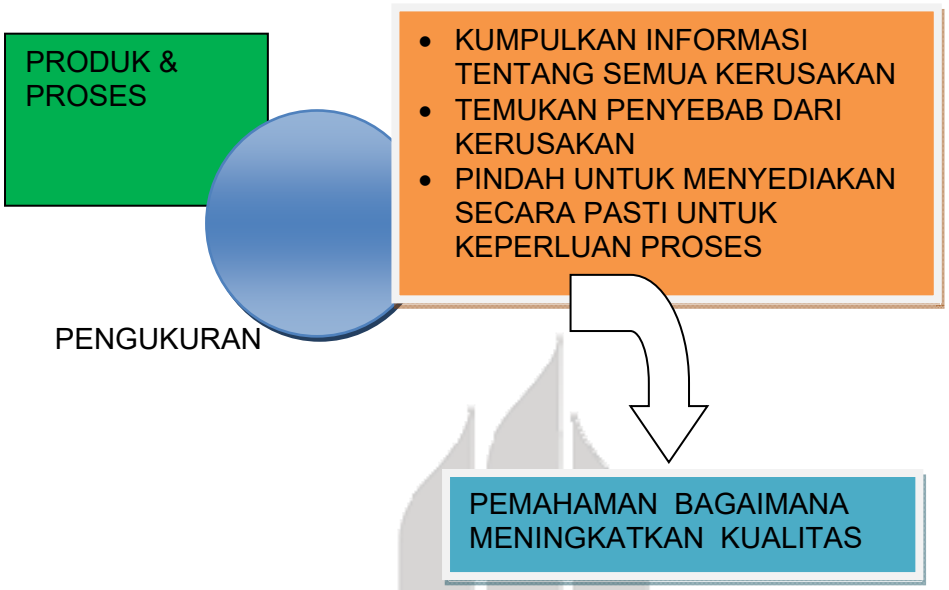
1. Be prepare – evaluate product before the review
2. Review the product, not the producer
3. Keep your tone mild, ask questions instead of making accusations
4. Stick to the review agenda
5. Raise issues, don't resolve them
6. Avoid discussion of style, stick to technical correctness
7. Schedule reviews as project tasks
8. Record and report all review results

PANDUAN REVIEW

1. Review the product, not the producer
2. Set an agenda and maintain it
3. Limit debate and rebuttal
4. Enunciate the problem areas, but don't attempt to solve every problem noted
5. Take written notes
6. Limit the number of participants and insists upon advance preparation
7. Develop a checklist for each product that is likely to be reviewed

8. Allocate resources and schedule time for FTRs
9. Conduct meaningful training for all reviewers
10. Review your early review

STATISTIK SQA



KEANDALAN PERANGKAT LUNAK (SOFTWARE RELIABILITY)

- The probability of failure-free operation of a computer program in a specified environment for a specified time
- Measures:
  MTBF = MTTF + MTTR
- Availability = (MTTF/(MTTF+MTTR)) * 100%
- Software safety is
  - SQA activity that focuses on the identification & assessment of potential hazards that may affect software negatively and cause system to fail

ISO 9000

- Describes quality assurance elements in generic terms that can be applied to any business regardless of the products or services offered
- Elements of quality assurance system:
  - organizational structure,
  - procedures,

- processes, and
- resources to implement quality planning,
- quality control,
- quality assurance, and
- quality improvement

➢ ISO 9001 – for software engineering

PERENCANAAN SQA

- Provides road map for instituting software quality assurance
- The plan serves as template for SQA activities
- Contents:
  - ✓ initial section – purpose and scope
  - ✓ management section – organizational structure
  - ✓ documentation section – project documents, models, technical documents, user documents
  - ✓ test section
  - ✓ tools & methods, SCM, contract, records maintenance, training, risk

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

MANAJEMEN KONFIGURASI PERANGKAT LUNAK

(*SOFTWARE CONFIGURATION MANAGEMENT = SCM*)

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	09		Tim Dosen

Abstract

Diisidengan abstract

Kompetensi

Menjelaskan bahwa membangun perangkat lunak harus mengetahui kebutuhan-kebutuhan informasi yang di hasilkan dari P/L tersebut

PERTEMUAN-9

MANAJEMEN KONFIGURASI PERANGKAT LUNAK

(*SOFTWARE CONFIGURATION MANAGEMENT = SCM*)

APA ITU SCM ?

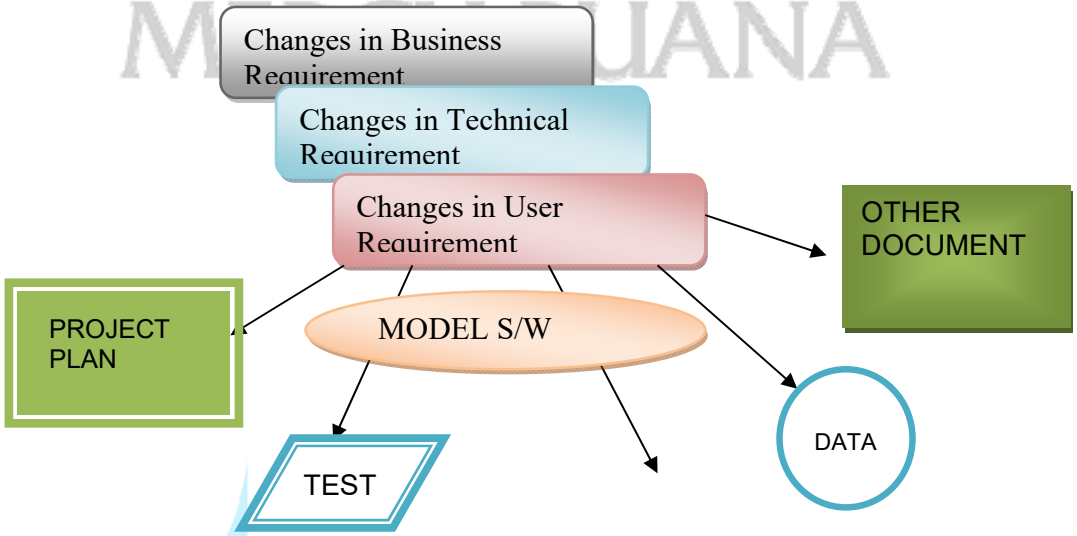
A set of activities designed to control change by

- identifying the work products that are likely to change,
- establishing relationships among them,
- defining mechanisms for managing different versions of these work products,
- Controlling the changes imposed, and
- Auditing & reporting the changes made

HUKUM PERTAMA

> No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle……
>
> Bersoff

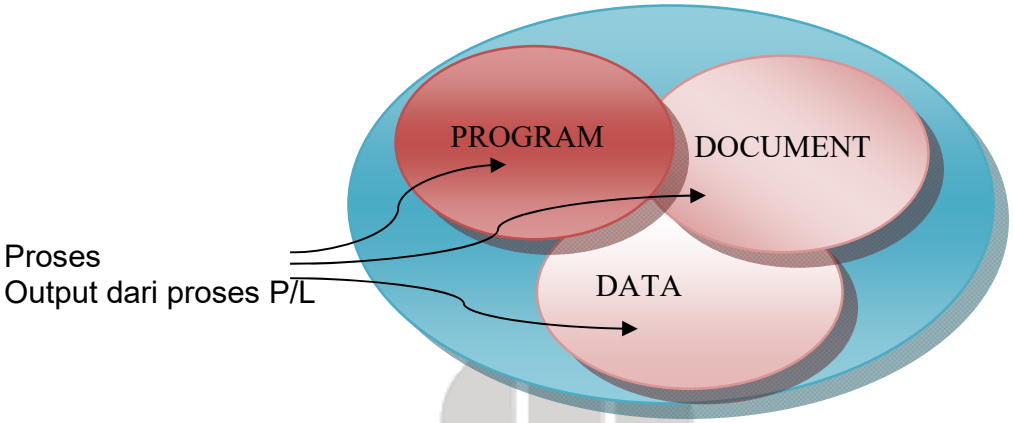
HAL APA SAJA YANG BERUBAH ?



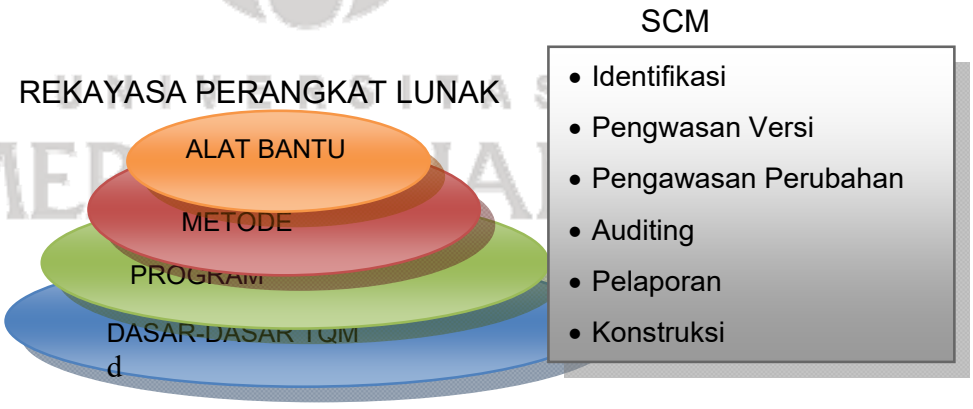
CODE

KONFIGURASI PERANGKAT LUNAK

The items that comprises all information produced as part of the software process



PERUBAHAN DAN SCM



BASE LINES

Baseline adalah sebuah konsep manajemen konfigurasi perangkat lunak yang membantu kita mengontrol perubahan tanpa harus secara serius menggangu perubahan yang dapat dibenarkan, mendefinisikan baseline sebagai :

“Suatu spesifikasi atau produk yang telah dikaji secara formal dan disetujui, yang kemudian berfungsi sebagai dasar bagi pengembangan lebih jauh, serta dapat diubah hanya melalui prosedur control perubahan formal”

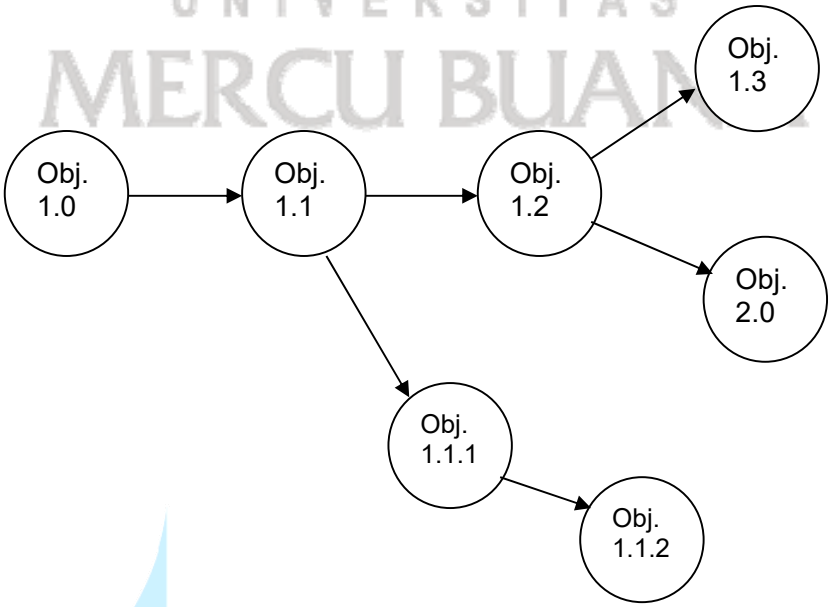
(“*A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures*”)

- Contoh ; System Specification, SW requirements, design specification, source code, test plan, operational system

SCI (Software Configuration Items)

- Information that is created as part of the software engineering process
  - ✓ eg,. A single section of a large specification, or one test case in a large suite of tests
- It is a document, an entire suite of test cases, or a named program component (eg. C++ function)
- Specific versions of editors, compilers, and other CASE tools may be “frozen” as part of the software configuration

GRAFIK EVOLUSI



PENGAWASAN PERUBAHAN (CHANGE CONTROL)

Pengawasan perubahan mengkombinasikan prosedur manusia dan piranti otomatis untuk menyediakan sebuah mekanisme untuk mengontrol perubahan.

ADAPUN LANGKAH-LANGKAH MEKANISME PERUBAHAN SEBAGAI BERIKUT :

ELEMEN DARI PENGAWASAN PERUBAHAN

- **Access Control:** governs which software engineers have the authority to access and modify a particular configuration object
- **Synchronization control:** helps to ensure that parallel changes, performed by two different people, don't overwrite one another

AUDIT (untuk konfigurasi P/L)

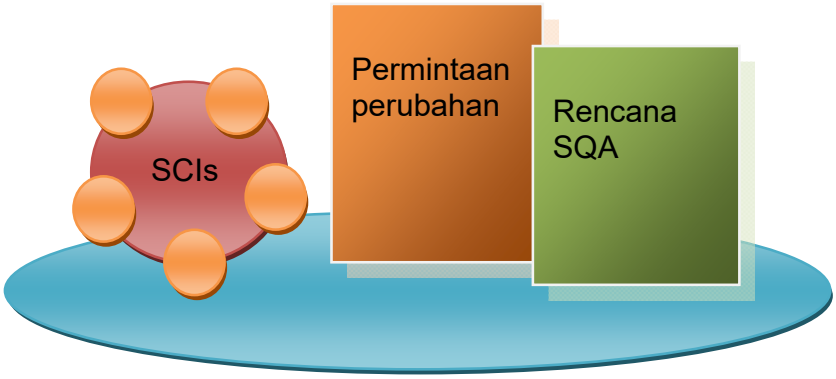
Audit konfigurasi perangkat Lunak melengkapi kajian teknis formal dengan menilai suatu objek konfigurasi untuk karakteristik yangsecara umum tidak dipertimbangkan selama kajian. Audit harus menanyakan dan menjawab pertanyaan-pertanyaan berikut :

1. Sudahkah perubahan yang dikhususkan bagi ECO (Engineering Change order) dibuat ?
2. Sudahkan suatu kajian teknis formal dilakukan untuk menilai kebenaran teknis ?
3. Sudahkan standar rekayasa perangkat lunak diikuti secara tepat ?
4. Sudahkah perubahan ditandai dalam SCI ?.
5. Sudahkan tanggal perubahan dan penulis perubahan ditentukan ?
6. Apakah atribut objek konfigurasi mercerminkan perubahan ?
7. Sudahkah prosedur SCM untuk menulis perubahan, mencatat dan melaporkannya diikuti ?
8. Sudahkan semua SCI yang berhubungan diperbarui dengan tepat ?

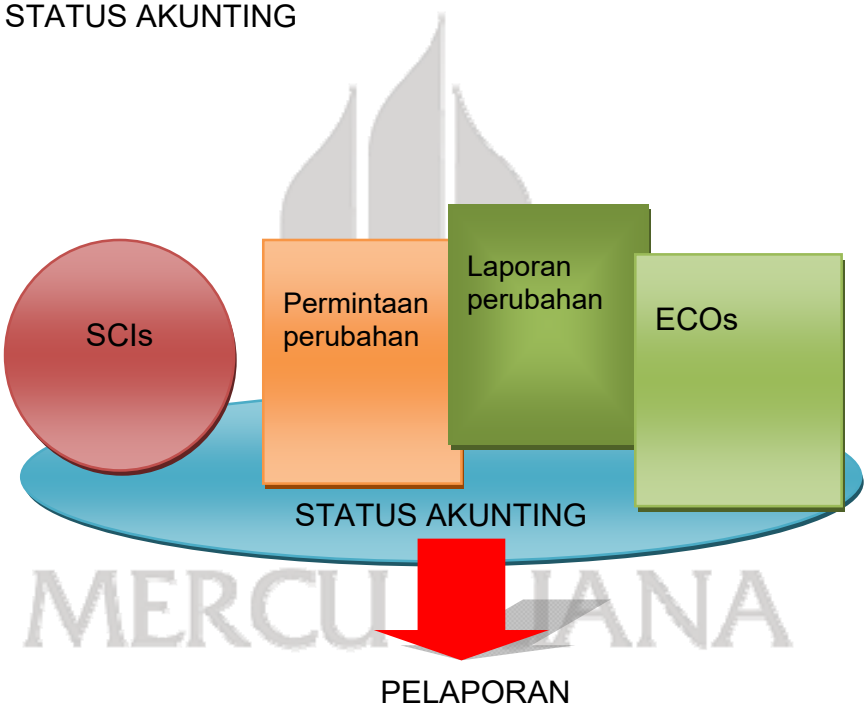
BAGAIMANA KITA YAKIN BAHWA PERUBAHAN SUDAH DIIMPLEMENTASI KAN DENGAN TEPAT ?

1. **Formal Technical Reviews:** focuses on the technical correctness of the configuration object that has been modified
2. **Software Configuration Audit:** complements the Formal Technical Reviews by assessing for a configuration object for characteristics that are generally not considered during review

GAMBARAN SCM AUDIT



GAMBARAN STATUS AKUNTING



CONFIGURATION STATUS REPORTING (CSR)

Tugas / kerja SCM yang menjawab pertanyaan-pertanyaan berikut :

1. Apa yang terjadi (What happened) ?
2. Siapa yang mengerjakannya (Who did it) ?
3. Kapan ini terjadi (When did it happen) ?
4. Adakah yang lain akan terpengruh (What else will be affected) ?

RANGKUMAN

- Manajemen konfigurasi perangkat lunak adalah aktivitas pelindung yang diterapkan pada seluruh proses perangkat lunak
- SCM mengidentifikasi control, audit, dan modifikasi laporan, yang selalu terjadi pada saat perangkat lunak sedang dikembangkan dan setelah dilepas ke pelanggan.
- Semua informasi yang diproduksi sebagai bagian dari proses perangkat lunak menjadi bagian dari suatu konfigurasi perangkat lunak.
- Konfigurasi tersebut harus diorganisir dengan cara memungkinkan control perubahan secara teratur.
- Sekali suatu objek konfigurasi dikembangkan dan dikaji, objek tersebut menjadi baseline
- Perubahan terhadap objek baseline menghasilkan kreasi versi baru dari objek tersebut
- Evolusi dari suatu program dapat ditelusuri dengan mengamati sejarah revisi dari semua konfigurasi objek
- Kontrol perubahan (change control) adalah aktivitas procedural yang menjamin kualitas dan konsistensi pada saat perubahan dibuat untuk suatu objek konfigurasi.
- Audit konfigurasi adalah aktivitas SQA yang membantu memastikan bahwa kualitas dijaga pada saat perubahan dilakukan.

UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Rekayasa Sistem

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	10		Tim Dosen

Abstract

Mejelaskan prinsip analisis perangkatm lunak

Kompetensi

Menjelaskan proses dan unsusr-unsur dalam mengembangkan atau membangun sebuah sistem

REKAYASA SISTEM
(*SYSTEM ENGINEERING*)

Machiavelli pernah berkata “Tidak ada yang lebih sukar dilakukan, lebih membahayakan untuk dilakukan atau lebih tidak pasti dalam keberhasilannya, daripada memimpin didalam pembukaan order yang baru”.

Rekayasa Perangkat Lunak terjadi :

- sebagai konsekuensi dari suatu proses yang disebut rekayasa system.

Rekayasa sistem berfokus pada :

- variasi elemen-elemen, analisa, rancangan dan mengelola elemen-elemen tersebut kedalam sistem yang dapat berbentuk produk, layanan (service) atau teknologi untuk transformasi informasi atau control

Proses rekayasa system :

- disebut rekayasa informasi bila konteks kerja rekayasa berfokus pada perusahaan bisnis.
- pada saat produk akan dibuat, proses ini disebut rekayasa produk.

Baik rekayasa informasi maupun rekayasa produk :

- cenderung untuk membawa orde kepada pengembangan sistem berbasis komputer.

SISTEM BERBASIS KOMPUTER

Kata sistem banyak digunakan (misal sistem politik, sistem pendidikan, sistem penerbangan).

Menurut Webster, definisi system berbasis computer adalah sebagai :

> “Serangkaian atau tatanan elemen-elemen yang diatur untuk mencapai tujuan yang ditentukan sebelumnya melalui pemrosesan informasi”

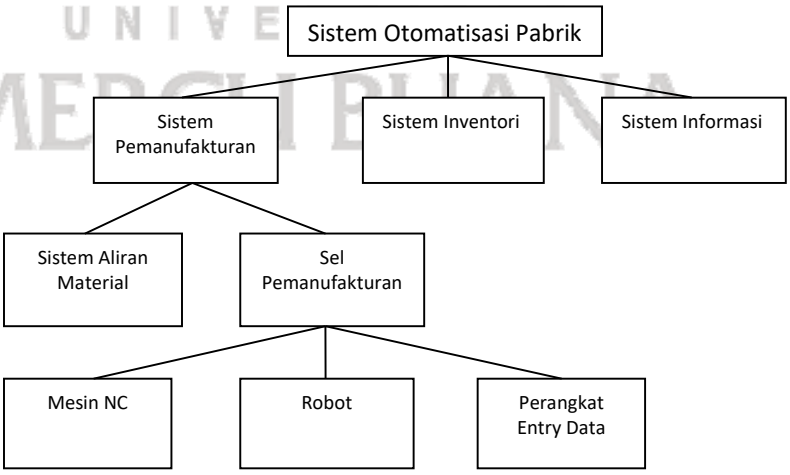
Untuk mencapai tujuan tersebut, system berbasis computer menggunakan berbagai elemen system :

- Perangkat lunak. Program computer, struktur data dan dokumen yang berhubungan yang berfungsi untuk mempengaruhi metode logis, prosedur dan control yang dibutuhkan.
- Perangkat keras. Perangkat elektronik yang memberikan kemampuan perhitungan dan perangkat elektromekanik (misalnya sensor, rotor, pompa) yang memberikan fungsi dunia eksternal

- Manusia. Pemakai dan operator perangkat keras dan perangkat lunak.
- Database. Kumpulan informasi yang besar dan terorganisasi yang di akses melalui perangkat lunak.
- Dokumentasi. Manual, formulir dan informasi deskriptif lainnya yang menggambarkan penggunaan atau pengoperasian system.
- Prosedur. Langkah-langkah yang menentukan penggunaan khusus dari masing-masing elemen system atau konteks procedural dimana system berada.

Elemen-elemen yang bergabung dengan berbagai cara untuk mentranformasi informasi, misal nya bagian pemasaran mentransformasi data penjualan kasar ke dalam profil pembeli produk tertentu dan sebuah robot mentransformasi sebuah file perintah yang berisi instruksi khusus ke dalam serangkaian sinyal control yang menyebabkan berbagai gerakan fisik tertentu.

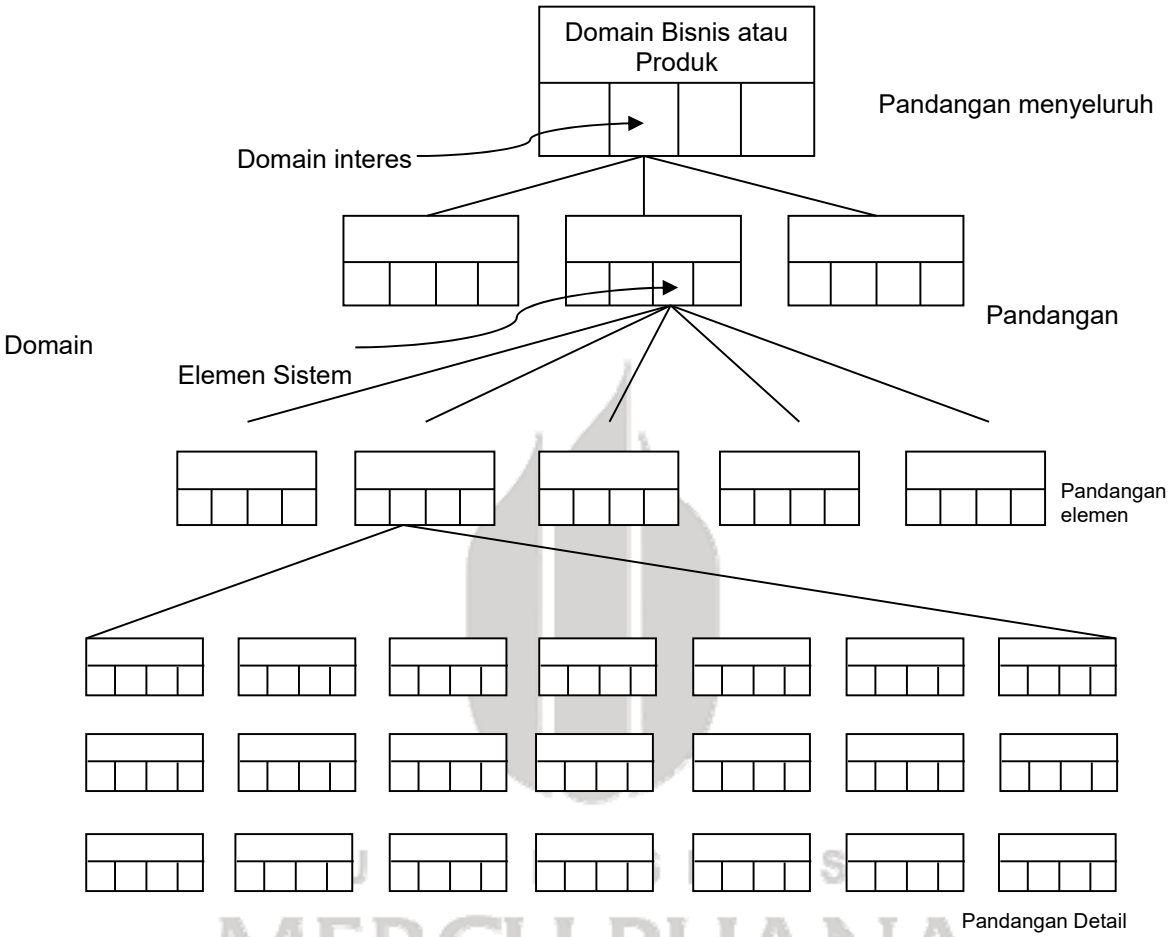
Satu karakteristik sistem berbasis komputer yang rumit adalah bahwa elemen yang berisi satu system juga dapat mewakili satu elemen makro dari suatu system yang sangat besar.
Elemen Makro adalah :
Suatu system berbasis computer yang merupakan bagian dari system berbasis computer yang lebih besar lagi, seperti contoh gambar 10.1 dibawah ini yaitu otomatisasi pabrik yang pada dasarnya merupakan hirarki system



Gambar 10.1 : Sistem dari banyak sistem

HIRARKI REKAYASA SISTEM

Tanpa melihat domain fokusnya, rekayasa melingkupi sekumpulan metode dari atas ke bawah (top-down) dan dari bawah ke atas (bottom-up) untuk mengendalikan hirarki yang terlihat pada gambar 10.2 berikut



Gambar 10.2 : Hirarki rekayasa perangkat lunak

PEMODELAN SISTEM

Untuk membangun sebuah model system, perekayasa harus mempertimbangkan sejumlah faktor pembatasan yaitu :

a. Asumsi : yang mengurangi jumlah permutasi dan variasi yang mungkin sehingga memungkinkan sebuah model mencerminkan masalah dengan cara yang dapat dipertanggung jawabkan.
b. Penyederhanaan : yang memungkinkan model diciptakan dengan waktu yang tepat
c. Pembatasan : yang membantu membatasi system

d. Batasan : yang menunjukkan cara dimana model diciptakan dan pendekatan dilakukan pada saat model diimplementasikan.
e. Preferensi : yang menunjukkan arsitektur yang dipilih untuk semua data, fungsi dan teknologi

REKAYASA PROSES BISNIS / INFORMASI ; SEBUAH KAJIAN

Tujuan dari rekayasa proses bisnis / informasi (information engineering) adalah untuk mendefinisikan / menentukan arsitektur yang memungkinkan suatu bisnis dapat memanfaatkan informasi secara efektif. Rekayasa informasi juga bekerja untuk membuat suatu rencana menyeluruh guna mengimplementasikan arsitektur-arsitektur tersebut. Ada tiga arsitektur berbeda yang harus dianalisis dan dirancang dalam konteks sasaran dan tujuan bisnis yaitu :

a. Arsitektur Data
b. Arsitektur Aplikasi
c. Arsitektur Teknologi

Arsitektur data :

Memberikan kerangka kerja untuk kebutuhan informasi dari bisnis atau fungsi bisnis. Blok bangunan individual dari arsitektur tersebut merupakan objek data yang digunakan oleh suatu database dan ditransformasikan untuk memberikan informasi yang melayani kebutuhan bisnis.

Arsitektur aplikasi :

Melingkupi elemen-elemen dari suatu system yang mentransformasikan objek ke dalam arsitektur data untuk keperluan bisnis. Dalam konteks buku ini, kita menganggap bahwa arsitektur aplikasi sebagai system program (perangkat lunak) yang melakukan transformasi tersebut.

Arsitektur teknologi :

Sebagai dasar bagi dan arsitektur aplikasi. Infrastruktur menyangkut perangkat keras dan perangkat lunak yang digunakan untuk mendukung aplikasi dan data yang mencakup computer dan jaringan computer, hubungan telekomunikasi, teknologi penyimpangan dan arsitektur (seperti client / server) yang telah dirancang untuk mengimplementasikan teknologi tersebut.

REKAYASA PROSES BISNIS (BUSINESS PROCESS ENGINEERING = BPE)

- Menggunakan sekumpulan prosedur yang terpadu, metode dan alat bantu untuk mengidentifikasi bagaimana system informasi dapat sesuai dengan tujuan strategis perusahaan.
- Perhatian utama adalah kepada tujuan perusahaan dan setelah itu ke area bisnis
- Menciptakan model perusahaan, model data dan model proses
- Membuat suatu kerangka kerja yang lebih baik untuk distribusi dan pengawasan manajemen informasi.

HIRARKI BPE

Ada 4 hirarki dalam BPE yaitu :

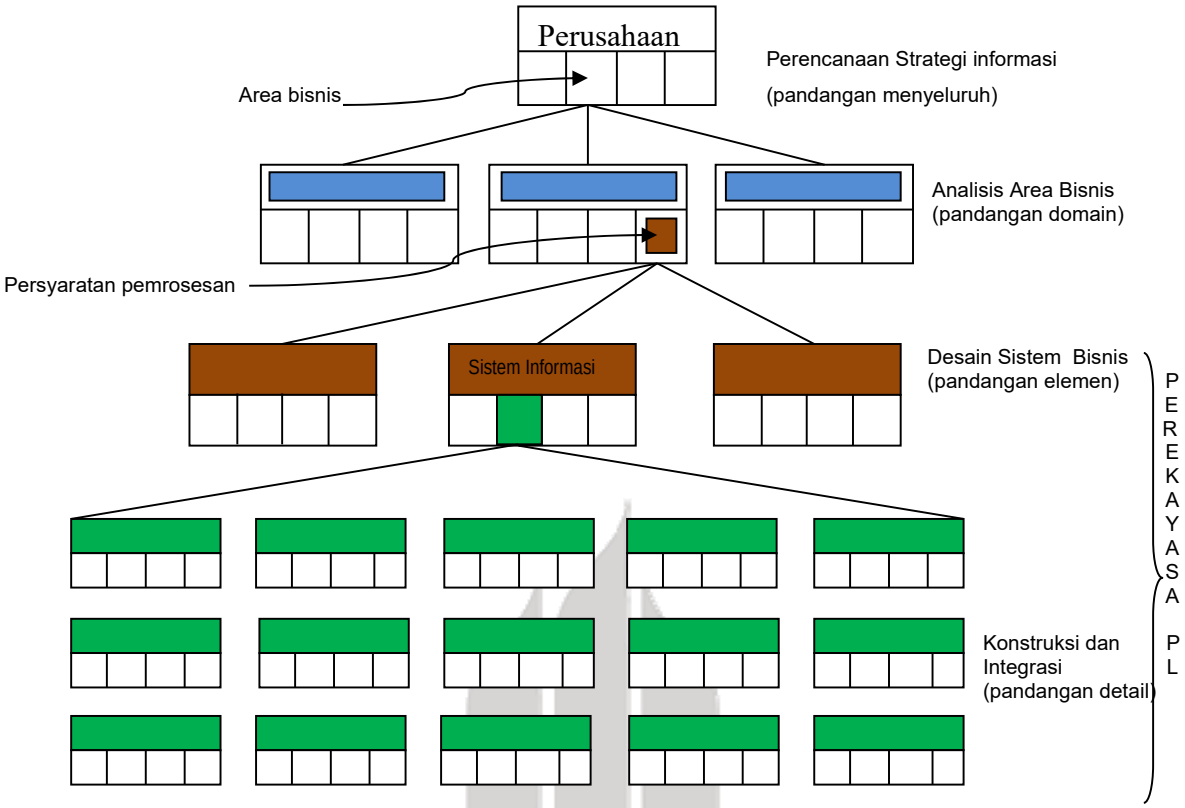
a. Perencanaan Strategi Informasi (Information Strategy Planning = ISP) yang bertanggung jawab menyiapkan :
   - Penentuan Tujuan Strategis
   - Identifikasi aturan-aturan bisnis dan factor-faktor sukses
   - Model perusahaan

b. Analisis Area Bisnis (Business Area Analysis = BAA) yang bertugas menyiapkan :
   - Model pelayanan dan proses
   - Hubungan proses dan data

c. Rekayasa Aplikasi yang bertugas menyiapkan :
   - Rekayasa perangkat lunak
   - Pemodelan aplikasi / modul yang ditujukana dan dibatasi oleh ISP

d. Konstruksi dan Pengiriman (delivery) yang bertugas menyiapkan :
   - Penggunaan CASE dan Ujicoba

Gambar 10.3 memperlihatkan hirarki rekayasa informasi.



Gambar 10.3 : Hirarki Rekayasa Informasi.

PERENCANAAN STRATEGI INFORMASI

Langkah pertama rekayasa informasi adalah perencanaan strategi informasi (ISP) dan sasaran keseluruhan dari ISP adalah :

1. Menentukan sasaran dan tujuan bisnis strategis
2. Mengisolasi faktor sukses kritis yang memungkinkan bisnis mencapai tujuan-tujuan dan sasaran tersebut
3. Menganalisis pengaruh teknologi dan otomatisasi terhadap tujuan dan sasaran
4. Menganalisis informasi yang ada untuk menentukan perannya dalam pencapaian sasaran dan tujuan

KONSTRUKSI DAN INTEGRASI

- Fokus terhadap rincian implementasi

- Contoh kasusnya seperti : membangun database yang tepat, membangun aplikasi yang menggunakan komponen perangkat lunak, memilih elemen yang tepat dari infrastruktur teknologi untuk mendukung rancangan selama proses rancangan system bisnis (BSD)

REKAYASA PRODUK

Tujuan Rekayasa Produk adalah untuk menerjemahkan keinginan pelanggan dengan serang kaian kemampuan yang terbatas kedalam produk yang sedang bekerja, seperti yang terlihat pada gambar 10.4 berikut.

KEBUTUHAN DALAM REKAYASA

- Menentukan apa yang diperlukan pelanggan
- Analisa dan negosiasi – mengerti hubungan antara kebutuhan pelanggan yang beragam dan rentang hubungan untuk mencapai hasil yang terbaik.
- Spesifikasi kebutuhan – membangun model fisik kebutuhan
- Pemodelan system – membangun contoh kebutuhan yang dapat membantu untuk mengoreksi kesalahan, kekurangan dan konsistensi
- Validasi – mengkaji ulang model
- Manajemen – Identifikasi, mengawasi dan menelusuri jalur kebutuhan dan perubahan yang akan terjadi

ANALISIS SISTEM

Analisis system dilakukan dengan sasaran sebagai berikut :

1. Mengidentifikasi kebutuhan pelanggan
2. Mengevaluasi konsep system untuk feasibilitas
3. Melakukan analisis teknis dan ekonomis
4. Mengalokasikan fungsi-fungsi untuk perangkat keras, perangkat lunak, manusia, database dan elemen system yang lain
5. Membuat batasan biaya dan jadual

6. Menciptakan definisi system yang membentuk fondasi bagi semua kerja rekayasa tingkat berikut, baik keahlian perangakt keras maupun perangkat lunak

STUDI KELAYAKAN

Selama rekayasa produk, kita perlu mengkonsentrasikan perhatian pada empat area utama yaitu :

1. Kelayakan ekonomis
2. Kelayakan teknis
3. Kelayakan legal
4. Alternatif

Namun demikian, pertimbangan yang biasanya dihubungkan dengan kelayakan teknis meliputi :

- Resiko Pengembangan : apakah elemen system dirancang dengan baik sehingga fungsi dan kinerja yang perlu dapat dicapai dalam batasan yang ditentukan selama analisis.
- Keberadaan sumber daya : Apakah staf terlatih dapat diperoleh untuk mengembangkan elemen system yang dibicarakan
- Teknologi : sudahkan teknologi yang relevan dikembangkan ke suatu keadaan yang dapat mendukung system

KEUNTUNGAN YANG MUNGKIN DARI (REKAYASA) SISTEM INFORMASI :

1. Keuntungan dari konstribusi tugas kalkulasi dan pencetakan
2. Keuntungan dari konstribusi pada tugas-tugas pemeliharaan record
3. Keuntungan dari konstribusi tugas pencarian record
4. Keuntungan dari konstribusi kapabilitas menstruktur ulang system
5. Keuntungan dari konstribusi analisis dan simulasi kapabilitas
6. Keuntungan dari konstribusi pada control proses dan sumber daya

BIAYA-BIAYA YANG MUNGKIN DARI (REKAYASA) SISTEM INFORMASI

1. Biaya-biaya usaha, seperti biaya konsultasi, beli peralatan, instalasi, modifikasi tempat
2. Biaya awal seperti software system operasi, instalasi peralatan komunikasi, biaya sewa

3. Biaya (yang berhubungan dengan) proyek, seperti pembelian software aplikasi, modifikasi software, pelatihan, dokumentasi
4. Biaya rutin, seperti biaya pemeliharaan, biaya listrik, telepon, biaya depresiasi.

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Rekayasa Sistem

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	11		Tim Dosen

Abstract

Mejelaskan prinsip analisis perangkatm lunak

Kompetensi

Menjelaskan proses dan unsusr-unsur dalam mengembangkan atau membangun sebuah sistem

PRINSIP DAN KONSEP ANALISA
(ANALYSIS CONCEPT AND PRINCIPLES)

PEMAHAMAN

Pemahaman lengkap mengenai persyaratan perangkat lunak sangat penting bagi keberhasilan usaha pengembangan perangkat lunak. Tidak peduli bagaimana perangkat lunak dirancang atau dikodekan, program yang dianalisis dan ditentukan secara tidak baik akan mengecewakan pemakainya dan akan membawa kegagalan bagi pengembangnya.

Dalam konteks perangkat lunak, analisis merupakan sebuah :

- Penemuan
- Perbaikan
- Pemodelan
- Spesifikasi (baru)

SIAPA YANG BERPERAN DALAM ANALISIS ?

- Pengembang maupun pelanggan harus berperan aktif
- Pelanggan berusaha memformulasikan kembali konsep yang tidak jelas dari fungsi perangkat lunak dan kinerja kedalam detail yang konkret.
- Pengembang bertindak sebagai integrator, konsultan dan pemecah masalah

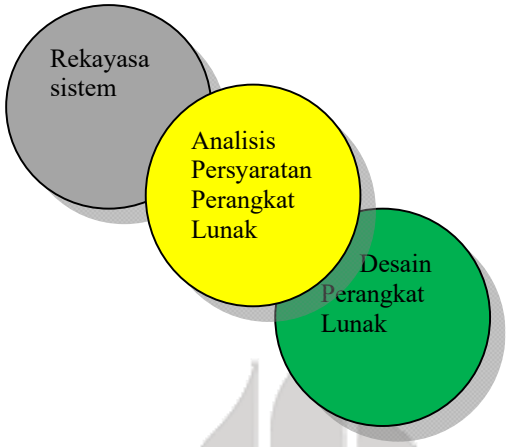
Ruang lingkup perangkat lunak secara mendasar dikembangkan oleh perekayasa system dan diperbaiki selama perencanaan proyek perangkat lunak dan diperbaiki secara detail (rinci).

APA YANG MENJADI MASALAH SEBENARNYA ?

- Pelanggan hanya memiliki ide yang samar-samar apa yang dibutuhkan
- Pengembang akan menghasilkan sesuatu dengan mengacu kepada “ide yang samar-samar”, dengan asumsi bahwa “kita akan mengerjakan rincian pekerjaan sesuai tahapan (langkah)”
- Pelanggan akan terus mengikuti perubahan
- Pengembang akan “dirugikan” oleh perubahan-perubahan ini, membuat kesalahan-kesalahan dalam spesifikasi dan pengembangan

ANALISIS PERSYARATAN

Analisis persyaratan adalah sebuah tugas rekayasa perangkat lunak yang menjembatani jurang antara alokasi perangkat lunak tingkat system dan perancangan perangkat lunak seperti terlihat pada gambar 11.1



Gambar 11.1 Analisis dan kesenjangan antara rekayasa system dan desain perangkat lunak

Analisis persyaratan perangkat lunak dapat dibagi menjadi 5 (lima) area kerja yaitu :

1. Pengenalan masalah

   Mempelajari spesifikasi system (bila ada) dan rencana proyek perangkat lunak dalam suatu konteks system dan mengkaji ruang lingkup perangkat lunak dalam suatu konteks system dan mengkaji ruang lingkup perangkat lunak yang telah digunakan untuk memunculkan estimasi perencanaan
2. Evaluasi dan sintesis
   - Membatasi semua objek data yang dapat diobservasi secara eksternal
   - Mengevaluasi aliran dan muatan informasi
   - Mendefinisikan dan menguraikan semua fungsi perangkat lunak
   - Memahami tingkah laku perangkat lunak dalam konteks kejadian yang mempengaruhi system
   - Membangun karakteristik interface system
   - Menemukan batasan desain tambahan
3. Pemodelan

   Menyiapkan system dalam ukuran yang kecil-kecil sebelum penerapan dengan system yang sebenarnya

4. Spesifikasi

   Menetapkan system dalam kondisi yang sebenarnya

5. Kajian

   Melakukan evaluasi dan pengujian formal terhadap penerapan yang telah dilakukan apakah sasaran yang yang ditetapkan tercapai atau tidak.

TEKNIK KOMUNIKASI

- Merupakan permulaan yang (selalu) perlu dilakukan agar seorang pelanggan memiliki masalah yang dapat dipertanggung jawabkan melalui pemecahan berbasis komputer
- Agar pengembang dapat merespon permintaan bantuan (help) dari pelanggan
- Biasanya jalan komunikasi ke pemahaman penuh dengan “lobang-lobang”

MENGAWALI PROSES

- Untuk menjembatani jurang / lobang-lobang komunikasi antara pelanggan dan pengembang, sekaligus untuk memulai proses komunikasi, perlu dilakukan pertemuan pendahuluan atau wawancara
- Harus dimulai dengan pertanyaan-pertanyaan yang bebas konteks :
  - Siapa dibalik permintaan untuk pekerjaan ini ?
  - Siapa yang akan menggunakan pemecahan ini ?
  - Apa keuntungan ekonomi dari pemecahan yang berhasil ?
  - Apakah ada sumber lain untuk pemecahan yang anda inginkan ?
- Dilanjutkan dengan pertanyaan agar seorang analis mendapat pemahaman yang lebih baik akan mengenai masalah dari pelanggan
  - Bagaimana anda akan menandai output yang baik ?
  - Masalah apa yang akan diselesaikan oleh pemecahan ini ?
  - Dapatkah anda memperlihatkan kepada saya (atau menjelaskan) lingku ngan dimana pemecahan tersebut akan digunakan ?
  - Apakah masalah atau batasan kinerja yang khusus yang akan mempenga ruhi cara pemecahan tersebut didekati ?

- Diakhiri dengan pertanyaan yang berfokus pada efektivitas pertemuan
  - Apakah anda adalah orang yang tepat untuk menjawab pertanyaan-pertanyaan ini ? dan apakah jawaban anda bersifat resmi ?

- Apakah pertanyaan saya ini relevan dengan masalah yang anda hadapi ?
- Apakah anda mengajukan terlalu banyak pertanyaan ?
- Apakah ada orang lain yang dapat memberikan informasi tambahan ?
- Apakah ada hal lain yang harus saya tanyakan kepada anda ?

FAST (Facilitated Application Specification Techniques)

TENTANG FAST

Memacu kreasi kerjasama dari tim (pelanggan dan pengembang) yang bekerja sama untuk :

- Mengidentifikasi masalah
- Menyiapkan elemen-elemen solusi
- Menegosiasikan pendekatan yang berbeda
- Menetapkan sebelumnya kebutuhan solusi yang diperlukan

Banyak pendekatan yang digunakan dan masing-masing pendekatan menggunakan scenario yang berbeda, namun semuanya menerapkan variasi tuntunan dasar berikut ini:

- Pertemuan dilakukan di sisi netral dan dihadiri baik oleh pengembang maupun pelanggan
- Aturan main untuk persiapan dan partisipasi dibuat
- Perlunya agenda
- Perlunya seorang fasilitator
- Harus adanya mekanisme definisi

PANDUAN FAST

J. Wood dan D. Silver menyarankan beberapa panduan umum FAST yang dapat digunakan yaitu :

- Peserta harus menghadiri semua rapat
- Semua peserta adalah sama
- Persiapan harus sama pentingnya dengan rapat yang sebenarnya
- Semua dokumen sebelum rapat harus dikaji ulang
- Lokasi rapat diluar ruangan terkadang diperlukan
- Tentukan agenda dan jangan sampai mengalami perubahan
- Jangan sampai terbawa dalam hal-hal teknis yang terlalu rinci

PENYEBARAN FUNGSI KUALITAS (QUALITY FUNCTION DEPLOYMENT = QFD)

QFD sebagai perkenalan :

- Teknik manajemen kualitas yang menterjemahkan kebutuhan pelanggan kedalam kebutuhan teknis untuk perangkat lunak
- Pertama kali diperkenalkan di Jepang untuk memaksimalkan kepuasan pelanggan
- Menekankan pemahaman tentang apa yang berguna kepada pelanggan dan kemudian menyebarkan nilai-nilai tersebut melalui proses rekayasa

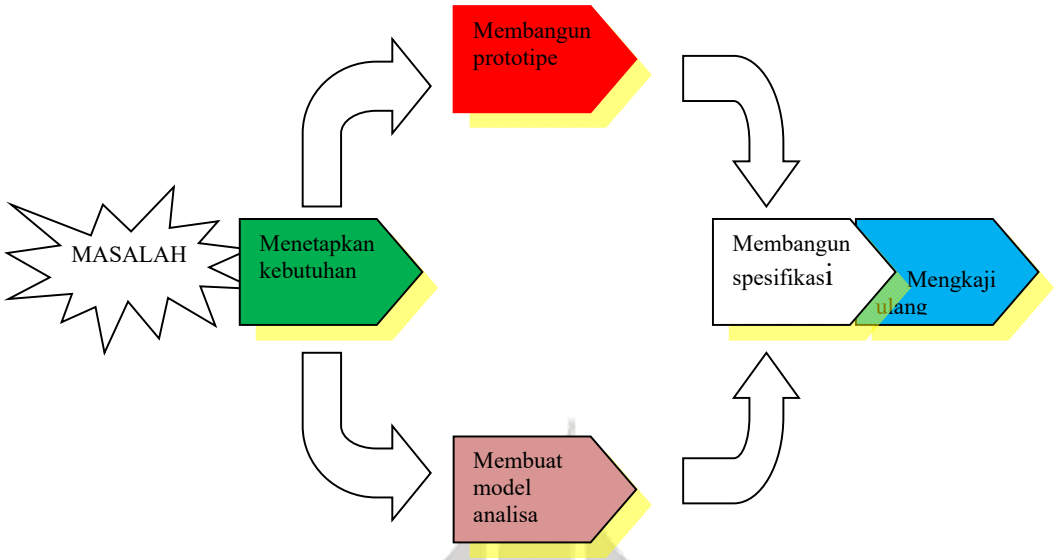
QFD mengidentifikasi tiga tipe persyaratan yaitu :

1. Persyaratan normal : Sasaran dan tujuan bagi sebuah produk atau system selama pertemuan dengan pelanggan. Bila persyaratan ini ada, maka pelanggan akan menjadi puas, misalnya tampilan grafis yang sempurna.
2. Persyaratan yang diharapkan : Persyaratan ini implicit terhadap produk atau system yang sangat fundamental sehingga pelanggan tidak menyatakannya secara eksplisit. Ketidakhadirannya akan menyebabkan ketidakpuasan yang sangat mendalam. Contohnya adalah mudahnya operasional interaksi manusia dan mesin, reliabilitas dan kebenaran operasional keseluruhan dan mudahnya instalasi perangkat lunak
3. Exciting requirement : Persyaratan ini sangat diharapkan oleh pelanggan dan terbukti sangat memuaskan bila ada, misal kemampuan perangkat pengolah kata yang memiliki kemampuan layout halaman, dsb.

GAMBARAN KONSEP QFD :

- Penyebaran fungsi, menentukan nilai (seperti yang diharapkan pelanggan) dari setiap fungsi yang dibutuhkan oleh system.
- Penyebaran informasi, mengidentifikasi objek data dan kejadian
- Penyebaran tugas, yang melatih kebiasaan dari system
- Analisa nilai, menetapkan prioritas relative kebutuhan

ANALISA PROSES SECARA UMUM



PRINSIP ANALISA KESATU

Data Domain Model :

- Menetapkan objek data
- Menggambarkan atribut data
- Menetapkan hubungan data

PRINSIP ANALISA KEDUA

Fungsi Model :

- Mengidentifikasi fungsi yang (dapat) merubah objek data
- Mengindikasikan berapa data yang melalui system
- Mewakili data produsen dan konsumen

PRINSIP ANALISA KETIGA

Model Kebiasaan :

- Mengindikasikan states yang berbeda dari system
- Menetapkan kejadian yang mungkin menyebabkan perubahan pada state

PRINSIP ANALISA KEEMPAT

Partisi Model :

- Menyaring setiap model untuk mewakili level yang lebih rendah dari abstraksi
    - Menyaring objek data
    - Membuat hirarki fungsi
    - Mewakili kebiasaan pada tingkatan yang berbeda tiap detil
- Membuat partisi horizontal dan vertikal

PRINSIP ANALISA KELIMA :

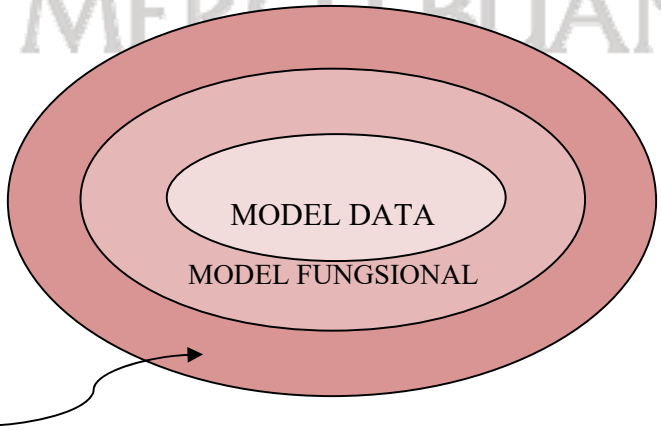
Intisari :

- Memulai focus intisari masalah tanpa memperhatikan rincian implementasi

BEBERAPA PRINSIP YANG DIKEMUKAKAN DAVIS :

- Mengerti masalah sebelum kita memulai menciptakan model analisa
- Membangun protipe yang memungkinkan pelanggan untuk mengerti bagaimana pelanggan mengerti interaksi manusia dan mesin dapat terjadi
- Mencatat hal-hal yang baru dan alasan untuk setiap kebutuhan
- Menggunakan gambaran bertingkat setiap kebutuhan
- Memprioritaskan kebutuhan
- Bekerja untuk menghilangkan keragu-raguan

MODEL ANALISA :



BEHAVIORAL MODEL

KAJIAN SPESIFIKASI :

Kajian dari suatu spesifikasi persyaratan perangkat lunak dilakukan baik oleh pelanggan maupun pengembang perangkat lunak dan harus dilakukan dengan sangat hati-hati.

Kajian ini akan memastikan bahwa spesifikasi sudah lengkap, konsisten dan akurat. Untuk itu, pertanyaan-pertanyaan di bawah ini dapat diajukan :

- Apakah tujuan dan sasaran yang dinyatakan bagi perangkat lunak tetap konsisten dengan tujuan dan sasaran system ?
- Apakah interface penting ke semua elemen system sudah digambarkan ?
- Apakan aliran informasi dan struktur didefinisikan dengan tepat bagi domain masalah ?
- Apakah diagram jelas ? dapatkah masing-masing berdiri sendiri tanpa teks pendamping ?
- Apakah fungsi mayor tetap ada dalam ruang lingkup, dan sudahkan digambarkan dengan tepat ?
- Apaka tingkah laku perangkat lunak konsisten dengan informasi yang harus diprosesnya dengan fungsi yang harus dilakukannya /
- Apakah batasan desain realistis ?
- Apakah resiko teknologis pengembangan sudah dipertimbangkan ?
- Apakah criteria validasi dinyatakan secara detail ? Apakah criteria itu tepat untuk menggambarkan sebuah system yang berhasil ?
- Apakah ada inkonsistensi, penghilangan atau redundancy ?
- Apakah kontak dengan pelanggan sudah lengkap ?
- Apakah pemakai sudah mengkaji manual pemakai permulaan atau prototype ?
- Bagaimana estimasi perencanaan mempengaruhi ?

RANGKUMAN

- Analisis persyaratan adalah langkah teknis pertama pada proses rekayasa perangkat lunak
- Analisis harus berfokus pada domain informasi, fungsional dan tingkah laku dari masalah

- Dalam beberapa kasus tidaklah mungkin untuk secara lengkap memspesifikasi suatu masalah pada tahap awal
- Spesifikasi persyaratan perangkat lunak dikembangkan sebagai akibat dari analisis

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

PEMODELAN ANALISIS
(ANALYSIS MODELLING)

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	12		Tim Dosen

Abstract

Menggambarkan hasil analisis pada sebuah model

Kompetensi

Mengupayakan mahasiswa mampu mengerti pemodelan serta menggambarkan sistem

PEMODELAN ANALISIS
(ANALYSIS MODELLING)

PEMODELAN ANALISIS

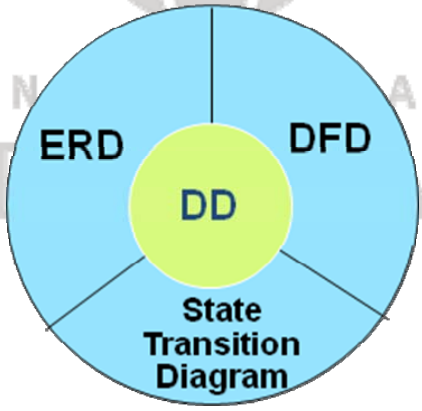
- Adalah model yang menggunakan kombinasi teks dan diagram untuk menggam barkan kebutuhan data, fungsi dan kebiasaan yang mudah dimengerti dan ditinjau.
- Biasanya menggunakan metode :
  1. Analisa terstruktur
  2. Analisa Berorientasi Objek

ELEMEN MODEL ANALISIS

Model analisis harus dapat mencapai tiga sasaran utama yaitu :

1. Untuk menggambarkan apa yang dibutuhkan pelanggan
2. Untuk membangun dasar bagi pembuatan desain perangkat lunak
3. Untuk membatasi serangkaian persyaratan yang dapat divalidasi begitu perangkat lunak dibangun

STRUKTUR MODEL ANALISIS



ERD = Entity Relationship Diagram

DFD = Data Flow Diagram

DD = Data Dictionary

Pada inti model ada Kamus Data (DD) – penyimpan yang berisi deskripsi dari semua objek data yang dikonsumsi atau diproduksi oleh perangkat lunak.

Disitu ada tiga diagram yang mengelilingi inti yaitu :

a. Entity Relationship Diagram yang menggambarkan hubungan antara objek data. ERD adalah notasi yang digunakan untuk melakukan aktivitas pemodelan data.
b. Data Flow Diagram yang melayani dua tujuan :
    i. Memberikan indikasi mengenai bagaimana data di transformasi pada saat data bergerak melalui system
    ii. Untuk menggambarkan fungsi-fungsi (dan sub-fungsi) yang mentransformasi aliran data
c. State Transition Diagram yang menunjukkan bagaimana system bertingkah laku sebagai akibat dari kejadian eksternal. Untuk melakukannya, STD menunjukkan berbaga model tingkah laku (disebut state) system dan cara dimana transisi dibuat dari satu state ke state lainnya. STD berfungsi sebagai dasar bagi pemodelan tingkah laku.

DIMANA HARUS MEMULAI PEMODELAN ANALISIS ?

- A statement of scope can be acquired from:
    - the FAST working document
    - A set of use-cases
- the statement of scope must be “parsed” to extract data, function and behavioral domain information

STATEMENT OF SCOPE

- Adalah deskripsi relatif dari sistem yang dibangun untuk
    - indicates data that are input, output and basic functionality
    - indicates conditional processing (at a high level)
    - implies certain constraints and limitations

IDENTIFIKASI OBJEK DAN OPERASI

- Tentukan “objek” dengan mengaris bawahi semua kata benda yang ditulis dalam statement of scope seperti :
    - Data produsen maupun konsumen
    - Tempat dimana data akan disimpan
    - “Composite” item data

- Tentukan “operasi” dengan member garis bawah yang ganda untuk semua kata kerja aktif terhadap
  - Proses-proses yang berkaitan dengan aplikasi
  - Tranformasi (perubahan bentuk) data
- Pertimbangkan “layanan” lain yang akan dibutuhkan oleh objek (masalah)

DATA MODELING dan DIAGRAM HUBUNGAN ENTITAS (ERD)

Mengapa (menggunakan) Data Modeling?

- memeriksa objek data terbebas dari proses
- memusatkan perhatian pada domain data
- membuat model pada level abstraksi pelanggan
- mengindikasi bagaimana hubungan objek satu dengan lainnya

Apa itu Objek Data ?

Objek adalah sesuatu yang digambarkan dengan sekumpulan atribut (data item) dan akan dimanipulasi dengan menggunakan perangkat lunak

- Setiap contoh dari objek (seperti buku) dapat diidentifikasi secara unik
- Setiap bentuk proses memiliki peranan penting dalam system
- Setiap objek digambarkan dengan atribut dari data itemnya sendiri

Bentuk-bentuk objek

a. Eksternal entity (printer)
b. Benda (laporan, tampilan)
c. Kejadian
d. Peranan (manajer, supervisor)
e. Unit organisasi (divisi, kelompok)
f. Tempat (pabrik, bank)
g. Struktur (record pegawai)

Objek Data dan Atribut

Adalah objek data yang terdiri dari kumpulan atribut yang bertindak seperti aspek, kualitas, cirri / karakteristik atau pendefinisj objek

Contoh objek : mobil

, maka atributnya adalah :

- ✓ Membuat
- ✓ model
- ✓ jenis bodi
- ✓ harga
- ✓ kode-kode pilihan

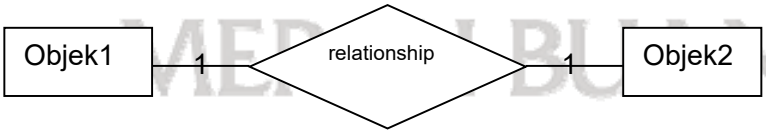
Apa yang dimaksud dengan relationship ?

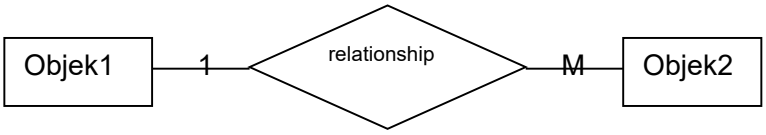
Kaitan yang mengindikasikan “hubungan” dari “fakta” yang harus “diingat” oleh system dan tidak dapat dihitung dan disampaikan secara mekanik.

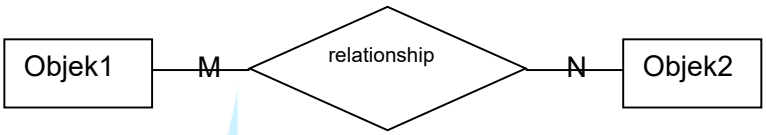
- several instances of a relationship can exist
- objects can be related in many different ways

NOTASI ERD

ada tiga bentuk relationship yang dapat terjadi antar entitas





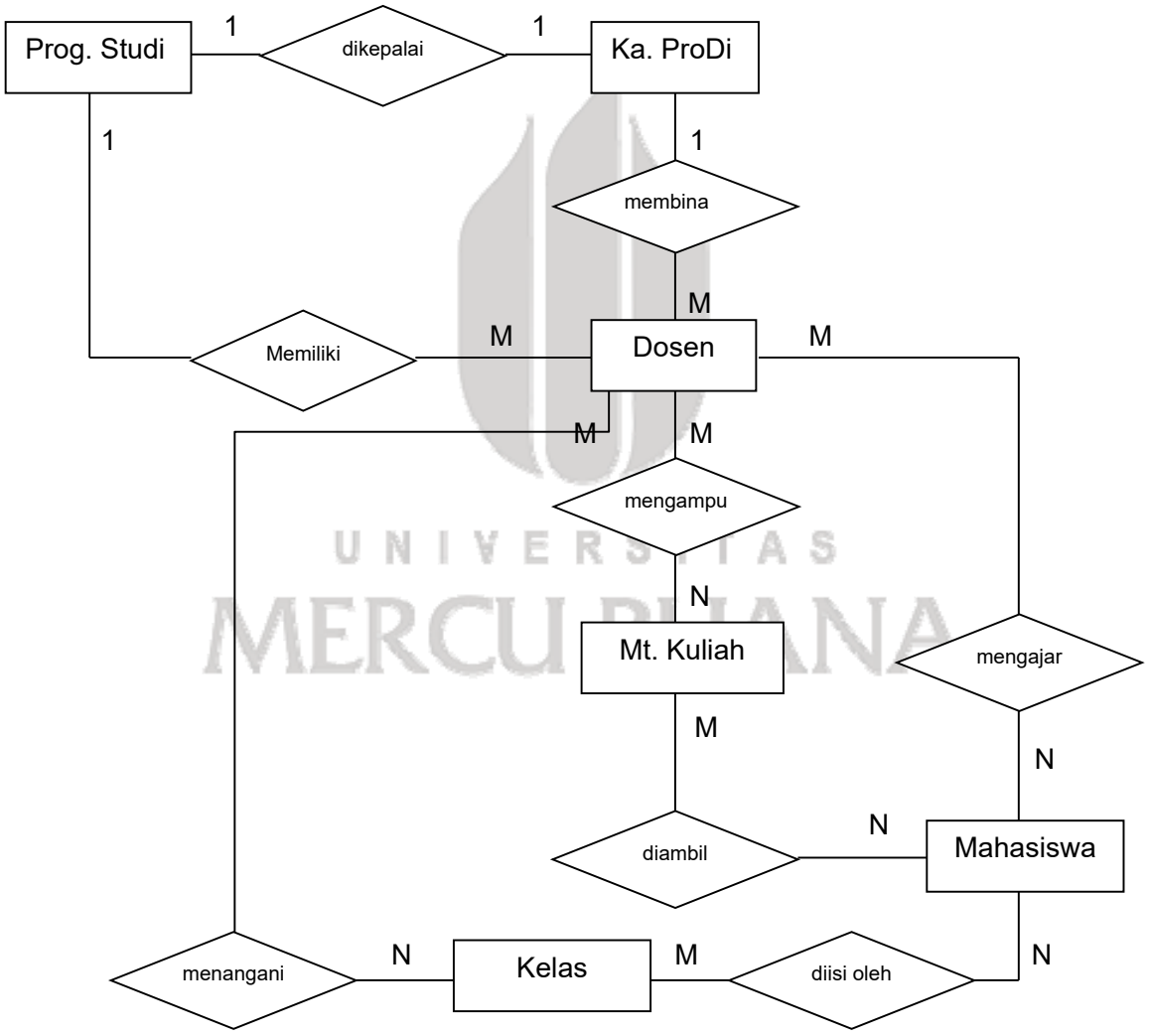


MEMBANGUN ERD

1. Level 1—modelkan semua objekdata (entiti) dan koneksinya dengan yang lain
2. Level 2—modelkan semua entitas dan hubungan
3. Level 3—modelkan semua entitas, hubungan dan atribut yang disiapkan lebih jauh

Contoh :

Program Studi Sistem Informasi dikepalai oleh seorang **Ketua Program Studi** yang membina beberapa **Dosen**. Satu Dosen dapat mengajar satu atau lebih **Mata Kuliah**, namun satu dosen tadi bisa saja mengajar lebih dari 1 kelas. 1 kelas diisi oleh banyak **Mahasiswa**, dimana mahasiswa tsb bisa saja mengambil beberapa mata kuliah

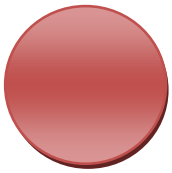


MEMBUAT MODEL ALIRAN (FLOW MODEL)

Setiap system berbasis computer adalah transformasi informasi ……

NOTASI YANG DIGUNAKAN DALAM FLOW MODEL

EKSTERNAL ENTITI



PROSES

ALIRAN DATA

DATA STORE

EKSTERNAL ENTITI

- Adalah penghasil atau pengguna data
- Data harus selalu berasal dari suatu / beberapa tempat
- Dan harus selalu dikirim / disampaikan ke tempat lain
- Objek yang tertulis didalamnya harus kata benda
- Bila diwakili oleh departemen / orang, harus memperlihatkan jabatan (Direktur, Kepala Gudang, Pelanggan, dsb)

PROSES

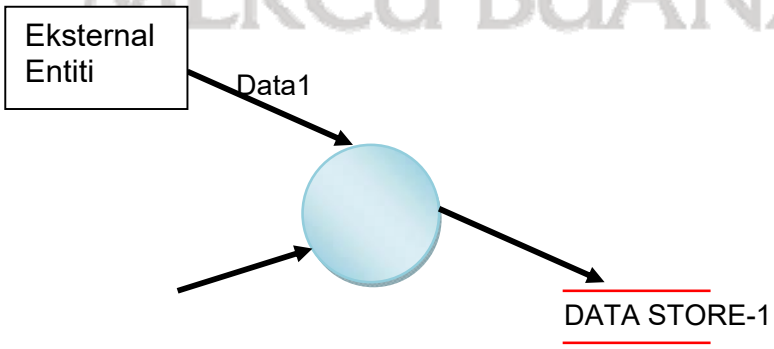
- Berfungsi merubah data (input menjadi output)
- Data harus selalu diproses dalam beberapa cara untuk mencapai fungsi system
- Nama proses MINIMAL terdiri dari gabungan dua kata, yaitu gabungan kata kerja dan kata benda (misal Hitung Gaji, Cetak Laporan)
- TIDAK BOLEH menggunakan kata-kata PROSES

DATA FLOW

- Data flows menunjukkan arah mengalirnya data / informasi melalui system dan data apa yang dibawa
- Sebagai penunjuk dari mana data tsb berasal dan ke arah mana data / data yang telah diolah (informasi) nya dituju
- Nama aliran data HARUS KATA BENDA
- TIDAK BOLEH menggunakan kata-kata DATA atau INFORMASI

DATA STORE

- Sebagai tempat penyimpanan data yang akan digunakan untuk keperluan selanjutnya
- Nama harus sesuai content



PANDUAN UNTUK (MENGGAMBARKAN) DATA FLOW DIAGRAM

- Semua ikon harus diberi label dengan nama yang mempunyai arti kecuali koneksi antara proses dan data store

- DFD berkembang menjadi beberapa tingkatan yang lebih rinci
- Selalu dimulai dengan diagram konteks
- Selalu memperlihatkan eksternal entity
- Selalu ada label data flow yang ditunjukkan dengan panah
- Tidak menggambarkan prosedur logika

KONEKSI YANG DIBOLEHKAN

- Koneksi antara proses dengan proses (namun tidak dianjurkan)
- Koneksi antara entity dengan proses
- Koneksi antara proses dengan data store

KONEKSI YANG TIDAK DIBOLEHKAN

- Koneksi antara entity dengan entity
- Koneksi antara data store dengan data store
- Koneksi antara data store dengan entity

CONTOH KASUS DFD

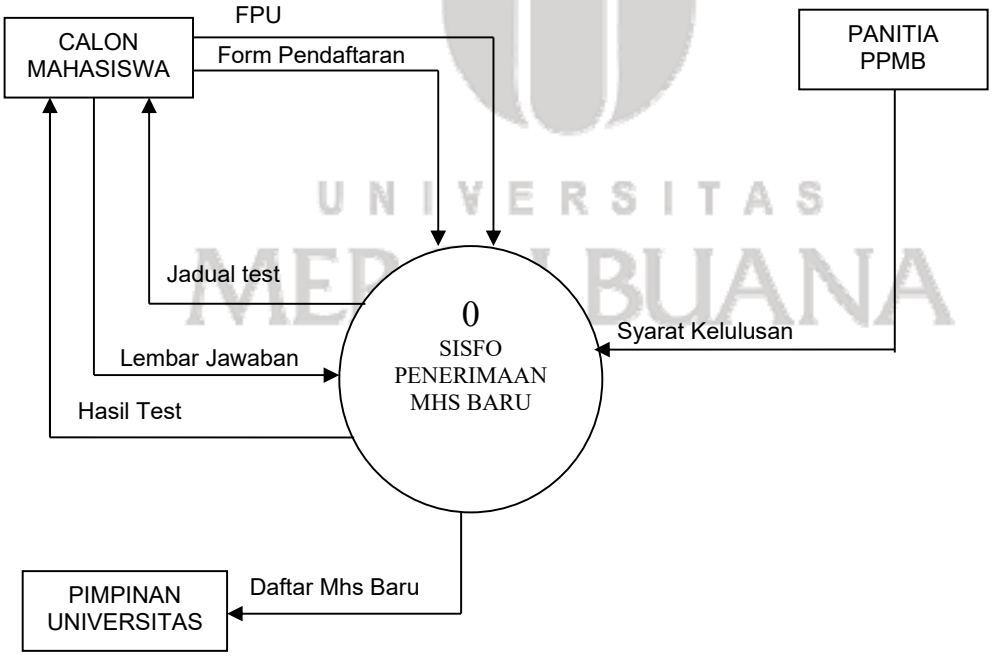


DIAGRAM KONTEKS (LEVEL 0) SISFO PENERIMAAN MAHASISWA BARU

CONTOH DFD

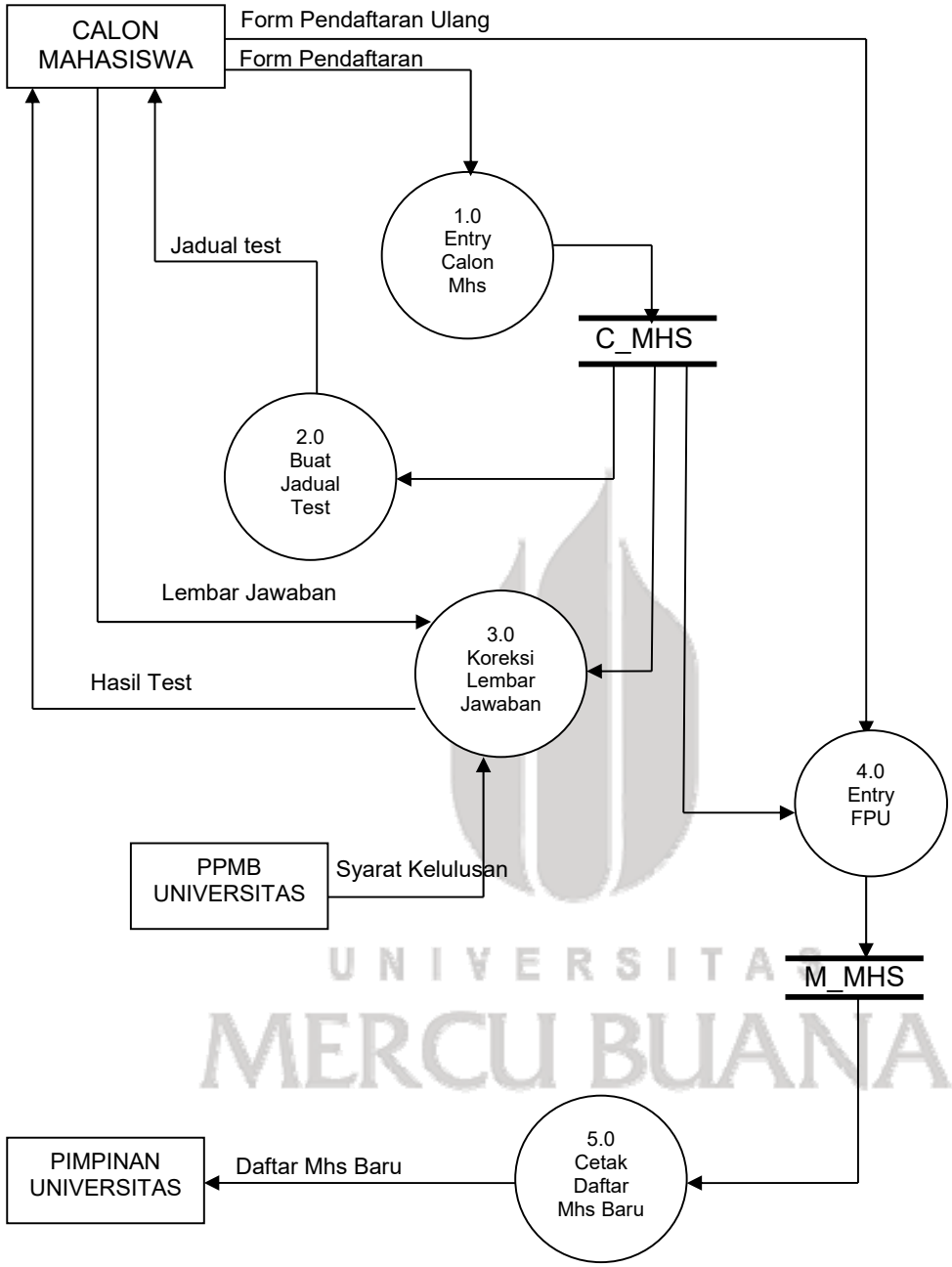


DIAGRAM NOL (LEVEL 1) SISFO PENERIMAAN MAHASISWA BARU

PANDUAN LAIN YANG HARUS DIPERHATIKAN

- Keseimbangan aliran data dan DFD harus diperhatikan dalam tingkatan lanjutannya
- Membangun level 1 DFD akan lebih banyak membantu
- Gunakan rasio maksimum 1:5 (kira-kira) dalam pengembangan setiap proses
- Setiap proses di uraikan sampai masing-masingnya hanya mengerjakan 1 tugas

UNIVERSITAS
MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Konsep Dan Prinsip Desain

Fakultas	Program Studi	TatapMuka	Kode MK	DisusunOleh
Ilmu Komputer	Informatika	13		Tim Dosen

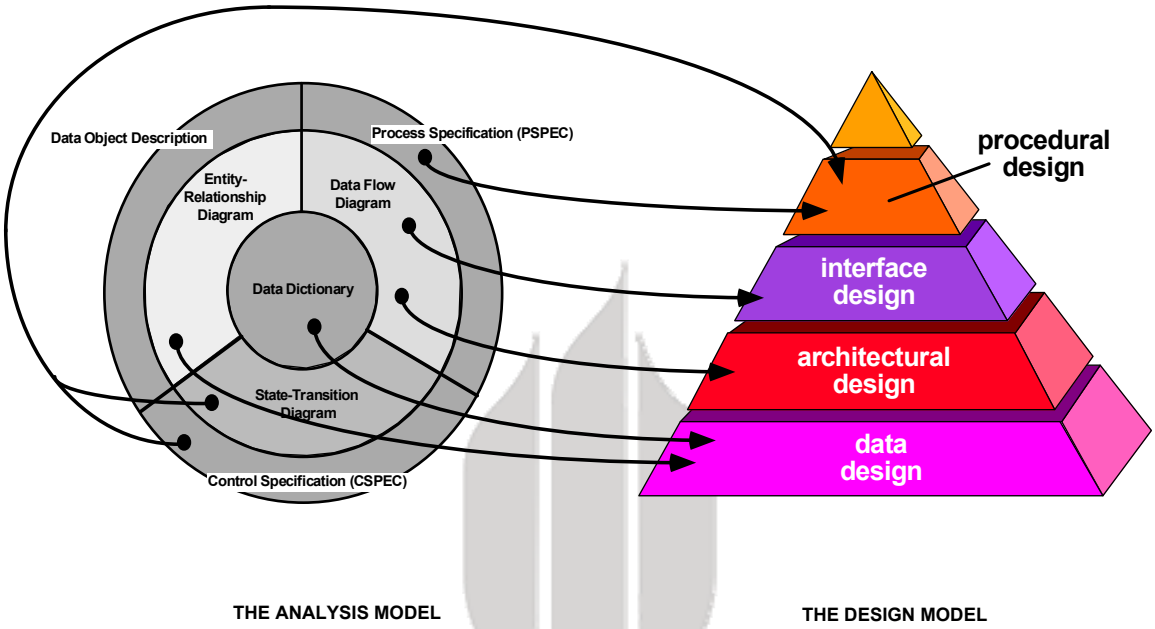
Abstract

Kompetensi

Menjelaskan proses dan unsusr-unsur dalam mengembangkan atau membangun sebuah sistem

KONSEP DAN PRINSIP DESAIN
(DESIGN CONCEPTS AND PRINCIPLES)

Aliran Informasi Selama Desain Perangkat Lunak



Desain Perangkat Lunak dan Rekayasa Perangkat Lunak

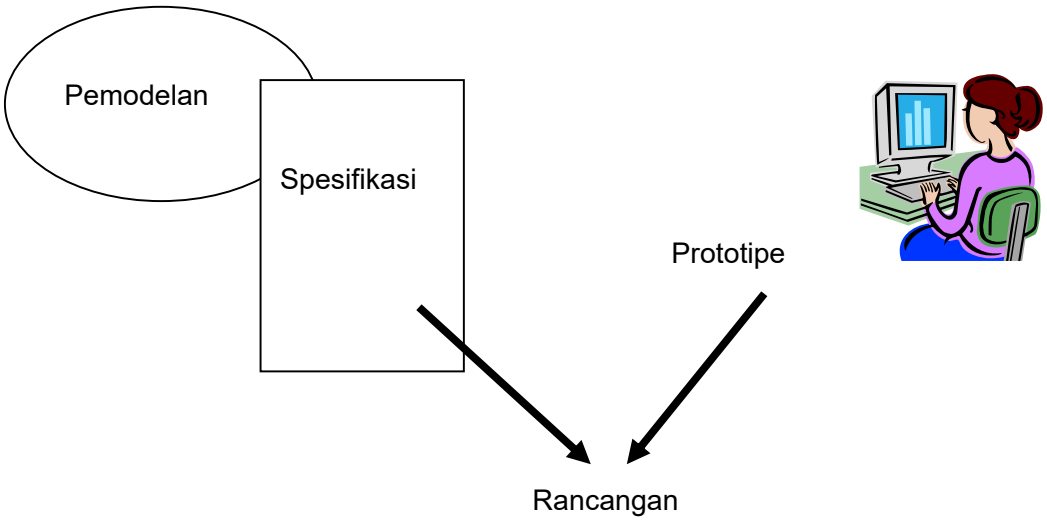
Desain Data mentransformasi model domain informasi yang dibuat selama analisis ke dalam struktur data yang akan diperlukan untuk mengimplementasikan perangkat lunak

Desain Arsitektur menentukan hubungan diantara elemen-elemen struktur utama dari program

Desain Interface menggambarkan bagaimana perangkat lunak berkomunikasi dalam dirinya sendiri dengan system yang berinteroperasi dengannya dan dengan manusia yang menggunakannya

Desain Prosedural mentransformasi elemen-elemen structural dari arsitektur program ke dalam suatu deskripsi procedural dari komponen-komponen perangkat lunak.

Dimana Kita Harus Memulai ?



PRINSIP-PRINSIP RANCANGAN

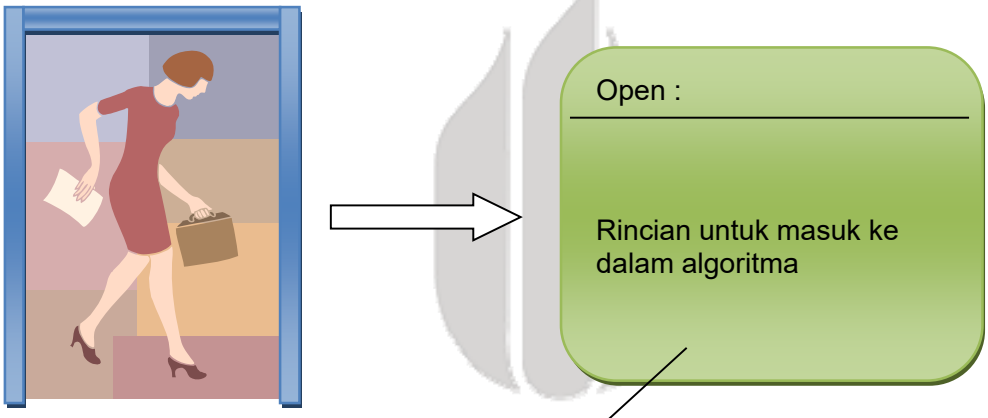
- The design process should not suffer from 'tunnel vision.' ➔ harus ada pendekatan-pendekatan alternative dan menilai masing-masing pendekatan berdasarkan persyaratan masalah.
- The design should be traceable to the analysis model.
- The design should not reinvent (menciptakan kembali) the wheel. ➔ tidak boleh berulang
- The design should "minimize the intellectual distance"(meminimalkan kesenjangan intelektual) [DAV95] between the software and the problem as it exists in the real world.
- The design should exhibit uniformity (memperlihatkan kesatuan) and integration.
- The design should be structured to accommodate change. (terstruktur untuk mengakomodasi perubahan)
- The design should be structured to degrade gently, even when aberrant (menyimpang dari kebiasaan) data, events, or operating conditions are encountered.
- Design is not coding, coding is not design.
- The design should be assessed (diperkirakan / ditaksir) for quality as it is being created, not after the fact.
- The design should be reviewed to minimize conceptual (semantic) errors.

KONSEP YANG MENDASAR

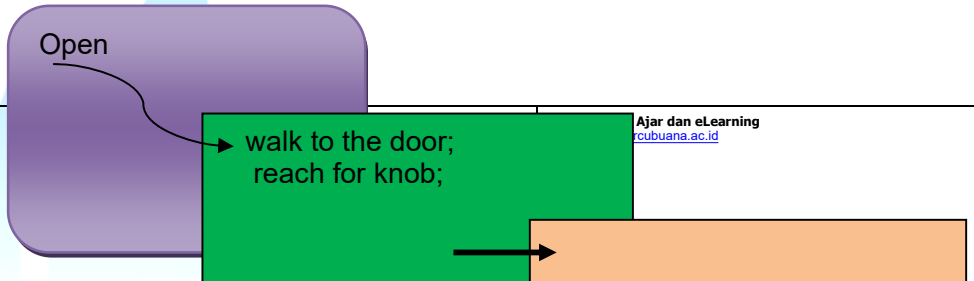
- Abstraction—data, procedure, control

- Refinement (penyaringan) —elaboration of detail for all abstractions
- Modularity— atribut tunggal dari perangkat lunak yang memungkinkan sebuah program untuk dikelola secara intelektual) compartmentalization of data and function
- Architecture—overall structure of the software
  - Structural properties
  - Extra-structural properties
  - Styles and patterns
- Procedure—the algorithms that achieve function
- Hiding—controlled interfaces

GAMBARAN PROSEDURAL

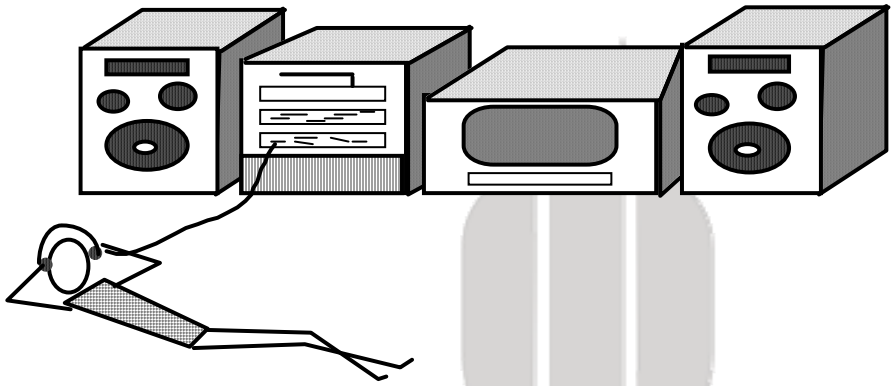


LANGKAH-LANGKAH



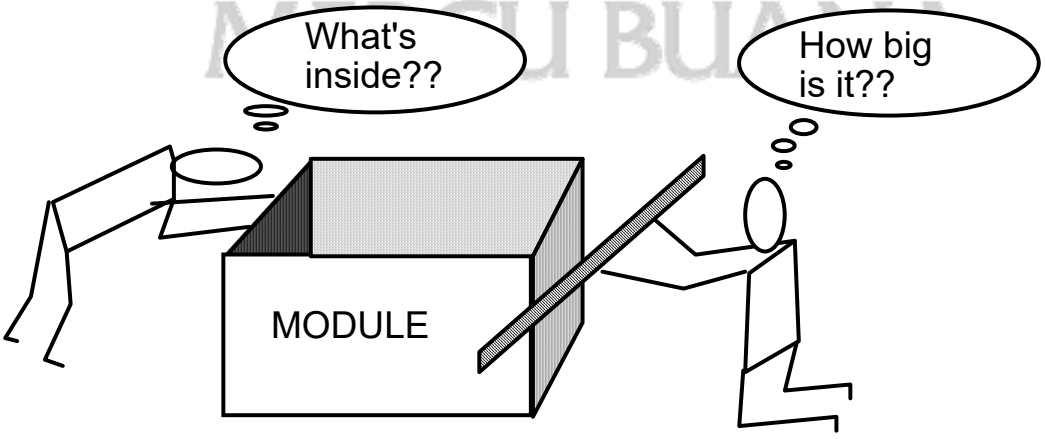
RANCANGAN MODULER

easier to build, easier to change, easier to fix ...



UKURAN MODUL

Ada 2 sudut pandang terhadap hal ini :



FUNGSIONAL INDEPENDEN

COHESION - the degree to which a module performs one and only one function.

COUPLING - the degree to which a module is "connected" to other modules in the system.

ARSITEKTUR

"The overall structure of the software and the ways in which that structure provides conceptual integrity for a system." (struktur keseluruhan perangkat lunak dan cara dimana struktur memberikan integrasi konseptual bagi suatu system) [SHA95a]

Structural properties. This aspect of the architectural design representation defines the components of a system (e.g., modules, objects, filters) and the manner in which those components are packaged and interact with one another. For example, objects are packaged to encapsulate both data and the processing that manipulates the data and interact via the invocation of methods .

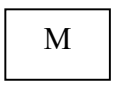
Extra-functional properties. The architectural design description should address how the design architecture achieves requirements for performance, capacity, reliability, security, adaptability, and other system characteristics.

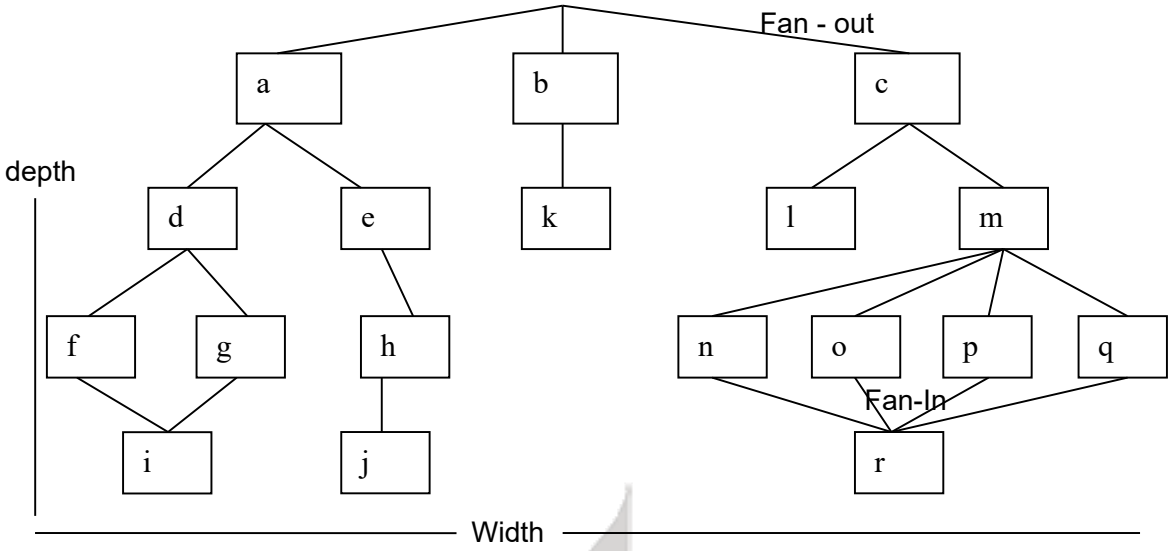
Families of related systems. The architectural design should draw upon repeatable patterns that are commonly encountered in the design of families of similar systems. In essence, the design should have the ability to reuse architectural building blocks (polanya dapat diulangi yang umumnya ditentukan dalam desain dari keluarga system yang sama)

HIRARKI KONTROL

- Represents the organization of program components & implies a hierarchy of control (merepresentasikan organisasi komponen program dan mengimplikasikan suatu hirarki control)
- Tapi tidak merepresentasikan (Does not represent) :
  - Procedural aspect of SW such as sequence, order, repetition (aspek prosedur dari perangkat lunak seperti urutan proses, kejadian / urutan dari suatu keputusan dan pengulangan operasi)
  - Applicability to all architectural styles

REPRESENTASI – SEPERTI DIAGRAM POHON





CATATAN UNTUK DIAGRAM DI ATAS

- Depth and width provide an indication of the number of levels of control and overall span of control, respectively
- Fan-out is a measure of the number of modules that are directly controlled by another module
- Fan-in indicates how many modules directly control a given modules
- A module that control another module is said to be super-ordinate to it
- A module controlled by another is said to be subordinate to the controller
- Eg. M is super-ordinate to a, b & c
- Eg. h is subordinate to e & ultimately to M

PEMBAGIAN STRUKTUR

- Pembagian secara Horizontal
- Pembagian secara Vertical (factoring)

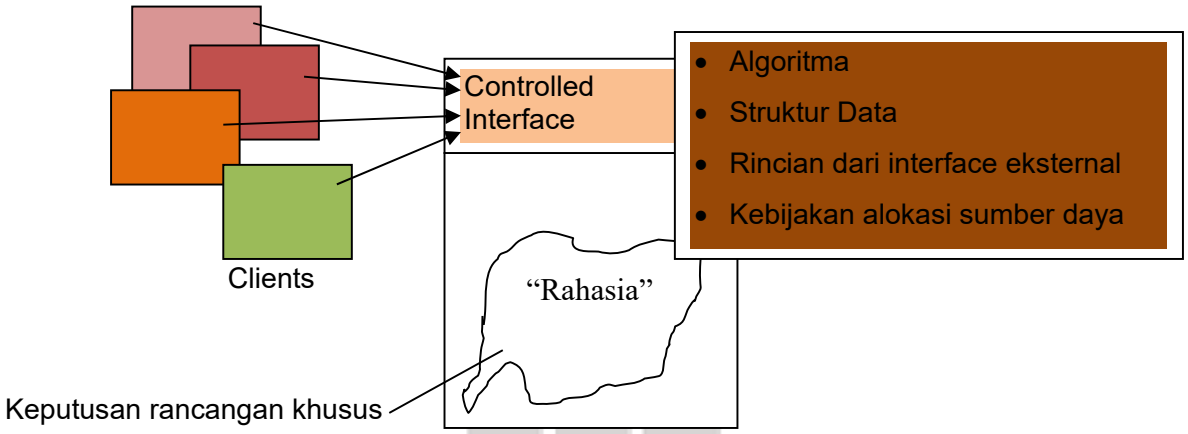
STRUKTUR DATA

- A representation of the logical relationship among individual elements of data

1. Scalar item
2. Sequential vector
3. N-dimensional space (array)
4. Linked list

- Hierarchical data structure – multi linked list
- Different level of abstraction, eg. Stack

INFORMATION HIDING



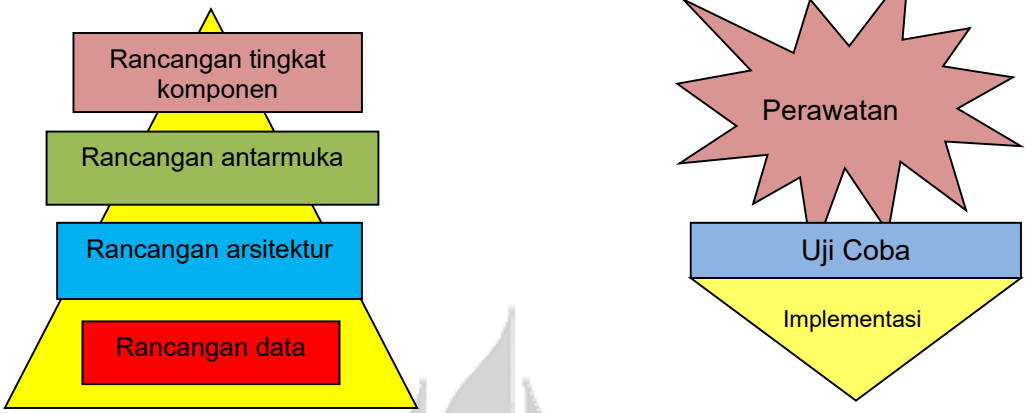
MENGAPA (PERLU) INFORMATION HIDING ?

- reduces the likelihood of "side effects"
- limits the global impact of local design decisions
- emphasizes communication through controlled interfaces
- discourages the use of global data
- leads to encapsulation—an attribute of high quality design
- results in higher quality software

DESIGN HEURISTICS

- Reduce coupling, improve cohesion
- Minimize fan-out, strive for fan-in
- Keep the scope of effect within the scope of control
- Evaluate module interface to reduce complexity and redundancy and improve consistency
- Function is predictable, but not overly restrictive
- Strive for "controlled entry" modules by avoiding "pathological connection"

MODEL RANCANGAN



DOKUMENTASI RANCANGAN

- Scope of design effort, yang mencakup :
  - Sasaran system
  - Persyaratan utama perangkat lunak
  - Batasan-batasan dan pembatasan desain
- Data design, yang mencakup :
  - Objek data dan struktur data resultan
  - Struktur file dan database
    - ✓ Struktur file eksternal
      a. Struktur logis
      b. Deskripsi record logis
      c. Metode akses
    - ✓ Data global
    - ✓ File dan referensi lintas data
- Architectural design, yang mencakup :
  - Kajian data dan aliran control

- Struktur program yang diperoleh

- Interfaces, yang meliputi :
  - Spesifikasi antarmuka manusia - mesin
  - Aturan desain antarmuka manusia-mesin
  - Desain antarmuka eksternal
    - ✓ Interface untuk data eksternal
    - ✓ Interface untuk system atau peralatan eksternal
- Components
  - Narasi
  - Deskripsi
  - Bahasa
  - Modul yang digunakan
  - Struktur data internal
  - Ketentuan / larangan
- Cross reference
- Test plan yang meliputi :
  - Panduan pengujian
  - Strategi integrasi
  - Pertimbangan khusus
- constraints
- Supplementary data

UNIVERSITAS MERCU BUANA

MODUL PERKULIAHAN

Rekayasa Perangkat Lunak

Pemeliharaan Perangkat Lunak

Fakultas	Program Studi	Tatap Muka	Kode MK	Disusun Oleh
Ilmu Komputer	Informatika	14		Tim Dosen

Abstract

Pertemuan 14 Rekayasa Perangkat Lunak Konsep dan Teknik Pemeliharaan Perangkat Lunak.

Kompetensi

Setelah mempelajari pertemuan ini mahasiswa akan dapat memahami Konsep dan Teknik Pemeliharaan Perangkat Lunak.

Meskipun bug ini hanya gangguan kecil, tetapi kadang-2 bisa berakibat fatal terhadap sistem, sehingga harus diantisipasi sedini mungkin, oleh karena itu proses ini termasuk dlm tahap pemeliharan sistem.

3. Penyempurnaan (Upgrading / Enhancement)

Suatu sistem setelah berjalan beberapa waktu biasanya membutuhkan penyempurnaan karena berbagai alasan, seperti perkembangan bisnis, perubahan organisasi, perubahan kebijakan manajemen, dsb. Penyempurnaan ini bisa berupa penyempurnaan tampilan maupun penyempurnaan isi, baik untuk laporan, format isian, maupun proses pengolahan data.

Penyempurnaan sistem ini penting, karena jika tidak dilakukan dapat merugikan organisasi.

4. Antisipasi Faktor-faktor di Luar Aplikasi

Walaupun suatu sistem telah dapat berfungsi sebagaimana yg diharapkan, kita masih harus mengantisipasi terhadap faktor-faktor diluar aplikasi, yaitu:

- Virus
- Kerusakan/kehilangan data
- Sistem diakses oleh user yg tidak berhak
- Dll.

Memelihara Perangkat Keras

Pemeliharaan perangkat keras terutama pemeliharaan preventif yang memerlukan reparasi, penggantian, atau penambahan suku cadang dan komponen untuk merestorasi atau menjaga agar perangkat keras tetap bekerja dengan baik. Komponen perangkat keras sistem informasi sebaiknya dicek dan diservis secara periodik.

Memelihara Perangkat Lunak

Perangkat lunak aplikasi mungkin terstruktur mungkin pula tidak, atau mungkin didokumentasikan mungkin pula tidak. Beberapa perangkat lunak yang tidak terstruktur dan tidak didokumentasikan mungkin hampir tidak dapat dipelihara.

1. Bahasa Pemrograman Standar.

Penggunaan bahasa pemrograman standar, misalnya C atau COBOL, akan mempermudah pekerjaan pemeliharaan.

Jika perangkat lunak C atau COBOL berisi dokumentasi internal yang jelas dan lengkap, seorang programmer pemeliharaan pemula atau pemakai dapat memahami apa yang sedang dikerjakannya. Lagipula C dan COBOL adalah bahasa Universal yang umumnya diketahui oleh sejumlah besar orang. Dengan demikian penggantian programmer pemeliharaan tidak begitu berdampak negatif pada kemampuan perusahaan untuk memelihara program C atau COBOL lama.

2. ***Rancangan Moduler.***
   Programmer pemeliharaan dapat mengganti modul program jauh lebih mudah daripada jika ia berurusan dengan keseluruhan program.

3. ***Modul yang Dapat Digunakan Kembali.***
   Modul biasa dari kode yang dapat digunakan kembali, dapat diakses oleh semua aplikasi yang memerlukannya.

4. ***Dokumentasi Standar.***
   Diperlukan sistem, pemakai, perangkat lunak dan dokumentasi operasi yang standar sehingga semua informasi yang diperlukan untuk beroperasi dan pemeliharaan aplikasi khusus akan tersedia.

5. ***Kontrol Sentral.***
   Semua program, dokumentasi, dan data tes seharusnya diinstal dalam penyimpanan pusat dari sistem CASE (*Computer-Aided Software Engineering* atau *Computer-Assisted Software Engineering*).