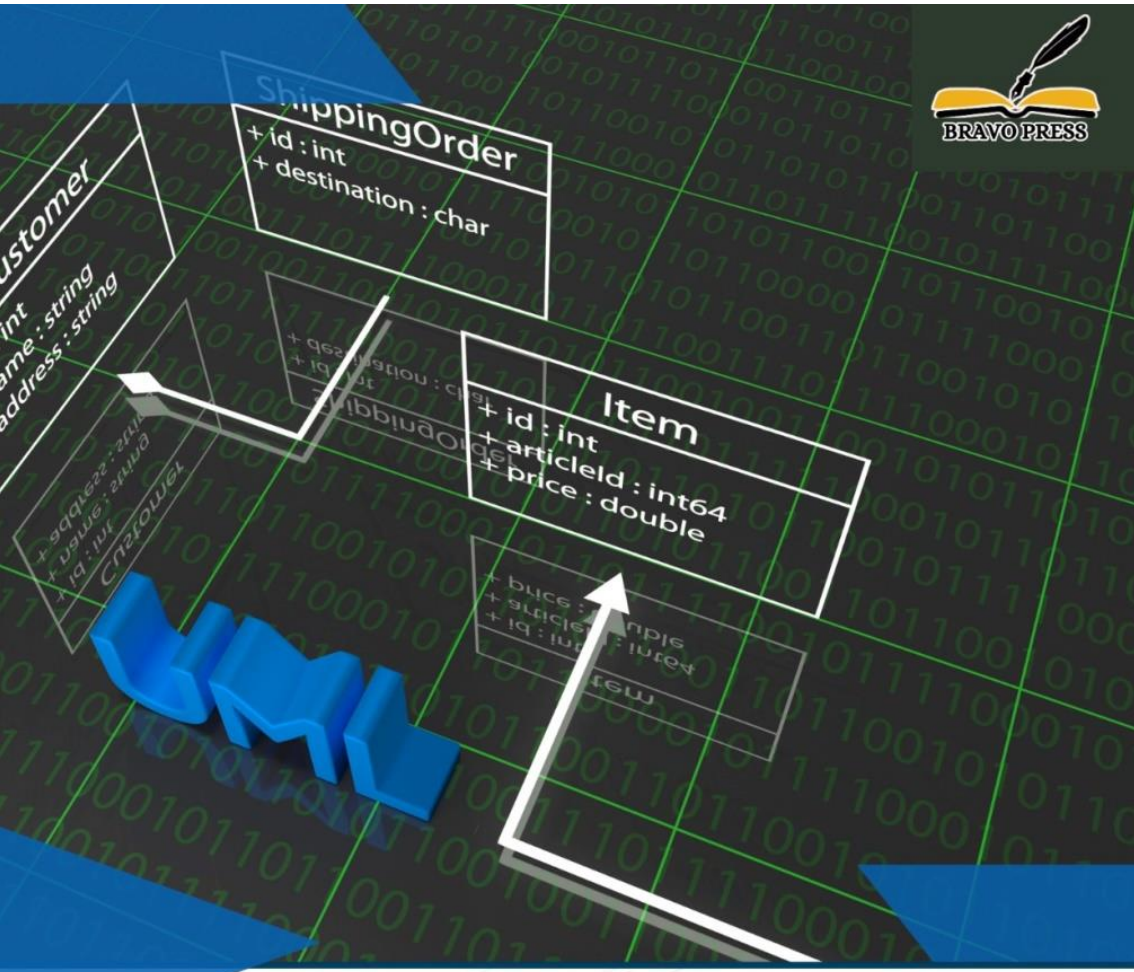
TEORI UML DAN IMPLEMENTASI PRAKTEK: Panduan Untuk Pengembangan Perangkat Lunak

Book · September 2024



TEORI UML DAN IMPLEMENTASI PRAKTEK

Panduan Untuk Pengembangan Perangkat Lunak

Chairun Nisa' | Andi Wijaya, M.Kom | Fathur Rizal, M.Kom

Editor: Weni Yuliani, S.Si., M.M

TEORI UML DAN IMPLEMENTASI PRAKTEK
Panduan Untuk Pengembangan Perangkat Lunak

Penulis:
Chairun Nisa'
Andi Wijaya, M.Kom
Fathur Rizal, M.Kom

BRAVO PRESS
CV BRAVO PRESS INDONESIA

TEORI UML DAN IMPLEMENTASI PRAKTEK
Panduan Untuk Pengembangan Perangkat Lunak

Penulis :
Chairun Nisa' l Andi Wijaya, M.Kom l Fathur Rizal, M.Kom

ISBN : 978-623-10-3680-3
Editor : Weni Yuliani, S.Si., M.M
Penyunting : Taufik Hidayat, S. Kom., M. Kom
Desain Sampul dan Tata Letak : Aviva Anisyah, S.Pd

Penerbit : CV BRAVO PRESS INDONESIA
Anggota IKAPI No. 022/RAU/2024

Redaksi :
Perumahan Indah Harisanda blok f6 Jalan saudara RT 03/RW 06 Kel/Desa Tuah Madani, Kec. Tuah Madani, Kota Pekanbaru, Riau
Website : www.bravopress.id
Email : bravopressindonesia@gmail.com

Cetakan pertama, September 2024

KATA PENGANTAR

Dengan mengucapkan puji Syukur kehadirat Allah SWT, atas limpahan Rahmat dan hidayah-Nya, maka Penulisan Buku Referensi dengan judul Teori Uml Dan Implementasi Praktek : Panduan Untuk Pengembangan Perangkat Lunak dapat diselesaikan. Buku ini berisikan bahasan tentang pemahaman konsep dasar UML, jenis-jenis diagram UML, dan bagaimana menerapkannya dalam proyek perangkat lunak nyata. Maka susunan buku referensi ini terdiri dari 10 bab bagian utama, diantaranya sebagai berikut:

- BAB 1 : Pengantar UML
- BAB 2 : Jenis-jenis Diagram UML
- BAB 3 : Pemahaman Mendalam tentang Diagram Kelas
- BAB 4 : Penggunaan Efektif Diagram Kasus Penggunaan
- BAB 5 : Pemodelan Proses dengan Diagram Aktivitas
- BAB 6 : Pemahaman Diagram Urutan
- BAB 7 : Integrasi Diagram dalam Proyek Perangkat Lunak
- BAB 8 : Praktik Terbaik dalam Pemodelan dengan UML
- BAB 9 : Studi Kasus : Proyek Pengembangan Perangkat Lunak dengan UML
- BAB 10 : Masa Depan UML

Buku ini masih banyak kekurangan dalam penyusunannya. Oleh karena itu, kami sangat mengharapkan kritik dan saran demi perbaikan dan kesempurnaan buku ini selanjutnya. Kami mengucapkan terima kasih kepada berbagai pihak yang telah membantu dalam proses penyelesaian Buku ini.

Semoga Buku ini dapat menjadi sumber referensi dan literatur yang mudah dipahami.

Probolinggo, September 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENGANTAR UML

A. Definisi dan Sejarah UML

Semakin berkembangnya teknologi perangkat lunak, maka diperlukannya sebuah Bahasa untuk memodelkan perangkat lunak serta adanya standarisasi sehingga mudah dipahami oleh semua pihak. Bahasa pemodelan pembangunan perangkat lunak yang sempat banyak digunakan adalah Data *Flow* Diagram (DFD) dengan menggunakan pemrograman struktur atau *Prosedural.* Selain itu ada juga *State Transition* Diagram (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata). Kemudian muncul sebuah standarisasi Bahasa pemrograman pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML) (Rosa & Shalahuddin, 2013).

UML merupakan sebuah bahasa pemodelan secara visual yang dipakai untuk perancangan sistem berorientasi objek. UML ini sebagai bentuk perkembangan dari desain berorientasi objek dan metode analisis berorientasi objek atau sering kita sebut OOA&D yang muncul sekitar akhir tahun 80-an dan awal 90-an dengan cakupan lebih luas (Corporation, 1997). UML telah menjadi standar dalam memvisualisasikan, merancang, membangun serta

mendokumentasikan penulisan dalam sebuah perangkat lunak yang akan membantu pengembang untuk menjelaskan dan mempresentasikan kepada pengguna atau pengembang bisnis terkait fungsionalitas dari sistem yang akan dibangun.

Menurut Grady Booch, UML adalah Bahasa pemodelan visual yang digunakan untuk menentukan, memvisualisasikan, merancang dan mendokumentasikan sistem kompleks. UML menyediakan seperangkat diagram standar untuk menggambarkan berbagai aspek sistem perangkat lunak, termasuk struktur statis, perilaku dinamis, interaksi antar komponen dan arsitektur sistem secara keseluruhan. Booch juga mendefinisikan UML sebagai Bahasa yang visual, berbasis objek dan eksistensial yang mana UML menggunakan diagram untuk kemudian menggambarkan sistem perangkat lunak agar mudah dipahami, didasarkan pada paradigma dominan dengan pemrograman berorientasi objek dan dengan Bahasa yang dapat diperluas untuk memenuhi kebutuhan spesifik proyek (Awaad et al., 1978).

UML secara resmi dimulai pada Oktober 1994, ketika Grady Booch dan Jim Rumbaugh berkolaborasi mengembangkan standar notasi untuk metode perangkat lunak berorientasi objek di *Relational Software Coorperation*, sebab pada saat itu tidak ada standar penggunaan model yang berbasis orientasi objek. Kemudian mereka berdua mengambil masing-masing pendekatan metode OO untuk membuat suatu model Bahasa yang seragam yang disebut UML. UML versi 0.8 berhasil dirilis pada Oktober 1995, dan tidak lama kemudian Ivar Jacobson bergabung dalam tim *object oriented development* mereka dan merilis versi 0.9 pada bulan Juni 1996. Pada tahun tersebut pengembangan

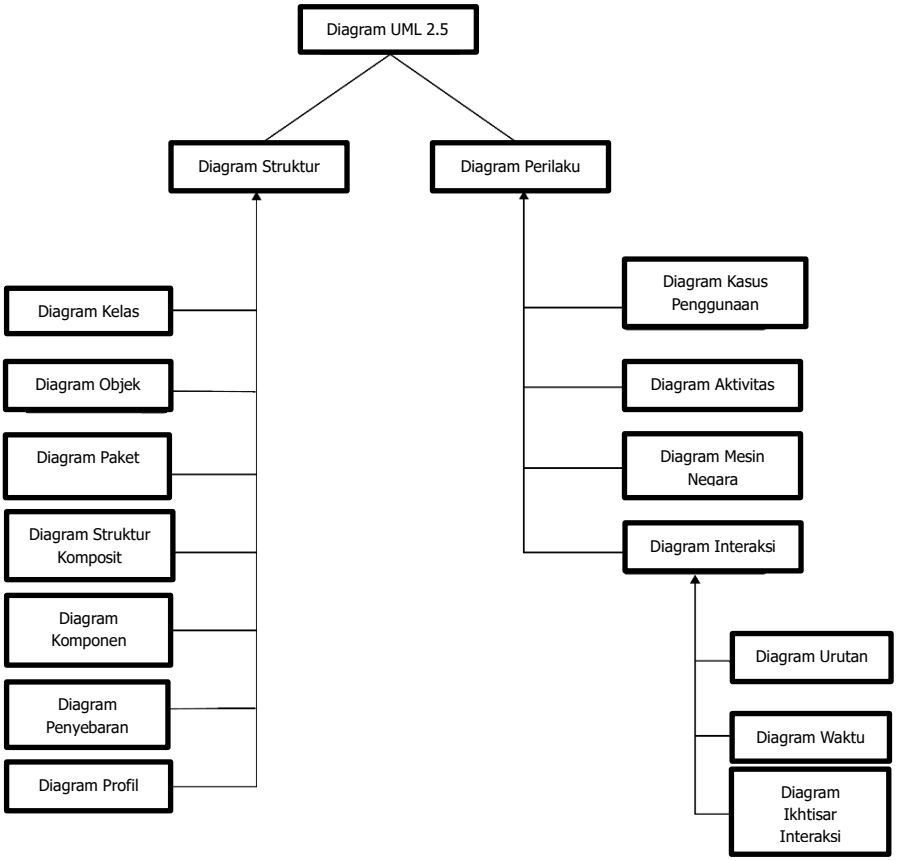
BAB 2
JENIS-JENIS DIAGRAM UML

Diagram UML merupakan gambaran grafis *parsial* dari model sistem yang sedang dirancang, diimplementasikan, maupun yang sudah ada. Elemen grafis atau simbol dalam diagram UML berupa node UML yang dihubungkan oleh tepi yang sering disebut rute atau aliran. Elemen-elemen ini mewakili komponen dalam model UML sistem. *Use Case* yang disediakan sebagai teks *template* merupakan salah satu jenis dokumentasi yang dapat dimasukkan ke dalam model UML sistem.

Simbol grafis utama yang ditampilkan pada diagram menggambarkan jenisnya. Misalnya diagram kelas, dimana kelasnya merupakan simbol utama dalam bagian model. Sementara pada diagram *Use Case* menampilkan aktor dan *Use Case* dan pada diagram urutan menggambarkan urutan pertukaran pesan antara lifeline nya.

Terdapat beberapa jenis diagram dalam UML yang masing-masing memiliki tujuan dan peranannya sendiri dalam proses pengembangan sistem. UML versi 2.5 terdiri dari 14 macam diagram yang kemudian dikelompokkan dalam dua kategori utama yaitu diagram struktur (structural diagrams) dan diagram perilaku (behavior diagrams) dalam spesifikasi UML tahun 2015.

Diagram Struktur menampilkan struktur statis yang komponennya terdiri dari berbagai tingkat abstraksi, implementasi, dan hubungan mereka. Sedangkan diagram perilaku menggambarkan perilaku dinamis dalam sistem yang dapat dipikirkan sebagai urutan modifikasi yang dilakukan pada sistem dari waktu ke waktu. Diagram yang dibuat dengan UML 2.5 ini dapat diatur secara hierarkis, seperti gambar berikut.



Gambar 2. 1 Struktur Diagram UML Versi 2.5

BAB 3
PEMAHAMAN MENDALAM TENTANG DIAGRAM KELAS

Diagram kelas merupakan salah satu diagram paling penting dalam UML karena menggambarkan struktur statis sistem dengan menunjukkan kelas, atribut, operasi atau metode serta hubungan antar kelas. Dengan memahami diagram kelas, kita dapat merancang dan mendokumentasikan sistem perangkat lunak secara lebih efektif. Berikut pembahasan mengenai struktur dan notasi dalam diagram kelas.

A. Struktur dan Notasi Diagram Kelas

Struktur diagram kelas sendiri terdiri dari beberapa komponen utama, yaitu kelas, atribut, metode, dan berbagai jenis hubungan antar kelas diantaranya association, agregation, composition, agregasi dan dependency. Selain itu, diagram ini juga dapat melibatkan notasi lain seperti *interface* dan *stereotype* untuk memberikan informasi tambahan pada elemen-elemen diagram.

Kelas merupakan sebuah spesifikasi yang apabila diinstansikan akan menghasilkan sebuah objek yang merupakan inti dari pengembangan dan desain berorientasi objek untuk menggambarkan suatu keadaan (atribut/properti) pada sistem serta memanipulasinya (Dharwiyanti, 2003). Setiap kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan

kebutuhan sistem agar dalam pembuatan perangkat lunak dapat sesuai dengan perancangan diagram kelas. Oleh karena itu, pada diagram kelas semestinya memiliki jenis-jenis untuk susunan struktur kelas yang baik, antara lain:

1) Main Class

Merupakan titik awal eksekusi program saat sistem dijalankan. Dalam sebuah aplikasi, biasanya hanya ada satu kelas main yang berisi metode "main()" sebagai inisialisasi objek-objek, menampilkan menu utama, memproses input pengguna serta menjalankan alur kerja utama aplikasi.

2) View Class

Merupakan kelas yang menangani tampilan sistem dalam pendefinisian maupun pengaturan tampilan ke pengguna. Selain itu, kelas ini juga memiliki kemampuan untuk mengatur tata letak elemen-elemen antarmuka dan memperbaikinya berdasarkan data model.

3) Controller Class

Kelas ini menangani fungsi-fungsi yang harus ada pada sistem sekaligus sebagai kelas proses yang menangani proses bisnis pada perangkat lunak. Controller class bertindak sebagai perantara antara view dan model, menerima input pengguna dari view untuk kemudian diproses dan memperbarui model sesuai kebutuhan. Kemudian mengambil data dari model dan meneruskannya ke *view* untuk ditampilkan.

BAB 4
PENGGUNAAN EFEKTIF DIAGRAM KASUS PENGGUNAAN

Dalam konteks UML, dilakukan konseptualisasi dengan pembuatan *Use Case* diagram yang menjelaskan bagaimana perangkat lunak akan digunakan . Diagram *Use Case* digunakan untuk menunjukkan interaksi yang terjadi antara sistem dan lingkungannya. Penggunaan diagram *Use Case* sebenarnya ada dua yaitu pada diagram perilaku untuk menggambarkan perilaku sistem dan diagram struktur khususnya diagram kelas yang menunjukkan Batasan pengklasifikasian sebagai aktor atau penggunaan *Use Case* yang terkait dengan asosiasi. Selain itu, *Use Case* menunjukkan perilaku dinamis suatu sistem dengan menggambarkan bagaimana sistem berfungsi dengan menggabungkan *Use Case*, aktor, dan hubungan ketiganya. *Use Case* juga dapat memberitahu pengguna bagaimana menangani suatu sistem.

A. Identifikasi dan Pemodelan Aktor

Aktor merupakan entitas yang berinteraksi dengan sistem yang memulai *Use Case* dari luar lingkup *Use Case*. Aktor mewakili siapapun atau apa saja yang berinteraksi dengan sistem serta memberikan dan menerima informasi ke dan dari sistem. Aktor dapat berupa individu, kelompok orang, atau sistem eksternal lainnya. Identifikasi dan pemodelan

aktor merupakan bagian penting dari proses pengembangan sistem. Berikut beberapa tujuan identifikasi dan pemodelan aktor antara lain:

- Memahami siapa yang akan menggunakan sistem dan kebutuhan dari sistem.
- Menentukan fungsionalitas yang harus disediakan oleh sistem untuk kebutuhan pengguna.
- Membuat dokumentasi sistem untuk menjelaskan alur sistem saat digunakan.

Identifikasi aktor merupakan Langkah awal yang krusial dalam pengembangan sistem. Berikut Langkah-langkah untuk melakukan identifikasi aktor pada diagram *Use Case*:

1. Melakukan identifikasi pada pengguna utama sistem.
2. Melakukan identifikasi pada pengguna sekunder atau pengguna yang hanya untuk tugas-tugas tertentu.
3. Melakukan identifikasi sistem eksternal yang berinteraksi dengan sistem.
4. Menganalisis peran dan tanggung jawab aktor dalam sistem.

Adapun pemodelan aktor merupakan proses memvisualisasikan interaksi antara sistem dan entitas eksternal. Pemodelan aktor juga memiliki beberapa Langkah-langkah yang dapat digunakan:

1. Temukan semua entitas eksternal yang berinteraksi dengan sistem.
2. Gunakanlah simbol orang untuk mewakili aktor pada diagram *Use Case*.
3. Menghubungkan aktor dengan *Use Case* dengan garis asosiasi untuk menunjukkan interaksi keduanya.

BAB 5
PEMODELAN PROSES DENGAN DIAGRAM AKTIVITAS

Interaksi elemen dinamis atau aliran keseluruhan sistem, terlibat dalam perilaku sistem melalui diagram aktivitas. Proses sistem dan prosedur dibangun menggunakan aliran logika yang disesuaikan dengan berbagai kondisi, kebutuhan kerja tim, akses data, gangguan, dan variasi aliran logika lainnya. Diagram aktivitas menunjukkan urutan proses sistem atau alur kerja yang berupa aktivitas dan tindakan pilihan atau pengulangan.

A. Notasi dan Konsep Diagram Aktivitas

Diagram aktivitas merupakan pengembangan terbaru dari *Use Case* yang memiliki alur aktivitas yang membuatnya terkait dengan menu tertentu dan memungkinkan melakukan proses bisnis dalam sebuah sistem. Diagram aktivitas memberikan gambaran tentang pemodelan sekuensial dan algoritma melalui proses paralel yang memiliki sifat kompleks.

Diagram aktivitas dapat digunakan untuk mengorganisasikan alur tampilan sistem tertentu atau mendefinisikan mereka. Dalam diagram aktivitas, tanda panah menunjukkan urutan aktivitas yang dilakukan dari awal hingga akhir. Hal ini membuat diagram aktivitas mirip dengan flowchart atau diagram alur. Namun diagram

aktivitas bekerja secara paralel, bercabang, bersamaan, atau tunggal yang membedakannya dengan flowchart.

Diagram aktivitas menggunakan simbol atau notasi tertentu untuk menggambarkan aktivitas yang terjadi dalam sebuah sistem secara jelas dan terstruktur. Berikut beberapa notasi yang ada dalam diagram aktivitas.

a) *Initial state*/Status Awal

Initial state atau status awal menggambarkan tahap awal atau simbol untuk memulai serangkaian Tindakan pada diagram aktivitas. Suatu proses hanya memiliki satu kondisi awal, kecuali untuk menggambarkan aktivitas bersarang. Notasi ini digambarkan dengan lingkaran hitam kecil.

b) Status Akhir

Final state atau status akhir menandakan keadaan akhir dari suatu aktivitas pada diagram aktivitas sebagai penyelesaian semua aktivitas atau aliran proses. Status akhir dalam UML digambarkan dengan lingkaran hitam di sekelilingnya atau biasa disebut *bull's eye* (mata sapi). Pada status awal hanya memiliki satu tindakan untuk memulai aktivitas, namun status akhir mungkin memiliki sejumlah tindakan akhir dari sebuah aktivitas.

c) Aktivitas

Aktivitas menggambarkan aktivitas yang terjadi atau dilakukan dalam sebuah aliran kerja dan menyimpan seluruh aktivitas pada sebuah sistem. Aktivitas diawali dengan kata kerja berupa tindakan konkret, seperti "pengiriman pesan" atau "pengecekan data", atau dapat berupa aktivitas yang abstrak, seperti "evaluasi kinerja" atau "proses pembayaran".

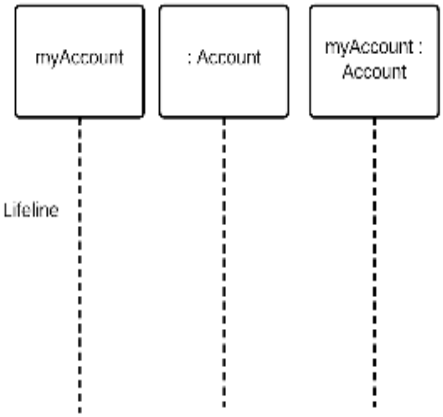
BAB 6
PEMAHAMAN DIAGRAM URUTAN

A. Komunikasi Antar Objek

Diagram sekuen merupakan alat visual yang digunakan untuk menunjukkan bagaimana objek atau komponen dalam suatu sistem berkomunikasi dengan menggunakan pesan untuk mencapai tujuan dan waktu tertentu. Memahami komunikasi antar objek dengan baik adalah komponen penting dalam diagram ini untuk mengetahui prosedur kerja sistem dan mengidentifikasi masalah apapun yang mungkin terjadi. Komunikasi antar objek tersebut digambarkan dengan beberapa elemen dasar, yaitu:

a) *Lifeline*

Lifeline merupakan bagian dari diagram sekuen yang menunjukkan waktu hidup objek dan membantu melihat bagaimana objek berinteraksi dan berkoordinasi untuk memproses pesan selama waktu hidup objek. Lifeline menunjukkan entitas peserta dalam skenario kolaborasi yang ditulis di bagian atas diagram secara horizontal serta garis vertikal putus-putus yang berada di bawah setiap objek yang menunjukkan keberadaan objek.



Gambar 6. 1 Notasi *Lifeline* Diagram Urutan

Nama objek diatas menyatakan bahwa "*myAccount*" sebagai nama objek yang khusus sedangkan "*myAccount:Account"* untuk nama objek yang umum. "*Account*" biasanya digunakan untuk menunjukkan objek apapun dalam kelas. Setiap objek memiliki *lifeline* yang diwakili oleh garis putus-putus di bawahnya. Panah yang menunjuk dari objek yang dikirim ke objek yang diterima menunjukkan pesan antar objek. Setiap objek dikendalikan oleh objek itu sendiri, memungkinkan control penuh atas tindakan seperti mentransfer data, mengembalikan data, menanggapi permintaan, dan melindungi sistem. Kelompok ini bekerja sama dengan berkomunikasi atau berinteraksi satu sama lain.

b) Aktor

Aktor merupakan komponen dalam diagram sekuen yang mewakili entitas eksternal yang berinteraksi dengan

BAB 7
INTEGRASI DIAGRAM DALAM PROYEK PERANGKAT LUNAK

Pada pengembangan perangkat lunak ditandai dengan abstraksi dan kompleksitas. Karena berbagai kode, algoritma, dan arsitektur sistem yang berantakan, ada kemungkinan gambaran besar akan hilang. Dengan sifatnya yang visual, diagram memiliki peran untuk mendorong kolaborasi antara pihak yang terlibat, memfasilitasi komunikasi yang efektif, dan menyederhanakan konsep yang rumit menjadi bahasan yang mudah dipahami.

Diagram terintegrasi dengan koreksi yang ketat digunakan selama proses pengembangan perangkat lunak untuk memastikan vitalitas dan efisiensi yang optimal. diagram ini tidak hanya membantu pengguna melihat struktur, desain, dan operasi sistem, tetapi juga memulai proses pengembangan perangkat lunak yang lebih efisien, efektif, dan berkualitas tinggi.

A. Pengembangan Langkah Demi Langkah

Untuk memastikan bahwa setiap tahap pengembangan perangkat lunak didokumentasikan dengan baik dan dipahami oleh seluruh tim, maka perlunya pendekatan secara sistematis untuk mengintegrasikan diagram UML dalam proyek perangkat lunak. Berikut Langkah demi

Langkah yang harus dilakukan dalam proyek perangkat lunak:

a. Identifikasi Kebutuhan dan Spesifikasi

Langkah pertama dalam pengembangan perangkat lunak adalah dengan memahami kebutuhan dan spesifikasi proyek. Langkah tersebut sangat penting untuk keberhasilan proyek secara keseluruhan, karena jika terdapat kesalahan pada Langkah ini dapat menyebabkan masalah yang serius. Proses identifikasi kebutuhan dan spesifikasi dijelaskan sebagai berikut:

1) Pengumpulan Kebutuhan

Pengumpulan kebutuhan merupakan proses untuk mengidentifikasi dan mencatat kebutuhan maupun keinginan pemangku kepentingan dari sistem yang akan dibangun. Pada tahap ini, beberapa tugas penting dilakukan seperti mengidentifikasi pemangku kepentingan utama, teknik pengumpulan kebutuhan, dan kategorisasi kebutuhan tersebut.

Identifikasi pemangku kepentingan tersebut dilakukan pada orang-orang yang akan menggunakan sistem maupun orang yang bertanggung jawab untuk pemeliharaan sistem seperti pengguna akhir, manajer proyek, pengembang, analis bisnis, serta sponsor proyek. Selanjutnya pengumpulan kebutuhan dengan melakukan wawancara, survei dan kuesioner, workshop dan sesi kolaborasi, observasi, dan dokumentasi yang ada sebagai tambahan wawasan. Kemudian suatu fungsional, non fungsional, dan kebutuhan domain merupakan bagian kategorisasi kebutuhan yang juga harus dilakukan oleh sistem.

BAB 8
PRAKTIK TERBAIK DALAM PEMODELAN DENGAN UML

A. Strategi dan Tips Pemodelan

UML merupakan alat yang sangat berguna untuk memvisualisasikan dan mendokumentasikan desain sistem. Untuk membuat diagram UML tersebut, penting untuk mengikuti praktik terbaiknya dalam laman (Geeks for Geeks, 2024):

1. Memahami audien seperti menyesuaikan detail dan pilihan agar sesuai dengan pemahaman audien, seperti apakah mereka pengembang, arsitek, atau pemangku kepentingan.
2. Menjaga diagram tetap sederhana dan fokus untuk menghindari kekacauan dan detail yang tidak perlu.
3. Gunakan konvensi penamaan yang konsisten pada penggunaan nama kelas, objek, atribut, metode, dan elemen UML sehingga dapat membantu orang lain memahami diagram dengan lebih baik.
4. Ikuti notasi UML standar untuk memastikan diagram yang dibuat mudah di pahami oleh orang lain.
5. Menjaga hubungan Eksplisit dengan menentukan dan menandai hubungan antar elemen untuk menunjukkan sifat koneksi antara kelas, objek, atau *Use Case*.

Gunakan panah yang sesuai, notasi multiplisitas, dan nama asosiasi.

B. Kolaborasi Tim dalam Pemodelan

Kolaborasi antar tim dalam proses pemodelan merupakan aspek yang sangat penting dalam menciptakan model yang akurat, komprehensif, dan berguna. Kolaborasi merupakan cara untuk mengatasi masalah yang tidak terdefinisikan dengan baik. Pengetahuan terbatas seseorang dapat mengakibatkan model yang tidak lengkap, bias atau bahkan salah. Dengan menyatukan ide dan keahlian dari berbagai individu, tim pemodelan dapat melakukan beberapa hal, antara lain:

a. Melibatkan semua pemangku kepentingan : interaksi antara anggota tim memungkinkan komunikasi antara semua pihak yang terlibat dalam proyek, termasuk pengembangan, desainer, dan pemangku kepentingan. Hal ini memastikan bahawa setiap perspektif dan kebutuhan disesuaikan dalam model, meningkatkan akurasi dan relevansi.
b. Meningkatkan efisiensi : kerja tim mengurangi duplikasi bisnis dan mempercepat proses pemodelan. Setiap anggota tim dapat berkonsentrasi pada bidang keahlian mereka. Meningkatkan produktivitas dan efisiensi keseluruhan.
c. Meningkatkan kualitas model : melalui kolaborasi, kita dapat secara efektif menghasilkan ide, menyempurnakan model, dan memecahkan konflik dengan cara yang konstruktif. Proses ini menghasilkan gambaran yang lebih koheren, konsisten dan realistis dari situasi.

BAB 9
STUDI KASUS : PROYEK PENGEMBANGAN PERANGKAT LUNAK DENGAN UML

Dalam bab ini, akan dibahas sebuah studi kasus proyek pengembangan perangkat lunak menggunakan UML. Proyek yang akan bangun adalah sistem informasi manajemen perpustakaan. Proyek ini bertujuan untuk mengembangkan sistem yang dapat membantu perpustakaan dalam mengelola data buku, anggota, peminjaman, dan pengembalian buku secara efektif dan efisien. Selain perpustakaan, sistem ini akan memudahkan anggota atau pengguna sistem dalam mengetahui informasi perpustakaan seperti ketersediaan buku yang ingin dipinjam.

A. Deskripsi Proyek

Perpustakaan memiliki peran penting dalam menyediakan akses informasi dan sumber daya Pendidikan bagi Masyarakat. Namun, pengelolaan perpustakaan secara konvensional sering menghadapi berbagai kendala seperti kesulitan dalam pengelolaan data buku, anggota, peminjaman, dan pengembalian buku. Oleh karena itu, diperlukan sebuah sistem informasi manajemen perpustakaan yang dapat mengotomatisasi dan mempermudah proses-proses tersebut. Adapun tujuan dari

proyek ini adalah merancang dan mengembangkan sistem informasi manajemen perpustakaan yang dapat membantu dalam:

1. Mengelola data buku secara efektif.
2. Mengelola data anggota perpustakaan.
3. Mengotomatisasi proses peminjaman dan pengembalian buku.
4. Memberikan kemudahan bagi anggota perpustakaan dalam mencari dan meminjam buku.
5. Menghasilkan laporan-laporan yang berguna bagi pengelola perpustakaan.

Sementara untuk ruang lingkup dari proyek sistem informasi manajemen perpustakaan ini mencakup:

1. Pengembangan modul untuk pengelolaan data buku dan data anggota.
2. Implementasi fitur peminjaman dan pengembalian buku.
3. Pembangunan antarmuka pengguna untuk anggota, pustakawan, dan admin.
4. Integrasi dengan sistem basis data untuk menyimpan dan mengelola informasi.

Sistem informasi manajemen perpustakaan ini akan dibangun dengan menggunakan diagram UML dengan beberapa deskripsi berikut:

a. Definisi Aktor

Berikut deskripsi dari pendefinisian aktor pada Sistem Perpustakaan Online:

BAB 10
MASA DEPAN UML

A. Tren dan Perkembangan UML

Sampai saat ini, UML telah menjadi standar yang diakui dalam pemodelan perangkat lunak sejak diperkenalkannya pada pertengahan 1990-an. UML umum digunakan dalam dunia kerja/bisnis untuk memvisualisasikan sistem dan aplikasi berbasis objek yang sudah ada sebelum memasuki tahap coding. Visualisasi tersebut berupa sebuah *Sistem analyst*, *software architect*, dan *technical writer* merupakan beberapa pekerjaan yang sering dihubungkan dengan UML untuk membantu mentransfer ilmu tentang sistem atau aplikasi yang sedang dikembangkan antara pengembang dan bisnis.

Selain di dunia kerja/bisnis, UML juga banyak digunakan di dunia Pendidikan khususnya sebagai alat dalam sejumlah mata kuliah di perguruan tinggi termasuk jurusan Ilmu Komputer, Teknik Informatika, Sistem Informasi dan beberapa jurusan lainnya. Mahasiswa akan mempelajari bagaimana menggunakan UML untuk merancang sistem atau aplikasi, bahkan digunakan sebagai alat bantu dalam pembuatan skripsi, tesis, dan disertasi (Fajar, 2016).

UML terus mengalami perkembangan dan penyempurnaan untuk dapat memenuhi kebutuhan industri yang selalu berubah melalui serangkaian diagram yang

menggambarkan berbagai aspek dari suatu sistem seperti struktur, perilaku, dan interaksi. Pada tahun 2015, Object Management Group (OMG) merilis versi terbaru UML yaitu UML 2.5. Versi ini melakukan perbaikan dengan meningkatkan fungsionalitas dan efisiensi pemodelan sistem. Di antara peningkatan ini sebagai dukungan yang lebih baik untuk pengembangan berbasis model dan pendekatan pengembangan berkelanjutan atau agile.

B. Tantangan dan Peluang

Pada bab-bab sebelumnya sudah dibahas mengenai fungsi maupun manfaat dari penggunaan diagram UML sebagai alat pemodelan dan desain sistem perangkat lunak yang terstandarisasi secara konsisten. Penggunaan diagram UML dalam proses pengembangan perangkat lunak dapat menimbulkan beberapa tantangan, diantaranya (pengertian.or.id, n.d.):

a. Kompleksitas

Kompleksitas dalam UML merujuk pada tingkat kesulitan yang terkait dengan penggunaan, pemahaman, dan penerapan berbagai elemen dan diagram yang tersedia. Banyak faktor dapat menyebabkan kompleksitas ini, seperti banyaknya jenis diagram, notasi yang berbeda, dan konsep-konsep yang harus dikuasai. Bagi pengguna baru UML, kompleksitas dapat menjadi tantangan yang signifikan seperti pemahaman konsep dasar UML, bagaimana memilih diagram yang tepat, penggunaan notasi dan mengintegrasikan UML dengan metodologi pengembangan yang digunakan.

DAFTAR PUSTAKA

A.S Rosa, S. m. (2014). *Activity Diagram*. *16071047*, 8–26.

Ahmad, A. (2022). Pengertian Object Diagram : Fungsi, Komponen, dan Contohnya. In *26 April*. ansoriweb.com. (2024). *Pengertian Sequence Diagram : Tujuan, Simbol, dan Contohnya*. 13 Februari. https://www.ansoriweb.com/2020/04/pengertian-sequence-diagram.html

Awaad, M. H. H., Krauss, H., & Schmatz, H. D. (1978). Electrophoretic characterization of bovine mycoplasma and acholeplasma reference strains. In *Zentralblatt fur Bakteriologie Mikrobiologie und Hygiene - Abt. 1 Orig. A* (Vol. 240, Issue 3).

Basic, V., Diagram, U. C., Diagram, A., Diagram, S., Diagram, C., Diagram, C., Diagram, D., Diagram, I. O., Diagram, O., Diagram, P., & Diagram, T. (2020). Fungsi Dan Pengertian UML. *Definitions*, 1–10.

Bhatt, B., & Nandu, M. (2021). An Overview of Structural UML Diagrams. *International Research Journal of Engineering and Technology*, *8*(8).

Blog. (2023). *Class* Diagram: Pengertian, Simbol, Manfaat beserta Contohnya. *27 June*. https://itbox.id/blog/*class*-diagram-pengertian-simbol-manfaat-beserta-contohnya/#:~:text=Untuk desain model diagram kelas dibagi menjadi dua.,mempunyai *class* kontrol%2C *class interface*%2C dan *class* entity.

Booch, G., & Rumbaugh, J. (n.d.-a). *(bagian dari informasi yang digunakan atau dihasilkan oleh proses pembuatan perangkat lunak,*. 1–28.

Booch, G., & Rumbaugh, J. (n.d.-b).

Modul_I_Dasar_Unified_Modeling_Language. 1–28.

Booch, G., Rumbaugh, J., & Jacobson, I. (2015). Unified Modeling Language User Guide , The (2nd Edition) (Addison-Wesley Object Technology Series) Unified Modeling Language User Guide , The Unified Modeling Language User Guide , The Many of the designations used by manufacturers and sellers to dist. In *ResearchGate* (Vol. 2nd, Issue August).

Dharwiyanti, S. (2003). Pengantar Unified Modeling Language (UML) 1–13.

Didik Widianto, E. (2012). *Pemodelan Sistem dengan UML Review Kuliah*.

Dzikry, M. (2023). *Deployment Diagram: Pengertian, Tujuan dan Fungsinya*. 10 Mei. https://masdzikry.com/pengertian-deployment-diagram/

Efendi. (n.d.). *Apa itu Package Diagram? Mengenal Pengertian Package Diagram*. https://www.nesabamedia.com/pengertian-package-diagram/

Fowler, M. (n.d.). *UML Distilled: A brief Guide to the Standard Object Modeling Language*.

GeeksforGeeks. (2024). *Diagram Bahasa Pemodelan Terpadu (UML)*. 25 Juni. https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/

Hatta, R. (2024). *Penjelasan Unified Modeling Language (UML)*. 24 Mei. https://medium.com/@rezza.hatta2018/penjelasan-unified-modeling-language-uml-6adc9782ed2d

Henderi, O., Kom, M., Perancangan, D., & Informasi, S. (2010). *Object Oriented Modelling With Unified Modeling Language (Uml)*.

Hendini, A. (2016). *Pemodelan UML Sistem Informasi*

GLOSARIUM

Analisis : Proses Pemeriksaan mendalam untuk memahami struktur, fungsi, dan hubungan dari suatu sistem atau data.

Array : Struktur data yang menyimpan elemen-elemen dalam urutan tertentu, yang dapat diakses dengan indeks numerik.

Dekomposer : Proses memecah sistem atau masalah menjadi bagian-bagian yang lebih kecil dan lebih mudah dikelola.

Ekspresif : Kemampuan suatu alat atau bahasa untuk mengkomunikasikan ide atau konsep dengan jelas dan mendalam.

Enkripsi Data : Proses mengubah data menjadi bentuk kode untuk melindunginya dari akses yang tidak sah.

Entitas : Objek atau hal yang dapat diidentifikasi dan dibedakan dalam konteks tertentu, setting kali dalam basis data atau pemodelan.

Fleksibel : Kemampuan untuk menyesuaikan diri atau berubah sesuai dengan kondisi atau kebutuhan yang berbeda.

Fundamental : Bersifat dasar atau mendasar, penting untuk dipahami sebelum mempelajari konsep-konsep yang lebih lanjut.

BIODATA PENULIS

Chairun Nisa'
Mahasiswa Prodi Teknik Informatika Universitas Nurul Jadid

Penulis lahir di Probolinggo pada tanggal 14 September 2001. Saat ini, penulis merupakan mahasiswi aktif pada Program Studi Informatika Fakultas Teknik Universitas Nurul Jadid Paiton Probolinggo. Penulis menyelesaikan Pendidikan dasar di Madrasah Ibtidaiyah Ihyaul Islam, kemudian melanjutkan Pendidikan di SMP Negeri 1 Pakuniran dan menyelesaikan Pendidikan menengah atas di SMA Negeri 1 Besuk. dan sebelumnya menempuh Pendidikan di SMAN 1 Besuk, Probolinggo. Penulis dapat dihubungi melalui e-mail : bismillahkhoir123@gmail.com

=======000=======

BIODATA PENULIS

Andi Wijaya, M. Kom
Dosen Prodi Rekayasa Perangkat Lunak Universitas Nurul Jadid

Penulis lahir di Bondowoso tanggal 03 Mei 1987. Penulis adalah Dosen Tetap pada Program Studi Rekayasa Perangkat Lunak Universitas Nurul Jadid. Penulis menyelesaikan pendidikan S1 di Sekolah Tinggi Teknologi Nurul Jadid Probolinggo pada tahun 2011 dan S2 di Universitas Dian Nuswantoro Semarang pada tahun 2013. Penulis menekuni bidang Teknologi Informasi dan Komunikasi, dan Pengembangan Perangkat Lunak. Penulis dapat dihubungi melalui e-mail: mr.andiwijaya@gmail.com

=======000=======

BIODATA PENULIS

Fathur Rizal, M. Kom
Dosen Prodi Teknik Informatika Universitas Nurul Jadid

Penulis lahir di Bondowoso tanggal 28 Agustus 1993. Penulis adalah Dosen Tetap pada Program Studi Teknik Informatika Universitas Nurul Jadid. Penulis menyelesaikan pendidikan S1 di Sekolah Tinggi Teknologi Nurul Jadid Probolinggo pada tahun 2015 dan S2 di Universitas Dian Nuswantoro Semarang pada tahun 2018. Penulis menekuni bidang Desain Sistem, Pengembangan Aplikasi Mobile dan Citra Digital. Penulis dapat dihubungi melalui e-mail: fatthurrizal@unuja.ac.id

=======000=======

TEORI UML DAN IMPLEMENTASI PRAKTEK

Panduan Untuk Pengembangan Perangkat Lunak

Buku ini bertujuan untuk memberikan pemahaman mendalam tentang Unified Modeling Language (UML) dan aplikasinya dalam pengembangan perangkat lunak. Disusun secara komprehensif, buku ini mencakup teori dasar UML, berbagai jenis diagram UML, dan cara mengintegrasikan serta mengimplementasikannya dalam proyek pengembangan sistem manajemen perpustakaan.
Bab-bab dalam buku ini meliputi:

1. Pendahuluan UML : Menjelaskan Sejarah, definisi, dan pentingnya UML dalam pemodelan perangkat lunak.
2. Keuntungan Pemodelan dengan UML : Menguraikan manfaat menggunakan UML dalam pengembangan perangkat lunak.
3. Memahami Konsep Dasar UML : Pengenalan elemen-elemen dasar UML seperti Things, Relationships, dan Diagrams.
4. Diagram UML : Penjelasan mendalam tentang berbagai diagram UML termasuk Use Case Diagram, Class Diagram, Sequence Diagram, Activity Diagram, dan lainnya.
5. Implementasi UML dalam Proyek Pengembangan Sistem Manajemen Perpustakaan : Studi Kasus yang mencakup Langkah-langkah pengembangan, implementasi diagram UML, dan evaluasi hasil.
6. Praktik Terbaik dalam Pemodelan dengan UML : Strategi dan tips untuk membuat diagram UML yang efektif dan kolaborasi tim dalam proses pemodelan.

BRAVO PRESS

IKAPI
IKATAN PENERBIT INDONESIA

ISBN 978-623-10-3680-3
9 786231 036803

bravopressindonesia@gmail.com
www.bravopress.id
bravo_press