Modul Praktikum

Rekayasa Perangkat Lunak

Di Susun Oleh :

Muhammad Fahmi

Informatika (S1)

Universitas Nusa Mandiri

2021

PERANGKAT LUNAK DAN REKAYASA PERANGKAT LUNAK

Perangkat Lunak (PL) merujuk pada sekumpulan program komputer, instruksi, dan data yang digunakan untuk melakukan tugas tertentu pada sistem komputer. Perangkat lunak merupakan bagian dari perangkat keras (hardware) yang bertanggung jawab untuk mengatur dan mengendalikan operasi perangkat keras serta menyediakan fungsionalitas yang dibutuhkan oleh pengguna.

Secara umum, perangkat lunak dapat dibagi menjadi dua kategori utama:

1. **Perangkat Lunak Sistem (System Software):** Ini adalah perangkat lunak yang bertanggung jawab untuk mengelola sumber daya perangkat keras dan menyediakan layanan dasar bagi perangkat lunak aplikasi. Contoh perangkat lunak sistem meliputi sistem operasi (seperti Windows, macOS, Linux), driver perangkat keras, dan utilitas sistem.
2. **Perangkat Lunak Aplikasi (Application Software):** Ini adalah perangkat lunak yang dirancang untuk memberikan solusi atau layanan spesifik kepada pengguna akhir. Perangkat lunak aplikasi dapat mencakup berbagai jenis, seperti perangkat lunak produktivitas (Microsoft Office, Adobe Photoshop), permainan, perangkat lunak keamanan, dan banyak lagi.

Dengan kata lain, perangkat lunak adalah bagian dari sistem komputer yang tidak terlihat secara fisik tetapi memberikan instruksi dan fungsi yang memungkinkan komputer menjalankan berbagai tugas dan aplikasi sesuai kebutuhan pengguna.

Karakteristik perangkat lunak mencakup berbagai atribut dan sifat yang menggambarkan ciri khas atau kualitas dari suatu program komputer. Berikut adalah beberapa karakteristik umum dari perangkat lunak: Perangkat lunak harus dapat diubah atau disesuaikan dengan kebutuhan yang berubah dari waktu ke waktu. Kemampuan untuk memodifikasi atau menyesuaikan perangkat lunak adalah penting agar dapat mengatasi perubahan dalam lingkungan atau persyaratan bisnis. Perangkat lunak sebaiknya dapat dijalankan di berbagai platform atau sistem operasi tanpa memerlukan banyak modifikasi. Portabilitas mempermudah perpindahan perangkat lunak dari satu lingkungan ke lingkungan lainnya. Kode sumber perangkat lunak harus mudah dibaca dan dimengerti oleh pengembang lain yang mungkin terlibat dalam pemeliharaan atau pengembangan lebih lanjut. Keterbacaan kode membantu dalam kolaborasi tim pengembang. Perangkat lunak sebaiknya dirancang untuk menggunakan sumber daya komputer (memori, CPU, jaringan, dll.) secara efisien. Efisiensi ini dapat berdampak pada kinerja dan responsivitas aplikasi. Perlindungan terhadap ancaman keamanan seperti akses tidak sah, serangan malware, dan kerentanan keamanan lainnya adalah karakteristik kritis dari perangkat lunak modern. Perangkat lunak harus memiliki lapisan keamanan yang memadai. Perangkat lunak seharusnya mudah diinstal, dikonfigurasi, dan dikelola. Kemudahan penggunaan dan administrasi dapat membantu pengguna atau administrator sistem dalam memanfaatkan perangkat lunak dengan efektif. Perangkat lunak harus dapat terintegrasi dengan perangkat lunak lainnya atau sistem yang sudah ada. Ini memungkinkan interaksi yang lancar antara berbagai komponen perangkat lunak. Perangkat lunak seharusnya dapat berkembang dan menangani beban kerja yang semakin besar atau lebih banyak pengguna tanpa mengalami penurunan kinerja yang signifikan. Perangkat lunak sebaiknya didokumentasikan dengan baik untuk memberikan informasi yang jelas tentang fungsionalitas, penggunaan, dan pemeliharaan. Dokumentasi yang baik dapat membantu pengguna dan pengembang. Perangkat lunak seharusnya dapat berjalan secara konsisten dan dapat diandalkan dalam berbagai kondisi. Kehandalan menjadi kunci, terutama dalam aplikasi yang memerlukan ketersediaan tinggi.

Karakteristik ini bersifat umum dan dapat berbeda tergantung pada jenis perangkat lunak dan kebutuhan bisnis atau penggunaannya.

A. Kategori Perangkat Lunak

PL Sistem (System Software)

PL Aplikasi (Application Software)

PL Rekayasa/Ilmiah (Engineering/Scientific Software)

PL yang tertanam (Embedded Software)

PL Lini Produk (Product-Line Software)

PL Aplikasi Web (Web/Mobile Applications)

PL Kecerdasan Buatan (Artificial Intelligence Software)

B. Jenis Perangkat Lunak Aplikasi

a. Stand-Alone Applications adalah contoh aplikasi seperti aplikasi office pada PC, program CAD, software manipulasi foto, dll

b. Interactive Transaction-Based Aapplications adalah aplikasi yang mengeksekusi pada computer remote dan yang diakses oleh pengguna dari PC mereka sendiri atau terminal

c. Batch Processing Systems adalah sistem bisnis yang dirancang untuk memproses data input yang besar untuk membuat output yang sesuai. Contoh: sistem penagihan telepon, dan system pembayaran gaji

d. Embedded Control Systems adalah sistem kontrol PL yang mengontrol dan mengelola perangkat keras, atau sistem yang tertanam pada jenis sistem lain. Contoh: PL yang mengontrol pengereman anti-lock mobil, dan software dalam oven microwave untuk mengontrol proses memasak.

e. Entertainment Systems adalah sistem yang terutama untuk penggunaan pribadi dan yang dimaksudkan untuk menghibur pengguna.

f. Systems for Modelling and Simulation adalah sistem yang dikembangkan untuk model proses fisik atau situasi, dengan banyak objek yang saling berinteraksi

g. Data Collection Systems adalah sistem yang mengumpulkan data dari lingkungan mereka menggunakan satu set sensor dan mengirim data ke sistem lain untuk diproses.

h. Systems of Systems adalah sistem yang terdiri dari sejumlah sistem PL lain.

Perangkat Lunak Warisan (Legacy Software) mengacu pada perangkat lunak yang telah ada dalam pengembangan atau digunakan dalam suatu organisasi atau sistem selama periode waktu yang lama. Perangkat lunak ini biasanya sudah berjalan dan mungkin telah melewati beberapa pembaruan, namun sering kali dirancang menggunakan teknologi, bahasa pemrograman, atau paradigma pengembangan yang sudah usang.

Beberapa ciri dari perangkat lunak warisan meliputi:

1. Perangkat lunak warisan mungkin tidak dapat mengadopsi teknologi terkini dengan baik. Ini bisa menjadi tantangan ketika organisasi ingin memperbarui atau memodernisasi sistem mereka.
2. Perangkat lunak warisan mungkin tergantung pada platform perangkat keras atau sistem operasi tertentu yang sudah tidak lagi didukung atau kurang relevan.
3. Kode sumber dari perangkat lunak warisan sering kali kompleks dan sulit dimengerti. Ini dapat menyulitkan pengembang dalam memahami, memodifikasi, atau memperbaiki bagian-bagian tertentu dari perangkat lunak.
4. Perangkat lunak warisan mungkin tidak dapat menyediakan fitur-fitur terbaru atau memenuhi kebutuhan yang berkembang karena keterbatasan arsitektur atau desain aslinya.

5. Perangkat lunak warisan mungkin memiliki keterbatasan dalam hal skalabilitas, yaitu kemampuannya untuk menangani peningkatan beban atau kebutuhan yang lebih besar.
6. Perangkat lunak yang sudah usang cenderung memiliki risiko keamanan yang lebih tinggi karena mungkin tidak menerima pembaruan keamanan terbaru atau karena kelemahan keamanan dalam desain aslinya.
7. Perangkat lunak warisan sering kali memerlukan pemeliharaan tambahan, dan kesulitan dalam menemukan orang yang memahami teknologi kuno tersebut dapat menjadi masalah.

Meskipun perangkat lunak warisan memiliki beberapa tantangan, banyak organisasi yang masih bergantung padanya karena dapat melibatkan investasi besar untuk melakukan migrasi atau modernisasi. Pengelolaan dan pemeliharaan perangkat lunak warisan dapat melibatkan strategi yang hati-hati, termasuk pembaruan kecil, migrasi bertahap, atau bahkan penggantian perlahan dengan solusi yang lebih modern.

Kegagalan Perangkat Lunak

Faktor-faktor penyebab kegagalan PL:

1. Meningkatnya tuntutan RPL membangun sistem yang lebih besar, sistem yang lebih kompleks menyebabkan tuntutan berubah. Sistem harus dibangun dan disampaikan lebih cepat, lebih besar, dan lebih kompleks. Sistem harus memiliki kemampuan baru yang sebelumnya dianggap mustahil.
2. Harapan yang rendah Hal ini relatif mudah untuk menulis program komputer tanpa menggunakan metode dan teknik RPL. Banyak Pengusaha yang tidak menggunakan metode RPL, akibatnya PL lebih mahal dan kurang dapat diandalkan.

Proses perangkat lunak merujuk pada serangkaian langkah atau aktivitas yang dijalankan dalam pengembangan perangkat lunak, dari perencanaan awal hingga pengiriman produk perangkat lunak yang siap digunakan. Proses ini membantu mengorganisir dan mengelola proyek pengembangan perangkat lunak secara efektif. Berikut adalah beberapa tahapan umum dalam proses pengembangan perangkat lunak: Tahap ini melibatkan penentuan tujuan proyek, estimasi biaya dan waktu, serta identifikasi sumber daya yang diperlukan. Perencanaan ini mencakup analisis kebutuhan pengguna, penjadwalan proyek, dan penentuan metrik keberhasilan. Dalam tahap ini, kebutuhan perangkat lunak dianalisis dan dicatat. Tim pengembang berusaha memahami kebutuhan pengguna, menciptakan spesifikasi fungsional, dan merancang arsitektur dasar sistem. Desain perangkat lunak mencakup pembuatan rancangan terperinci dari sistem berdasarkan spesifikasi yang telah ditetapkan. Desain ini mencakup struktur data, arsitektur perangkat lunak, dan perencanaan implementasi. Pada tahap ini, pengembang mulai mengubah rancangan desain menjadi kode sumber yang dapat dijalankan oleh komputer. Proses implementasi mencakup pemrograman, pengujian unit, dan integrasi komponen. Pengujian adalah tahap di mana perangkat lunak diuji untuk memastikan bahwa itu beroperasi sesuai dengan spesifikasi. Pengujian dapat melibatkan pengujian fungsional, pengujian kinerja, dan pengujian keamanan. Setelah berhasil melewati pengujian, perangkat lunak diimplementasikan dan siap digunakan oleh pengguna akhir. Pemeliharaan melibatkan perbaikan bug, pembaruan, dan peningkatan sesuai kebutuhan. Setelah implementasi, evaluasi dilakukan untuk mengevaluasi keberhasilan proyek, memastikan bahwa kebutuhan pengguna telah terpenuhi, dan mengevaluasi proses pengembangan untuk pembelajaran di masa depan.

Tahapan-tahapan ini sering kali dijalankan secara iteratif atau secara kombinasi, tergantung pada metodologi pengembangan perangkat lunak yang digunakan, seperti model pengembangan air terjun, model prototipe, atau model pengembangan iteratif dan inkremental. Proses pengembangan perangkat lunak bertujuan untuk menghasilkan perangkat lunak berkualitas tinggi yang memenuhi kebutuhan dan ekspektasi pengguna.

Terdapat beberapa model proses pengembangan perangkat lunak yang digunakan dalam industri perangkat lunak. Setiap model memiliki pendekatan unik terhadap siklus pengembangan perangkat lunak. Berikut adalah beberapa model proses pengembangan perangkat lunak yang umum digunakan:

Model Pengembangan Air Terjun (Waterfall Model):

Proses ini terdiri dari serangkaian tahapan linier, dimulai dari perencanaan, analisis, desain, implementasi, pengujian, hingga pemeliharaan. Setiap tahap harus selesai sebelum memulai tahap berikutnya.

Model Prototipe (Prototype Model):

Proses ini melibatkan pembuatan prototipe atau model awal dari perangkat lunak yang digunakan untuk memahami kebutuhan pengguna. Prototipe tersebut kemudian dianalisis, dan pengembangan dilanjutkan dengan menggabungkan umpan balik dari pengguna.

Model Spiral:

Model ini menggabungkan unsur-unsur dari model pengembangan air terjun dan model prototipe. Pengembangan dilakukan secara iteratif dan inkremental, dengan setiap iterasi menambah fungsionalitas baru.

Model Incremental:

Pengembangan dilakukan dalam tahap-tahap kecil atau increment, di mana setiap increment menambah fungsionalitas baru. Setiap increment diimplementasikan dan diuji secara terpisah.

Model Rapid Application Development (RAD):

RAD adalah pendekatan yang cepat dan fleksibel, dengan penekanan pada penggunaan prototipe dan pengembangan cepat. Proses ini fokus pada penyampaian produk perangkat lunak yang berfungsi dalam waktu singkat.

Model Iteratif dan Inkremental (Iterative and Incremental Model):

Model ini menggabungkan konsep pengembangan iteratif dan inkremental. Pengembangan dilakukan dalam siklus iteratif, dan fungsionalitas baru ditambahkan secara bertahap dalam setiap iterasi.

Model Agile:

Model ini menekankan kolaborasi tim, tanggapan terhadap perubahan, dan pengiriman perangkat lunak secara iteratif dan inkremental. Proses agile mencakup praktik-praktik seperti Scrum, Kanban, dan Extreme Programming (XP).

Model DevOps:

DevOps mengintegrasikan pengembangan (development) dan operasi (operations) dalam satu siklus hidup pengembangan perangkat lunak. Ini bertujuan untuk meningkatkan kerjasama tim, percepatan rilis perangkat lunak, dan meningkatkan kualitas pengembangan.

Tahapan Model Waterfall

1. Requirements Analysis and Definition Langkah ini merupakan analisa kebutuhan sistem. Berisi layanan-layanan sistem, kendala, dan tujuan yang ditetapkan melalui konsultasi dengan pengguna sistem, kemudian didefinisikan secara rinci yang berfungsi sebagai spesifikasi sistem.
2. System and Software Design Proses design mengalokasikan kebutuhan hardware dan software untuk membangun arsitektur system secara keseluruhan (struktur data, arsitektur PL,

interface, dan detail/algoritma prosedural). Proses design akan menerjemahkan syarat kebutuhan perancangan PL yang dapat diperkirakan sebelum dibuat coding.

Contoh:

Aplikasi web ini dibangun dengan beberapa desain, diantaranya desain database menggunakan ERD dan LRS, desain struktur program menggunakan Struktur Navigasi, dan desain sistem menggunakan UML.

3. Implementation and Unit Testing

Perancangan PL direalisasikan sebagai satu set program atau unit program (coding). Coding merupakan

penerjemahan design dalam bahasa yang bisa dikenali oleh komputer yang dilakukan oleh programmer.

Kemudian dilanjutkan dengan testing terhadap pengujian unit dengan melibatkan verifikasi setiap unit

agar memenuhi spesifikasinya.

Contoh:

Pembuatan code program menggunakan software Dreamweaver CS5, pengolahan database dengan MySQL, script untuk penjelajah web menggunakan Javascript, dst

4. Integration and System Testing

Program-program diintegrasikan dan diuji sebagai sistem yang lengkap untuk memastikan bahwa kebutuhan/persyaratan PL telah dipenuhi. Setelah pengujian, PL sistem dikirim ke pelanggan.

Contoh:

Setelah tahap pembuatan code program dilakukan pengujian program untuk menemukan kesalahan-

kesalahan program. Pengujian program menggunakan black box testing untuk memastikan bahwa seluruh input yang memerlukan verifikasi data sudah benar.

5. Operation and Maintenance

Operasi dan pemeliharaan adalah siklus hidup terlama. Sistem ini dipasang dan digunakan oleh user.

Perawatan melibatkan koreksi kesalahan yang tidak ditemukan sebelumnya, meningkatkan implementasi sistem dan meningkatkan layanan sistem saat persyaratan baru ditemukan.

Contoh:

Pemeliharaan sistem akan terus dilakukan dengan korektif terhadap kesalahan yang mungkin terjadi, adaptif dengan peningkatan layanan sistem, dst

Kelebihan model waterfall:

a. Memiliki proses yang urut dan bertahap, sehingga kualitas sistem/PL yang dihasilkan akan baik.

b. Setiap proses memiliiki spesifikasinya sendiri, karena setiap tahap harus terselesaikan dengan lengkap

sebelum melanjutkan ke tahap berikutnya.

c. Setiap proses tidak dapat saling tumpang tindih.

d. Metode ini akan lebih baik digunakan jika kebutuhan-kebutuhan sudah diketahui.

Kekurangan model waterfall:

a. Proses yang dilakukan cenderung panjang dan lama, karena proses pengembangan tidak dapat dilakukan berulang sebelum menghasilkan produk.

b. Kesalahan kecil pada satu tahapan akan menimbulkan masalah besar jika tidak diketahui sejak awal

pengembangan, berakibat pada tahapan selanjutnya

c. Biaya penggunaan metode yang cenderung mahal

d. Membutuhkan banyak riset dan penelitian pendukung untuk mengembangkan sistem sehingga pelanggan harus sabar karena pembuatan PL baru dimulai pada tahap perancangan.

e. Kenyataannya sulit untuk mengikuti aturan sequential, karena iterasi sulit dilakukan dan menyebabkan masalah baru.

REKAYASA KEBUTUHAN

a. Kebutuhan untuk suatu sistem adalah deskripsi tentang apa yang harus dilakukan oleh sistem berupa layanan yang diberikan dan kendala dalam operasinya.
b. Kebutuhan ini mencerminkan kebutuhan pelanggan untuk sistem yang melayani tujuan tertentu seperti mengontrol perangkat, menempatkan pesanan, atau mencari informasi.
c. Proses mencari tahu, menganalisis, mendokumentasikan serta memeriksa layanan dan kendala ini disebut Rekayasa Kebutuhan (Requirement Engineering).
d. Rekayasa Kebutuhan harus disesuaikan dengan kebutuhan proses, proyek, produk, dan orang-orang yang melakukan pekerjaan.

Jenis Kebutuhan:

1. Kebutuhan pengguna adalah pernyataan, dalam bahasa alami ditambah diagram, dari layanan apa yang diharapkan sistem untuk diberikan kepada pengguna sistem dan kendala di mana ia harus beroperasi.

2. Kebutuhan sistem adalah deskripsi yang lebih rinci tentang fungsi, layanan, dan kendala operasional

sistem perangkat lunak. Dokumen Kebutuhan system (kadang-kadang disebut spesifikasi fungsional) harus mendefinisikan secara tepat apa yang akan diimplementasikan. Ini mungkin menjadi bagian dari

kontrak antara pembeli sistem dan pengembang perangkat lunak.

Kegiatan pada Rekayasa Kebutuhan:

a. Pengenalan Permasalahan (Inception)
b. Pengenalan Lanjutan (Elicitation)
c. Elaborasi (Elaboration)
d. Negosiasi
e. Spesifikasi (Specification)
f. Validasi (Validation)
g. Manajemen Kebutuhan (Requirement Management)

Jenis Kebutuhan:

1. Kebutuhan pengguna adalah pernyataan, dalam bahasa alami ditambah diagram, dari layanan apa yang diharapkan sistem untuk diberikan kepada pengguna sistem dan kendala di mana ia harus beroperasi.
2. Kebutuhan sistem adalah deskripsi yang lebih rinci tentang fungsi, layanan, dan kendala operasional sistem perangkat lunak. Dokumen Kebutuhan system (kadang-kadang disebut spesifikasi fungsional) harus mendefinisikan secara tepat apa yang akan

diimplementasikan. Ini mungkin menjadi bagian dari kontrak antara pembeli sistem dan pengembang perangkat lunak.

Elaborasi (Elaboration)

1. Elaborasi dilakukan dengan cara membuat dan penyempurnaan skenario pengguna yang menggambarkan bagaimana pengguna akhir (dan actor lain) akan berinteraksi dengan sistem.

Negosiasi

1. Konflik yang terjadi antara pelanggan, pengguna dan stakeholder harus didamaikan dengan pendekatan yang bersifat iteratif untuk menentukan skala prioritas kebutuhan, menilai biaya-biaya dan risiko masing- masing.

Spesifikasi (Specification)

1. Spesifikasi kebutuhan adalah proses menuliskan kebutuhan pengguna dan kebutuhan sistem ke dalam dokumen kebutuhan.
2. Spesifikasi dapat berupa dokumen tertulis, model grafis, model matematika formal, kumpulan scenario penggunaan sistem/PL, prototipe, atau kombinasi dari semuanya.
3. Kebutuhan pengguna menggambarkan kebutuhan fungsional dan non-fungsional sehingga dapat dimengerti oleh pengguna sistem (perilaku eksternal dari sistem) yang tidak memiliki pengetahuan teknis.
4. Dokumen kebutuhan tidak boleh menyertakan rincian arsitektur atau desain sistem.

KONSEP PERANCANGAN

Perancangan PL merupakan tempat dimana aturan kreativitas (kebutuhan stakeholder, kebutuhan bisnis, dan pertimbangan teknis) semuanya secara bersamaan disatukan untuk membentuk sebuah produk atau sistem/PL. Perancangan menciptakan representasi atau model PL. Model perancangan memberikan detail tentang arsitektur PL, struktur data, antarmuka, dan komponen yang diperlukan untuk mengimplementasikan sistem. Tujuan perancangan PL adalah untuk menghasilkan model atau representasi PL yang memperlihatkan kekuatan, komoditi, dan kenyamanan.

Model Perancangan

1. Perancangan data/kelas

Mengubah model kelas menjadi realisasi kelas perancangan dan struktur data yang diperlukan untuk mengimplementasikan PL. Objek, hubungan dan konten data rinci yang digambarkan oleh atribut kelas dan notasi lainnya memberikan dasar untuk aktivitas perancangan data. Bagian dari perancangan kelas dapat terjadi bersamaan dengan perancangan arsitektur PL. Perancangan kelas yang lebih rinci terjadi karena setiap komponen PL dirancang.

2. Perancangan arsitektur

Mendefinisikan hubungan antara elemen structural utama dari PL, gaya dan pola arsitektur yang dapat digunakan untuk mencapai kebutuhan yang ditentukan untuk sistem, dan kendala yang mempengaruhi cara dimana arsitektur dapat diimplementasikan. Mewakili perancangan kerangka kerja arsitektur system berbasis komputer berasal dari model kebutuhan.

Model Perancangan

3. Perancangan antarmuka

Menggambarkan bagaimana PL berkomunikasi dengan sistem, dan dengan manusia yang menggunakannya. Antarmuka menyiratkan aliran informasi (misal: data atau kontrol) dan jenis perilaku tertentu. Perancangan antarmuka pada tingkat komponen mengubah elemen struktural dari arsitektur PL menjadi deskripsi prosedural komponen PL. Informasi yang diperoleh dari model berbasis kelas dan model perilaku berfungsi sebagai dasar untuk perancangan komponen.

Atribut-Atribut Kualitas PL

Tiga karakteristik umum yang berfungsi sebagai panduan untuk evaluasi perancangan yang baik:

1. Perancangan PL harus menerapkan semua spesifikasi kebutuhan yang secara eksplisit ada dalam model kebutuhan, dan mengakomodasi semua spesifikasi kebutuhan implisit yang diinginkan oleh stakeholder.
2. Perancangan PL harus menghasilkan produk kerja yang mudah dibaca dan dimengerti bagi mereka yang membuat kode-kode program dan yang akan melakukan mengujian untuk kemudian mendukung PL.
3. Perancangan PL seharusnya menyediakan gambaran lengkap tentang PL, menangani permasalahan data, fungsional, dan perilaku dari perspektif implementasi.

PEMODELAN SISTEM dengan UML

1. Unified Modelling Language

Menurut Booch, et.al. UML adalah bahasa standar untuk menulis blue print PL. UML dapat digunakan untuk memvisualisasikan, menentukan, membuat, dan mendokumentasikan artefak dari sistem PL secara intensif. UML sesuai untuk sistem pemodelan mulai dari system informasi perusahaan, aplikasi berbasis web yang terdistribusi, bahkan sampai sistem real time embedded yang sulit.

UML adalah proses yang independen, walaupun secara optimal harus digunakan dalam proses yang menggunakan case driven, architecture-centric, iterative, dan incremental.

Diagram-Diagram dalam UML

1. Class Diagram

Menunjukkan seperangkat kelas, antarmuka, dan kolaborasi dan hubungan di antara mereka. Class Diagram membahas desain statis dari suatu sistem.

2. Object Diagram

Menunjukkan satu set objek dan hubungan antara objek. Diagram objek memodelkan instance dari hal-hal yang terdapat dalam Class Diagram. Diagram objek digunakan untuk memodelkan desain statis suatu sistem, untuk memvisualisasikan, menentukan, dan mendokumentasikan model struktural, dan membangun aspek statis sistem melalui teknik maju (forward) dan mundur (reverse).

3. Component Diagram

Menunjukkan organisasi dan ketergantungan antar sekumpulan komponen.

4. Deployment Diagram

Menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. Diagram ini terdiri dari node yang merupakan perangkat keras dan membungkus satu atau lebih komponen.

5. Use Case Diagram

Menunjukkan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Diagram ini sangat penting dalam mengatur dan memodelkan perilaku suatu sistem.

PERANCANGAN BERORIENTASI OBJEK

1. PENDAHULUAN

Sistem berorientasi objek terdiri dari objek yang berinteraksi yang mempertahankan keadaan (state) local dan menyediakan operasi pada state tersebut. Representasi state bersifat pribadi dan tidak dapat diakses langsung dari luar objek. Proses desain berorientasi objek melibatkan perancangan kelas objek dan hubungan antara kelas-kelas tersebut.

Mengubah implementasi suatu objek atau menambahkan metode atau operasi tidak mempengaruhi objek lain dalam sistem.

Desain Arsitektur adalah suatu disiplin ilmu yang berfokus pada pemenuhan dan pemenuhan kebutuhan dan permintaan, menciptakan ruang hidup, menggunakan alat-alat tertentu dan terutama kreativitas.

Oleh karena itu, tujuannya adalah untuk menggabungkan teknologi dan estetika, meskipun ada anggapan umum bahwa arsitektur hanyalah tugas teknologi. Dengan cara yang sama, ia memadukan desain, yang dipahami sebagai proses kreatif, dan arsitektur, yang didasarkan pada penciptaan dan penyajian solusi pada tingkat teknis.

Dengan memadukan kedua disiplin tersebut, desain arsitektur mencari nilai dan kualitas formal karya, melalui pengalaman spasial. Secara umum, kami mengaitkannya dengan gambar, sketsa, atau garis besar suatu proyek, dan ini adalah salah satu dasar fundamentalnya.

Dalam aspek desain arsitektur ini, ada juga faktor lain yang terlibat, antara lain terkait dengan geometri, ruang atau estetika. Lagi pula, arsitektur, dan karenanya desain arsitektur, terdiri dari banyak elemen dan proses atau fase. Saat merancang, seorang arsitek harus memperhitungkan bahwa ia harus melakukan analisis, untuk merancang dan membangun sesuai dengan kebutuhan dan sumber daya, selalu mengingat estetika dan karakteristik teknis, serta aturan dasar konstruksi.

Itu sebabnya proses yang mampu mengidentifikasi semua variabel ini harus mempertimbangkan kebutuhan untuk mencerminkan kebutuhan, baik secara artistik maupun teknis di atas kertas (atau perangkat lunak).

Dalam hal ini, garis adalah elemen utama dari desain arsitektur, yang menentukan beberapa aspek seperti bentuk, dimensi, dan posisi dari berbagai ruang yang mengintegrasikan proyek.

Alasan arsitektur PL:

Representasi arsitektur PL adalah sesuatu yang memungkinkan terjadinya komunikasi di antara semua pihak yang tertarik pada pengembangan sistem berbasis computer. Arsitektur yang dibuat di awal perancangan akan memiliki efek yang menentukan pada semua pekerjaan RPL selanjutnya. Arsitektur menggambarkan model yang relatif kecil dan mudah dipahami, dan menggambarkan bagaimana sistem distrukturkan dan bagaimana komponen di dalamnya saling bekerja sama.

Sasaran dari deskripsi arsitektural:

• Untuk menetapkan kerangka kerja konseptual dan kosa kata yang digunakan selama perancangan arsitektur PL, Untuk menyediakan panduan yang rinci pada waktu, merepresentasikan deskripsi arsitektural, Untuk memandu praktek perancangan yang baik.

Beberapa pertimbangan dalam keputusan

Arsitektur:

1. Adakah arsitektur aplikasi generik yang dapat bertindak sebagai template untuk sistem yang sedang dirancang?

2. Bagaimana sistem akan didistribusikan ke sejumlah perangkat keras?

3. Pola atau gaya arsitektur apa yang digunakan?

4. Pendekatan fundamental apa yang digunakan untuk menyusun sistem?

5. Bagaimana komponen struktural dalam sistem akan terdekomposisi menjadi sub-komponen?

DASAR-DASAR PENGUJIAN PL

Pengujian perangkat lunak adalah proses menjalankan dan mengevaluasi sebuah PL secara manual maupun otomatis untuk menguji apakah PL sudah memenuhi persyaratan atau belum, atau untuk menentukan perbedaan antara hasil yang diharapkan dengan hasil sebenarnya. Pengujian bertujuan untuk mencari kesalahan. Pengujian yang baik adalah pengujian yang memiliki kemungkinan besar dalam menemukan kesalahan sebanyak mungkin dengan usaha sekecil mungkin.

Tujuan Pengujian

a. Menilai apakah perangkat lunak yang dikembangkan telah memenuhi kebutuhan pemakai.
b. Menilai apakah tahap pengembangan perangkat lunak telah sesuai dengan metodologi yang digunakan.
c. Membuat dokumentasi hasil pengujian yang menginformasikan kesesuaian perangkat lunak yang diuji dengan spesifikasi yang telah ditentukan.

Testability

Testability adalah kemampuan PL untuk dapat diuji artinya seberapa mudah sebuah program komputer untuk bias diuji.

Karakteristik Pengujian

a. Pengujian yang baik memiliki probabilitas tinggi untuk menemukan kesalahan

b. Pengujian yang baik tidak berulang-ulang, waktu dan sumber daya pengujian terbatas

c. Pengujian terbaik harus menjadi “bibit terbaik” yaitu pengujian yang memiliki kemungkinan tertinggi dalam mengungkap seluruh kelas kesalahan

d. Pengujian yang baik tidak terlalu sederhana atau tidak terlalu rumit

PENGUJIAN APLIKASI WEB

Pengujian Aplikasi WEB adalah serangkaian aktivitas yang tujuannya untuk menemukan kesalahan dalam isi, fungsi, kegunaan, kemampuan navigasi, kinerja, kapasitas dan keamanan aplikasi web sebelum aplikasi-aplikasi web yang dibuat dikirimkan ke end user. Hal ini penting karena jika end user menemukan kesalahan yang membuat mereka meragukan aplikasi web tersebut, mereka akan pergi ke web lain untuk mencari isi dan informasi, Langkahnya dimulai dengan fokus pada aspek aplikasi web yang terlihat oleh user dan berlanjut pada pengujian yang terkait dengan teknologi dan infrastruktur.

Kesalahan Atribut pada Pengujian Aplikasi Web

a. Pengujian web mengungkap masalah yang didapatkan pertama kali di sisi client, melalui antarmuka yang implementasinya pada sebuah browser atau perangkat komunikasi pribadi

b. Aplikasi web diimplementasikan pada beberapa konfigurasi dan lingkungan yang berbeda, kemungkinan sulit untuk menemukan kesalahan di luar lingkungan tempat kesalahan pertama kali ditemukan

Strategi Pengujian Langkah-langkahnya:

1. Model konten untuk aplikasi web ditinjau untuk menemukan kesalahan
2. Model antarmuka ditinjau untuk memastikan bahwa semua use-case dapat diakomodasi
3. Model perancangan ditinjau untuk mengungkap kesalahan navigasi
4. Antarmuka pengguna diuji untuk menggungkap kesalahan dalam presentasi dan/atau mekanik navigasi
5. Komponen fungsional diuji untuk setiap unit

Perencanaan Pengujian

Sebuah rencana aplikasi web mengidentifikasikan:

a. Himpunan tugas-tugas yang diterapkan ketika pengujian dimulai

b. Produk kerja yang dihasilkan ketika setiap tugas pengujian dijalankan

c. Cara dimana hasil pengujian dievaluasi, dicatat, dan digunakan kembali saat pengujian regresi dilakukan

IMPLEMENTASI

Perancangan dan implementasi PL adalah tahap dalam proses RPL dimana dikembangkan sistem PL yang dapat dieksekusi. Implementasi adalah proses mewujudkan desain sebagai sebuah program. RPL mencakup semua kegiatan yang terlibat dalam pengembangan PL dari persyaratan awal sistem hingga pemeliharaan dan pengelolaan sistem yang digunakan. Implementasi dapat melibatkan pengembangan program atau menyesuaikan dan mengadaptasi sistem generik, off-the-shelf untuk memenuhi persyaratan khusus dari suatu organisasi.

PERMINTAAN PEMELIHARAAN

Pengguna mengirimkan sebagian besar permintaan untuk pemeliharaan korektif dan adaptif ketika sistem tidak berfungsi dengan baik, atau jika mereka menginginkan fitur baru.

1. Determinasi Awal

   Ketika pengguna mengajukan permintaan pemeliharaan, administrator membuat penentuan awal, jika permintaan memerlukan perhatian segera, administrator akan mengambil tindakan sekaligus.

2. Komite Peninjau Sistem

   Ketika suatu permintaan melebihi tingkat biaya yang telah ditentukan atau melibatkan perubahan konfigurasi utama, komite peninjau sistem akan menetapkan prioritas, atau menolaknya.

3. Penyelesaian Tugas

   Administrator sistem bertanggung jawab untuk mempertimbangkan pengalihan tugas di antara staf IT atau membatasi tugas pemeliharaan kepada individu atau tim tertentu agar tugas dapat diselesaikan dengan baik

4. User Notification

   Pengguna yang memulai permintaan pemeliharaan mengharapkan tanggapan yang cepat, terutama jika situasi tersebut secara langsung mempengaruhi pekerjaan mereka. Bahkan ketika tindakan korektif tidak dapat terjadi dengan segera, pengguna akan menghargai umpan balik dari administrator system dan harus terus diberitahu tentang keputusan atau tindakan yang akan mempengaruhi pengguna.