BAB 2
LANDASAN TEORI

2.1. Gambaran Umum Perusahaan

Gambaran umum perusahaan menjelaskan mengenai sejarah perusahaan, profil perusahaan, visi dan misi, prinsip dan kebijakan, struktur organisasi dan *job description*. Berikut penjelasan dari masing-masing *point*.

2.1.1. Sejarah Perusahaan

PT. Imperial Putra Perdana di resmikan sebagai dealer resmi Honda mobil yang ke-88, pada tanggal 23 Maret 2011. PT. Imperial Putra Perdana merupakan dealer di bawah naungan PT. Honda Prospek Motor sebagai agen tunggal pemegang merk mobil Honda di Indonesia. PT. Imperial Putra Perdana atau lebih di kenal dengan sebutan Honda Ahmad Yani, merupakan perusahaan milik perseorangan dengan pemiliknya adalah Bp. Irwan Lesmana, dan menjabat sebagai komisaris utama. Dalam operasionalnya sehari – hari Honda Ahmad Yani di pimpin oleh seorang Presiden direktur yaitu Bp. Christian Lesmana. Honda Ahmad Yani merupakan dealer 3S1B, yaitu *sales, service*, *spare part*, dan *body paint.*

2.1.2. Profile Perusahaan

Kode Dealer	: 0208
Nama Dealer	: Honda Ahmad Yani
Perusahaan	: PT. Imperial Putra Perdana
Alamat	: Jl. Ahmad Yani 352, Cicadas, Bandung, Jawa Barat
Phone	: (+62)22-7217271

Fax	: (+62)22-721171
Bidang Usaha	: Sales, Service, Spare Part, dan Body Paint
Jumlah Karyawan	: 165 orang
Luas Lahan	: 3000 m^2
Luas Bangunan	: *Showroom* 420 m^2, *Workshop* 2000 m^2, lokasi *Spare Part* 145 m^2, dan lokasi *Body Paint* 1500 m^2
Hari Kerja	: 6 hari / minggu
Jam Kerja	: 08:00 – 19:00 WIB



Gambar 2.1 Dealer Honda Ahmad Yani

2.1.3. Visi dan Misi

Berikut penjelasan mengenai visi dan misi yang di miliki Honda.

a. Visi

   Menjadi perusahaan yang berorientasi penuh kepada konsumen dengan menciptakan tingkat pelayanan kualitas tinggi bagi perusahaan dan produk yang di hasilkan.

b. Misi

   1. Memberikan yang terbaik kepada konsumen dengan mengedepankan inovasi produk yang berteknologi tinggi, nyaman, dan ramah lingkungan.
   2. Mewujudkan sebuah perusahaan yang ideal dengan terus menerus berusaha menciptakan kondisi kerja yang harmonis dan turut berperan aktif dalam pengembangan industri otomotif Indonesia.
   3. Menghargai hak-hak setiap individu, sehingga seluruh karyawan merasa bangga bekerja dan menjadi bagian dari Honnda Ahmad Yani.

2.1.4. Prinsip dan Kebijakan Perusahaan

Honda memiliki prinsip dan kebijakan pada perusahaanya. Berikut prinsip dan kebijakaan Honda.

a. Prinsip Dasar

   1. Kami Percaya dengan nilai-nilai komitmen, integritas, loyalitas, trusthwortiness, jujur, disiplin, kompetensi, konsistensi, teamwork, dan komunikasi, maka Honda Ahmad Yani Bandung akan jaya.
   2. Kami menyadari untuk membangun bisnis yang tangguh membutuhkan semangat pantang menyerah, tidak pernah berhenti belajar, sukses dalah hak, gagal adalah pilihan, dan selalu optimis.

b. Prinsip Perusahaan

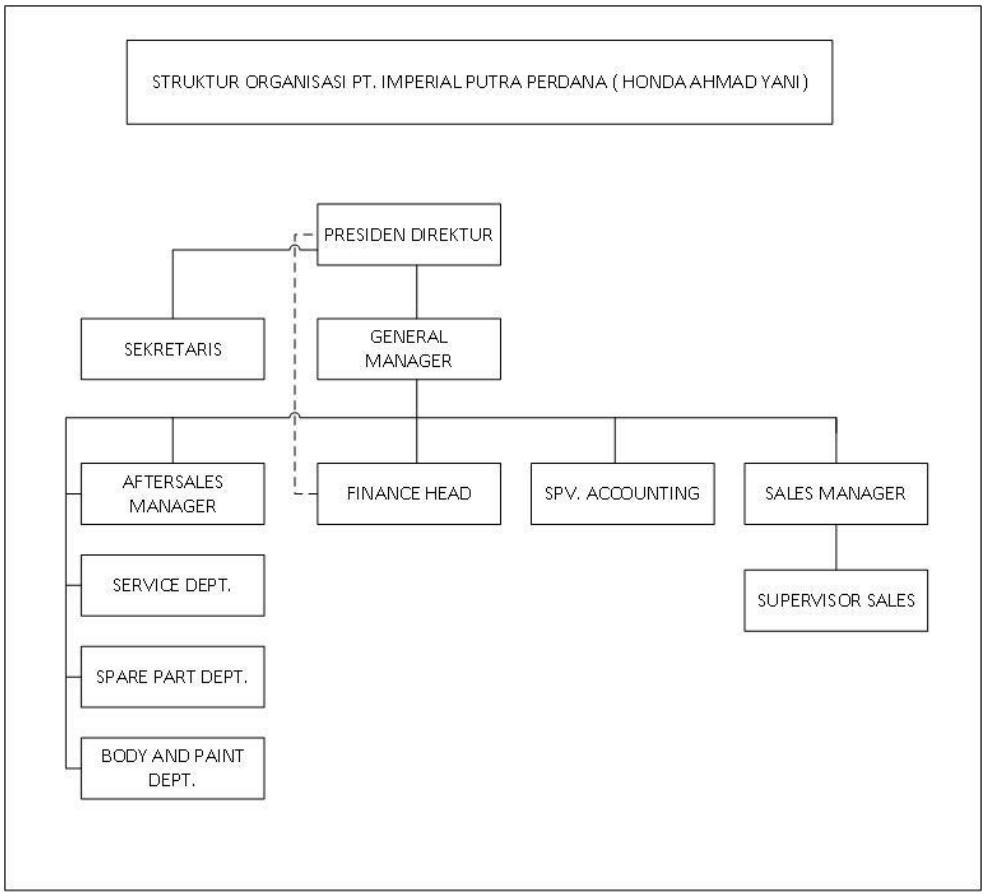
Demi menjaga opini publik, Honda Ahmad Yani berdedikasi untuk menyalurkan produk berkualitas terbaik dengan harga yang sesuai dengan kepuasan konsumen.

c. Kebijakan Manajemen

1. Selalu memiliki ambisi dan energis.
2. Menghargai teori, mengembangkan ide-ide baru, dan menggunakan waktu sebaik-baiknya.
3. Nikmati pekerjaanmu dan saling terbuka satu sama lainnya.
4. Selalu berusaha menjaga keharmonisan kinerja tim.
5. Selalu mengingat pentingnya riset dan kerja keras.

2.1.5. Struktur Organisasi

Gambar 2.2 menjelaskan gambar struktur organisasi di Honda.



Gambar 2.2 Struktur Organisasi Honda Ahmad Yani

2.1.6. Job Description

Wewenang dan tanggung jawab dalam struktur organisasi yang ada di Honda Ahmad Yani Bandung, adalah sebagai berikut :

1. Presiden Direktur

   Selaku pemimpin di Honda Ahmad Yani yang bertugas mengelola dan menerima laporan-laporan mengenai operasional perusahaan dan mengawasi perkembangan perusahaan dan menetapkan segala keputusan dan kebijakan perusahaan.

2. *General Manager*

   Berkedudukan sebagai pemimpin manajer yang berperan mewakili direktur dalam mengelola perusahaan dalam urusan *marketing* dan keuangan.

3. *Sales Manager*

   Bertanggung jawab terhadap *unit sales* dan *marketing* di perusahaan, membawahi *sales supervisor* terhadap operasional *sales.*

4. *Supervisor Sales*

   Bertanggung jawab terhadap penjualan *retail sales* yang dilakukan oleh *salesman* atau *salesgirl* di bawahnya. Melaporkan kegiatan kepada *sales manager*

5. *Aftersales Manager*

   Bertanggung jawab terhadap operasional *aftersales*, yaitu *service, spare part,* dan *body paint.* Melaporkan kegiatan *general manager* atau presiden direktur.

6. *Finance Manager*

   Bertanggung jawab terhadap pengelolaan keuangan dan operasional administrasi di perusahaan. Melaporkan kegiatan presiden direktur.

7. *Accounting Manager*

   Bertanggung jawab terjadap pengelolaan omset, *asset*, dan pelaporan pajak. Melaporkan kegiatan presiden direktur.

2.2. Android

Android merupakan sistem operasi yang di gunakan di smartphone. *Android* menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi yang dapat digunakan sebagai sarana promosi, sarana penyedia informasi atau bahkan untuk mengembangkan sebuah permainan *mobile.*

Fitur yang disediakan android, antara lain :

a. *Framework,*aplikasi yang memungkinkan untuk daur ulang dan penggantian komponen
b. *Browser,* alat yang sudah terintegrasi dengan android dan digunakan untuk sebagai *Webkit.*
c. Konektifitas, android mendukung berbagai macam konektifitas, seperti GSM(*Global System for Mobile Communications)*/EDGE(*Enchanced Data rates for GSM Evolution),* EV-DO(*Evolution-Data Optimized),* UMTS(*Universal Mobile Telecommunications System*), *Bluetooth* dan *Wi-Fi*(*Wireless Fidelity).*
d. *Message,* android mendukung pengiriman pesan dalam bentuk SMS(*Short Message Service*) dan MMS(*Multimedia Messaging Service*) dengan dukungan Java.
e. *Java, Software* yang dapat mengkompilasi dan akan mengeksekusi pada mesin *virtual dalvik*, yang merupakan implementasi dari *virtual mechine*(VM) yang dirancang khusus untuk perangkan *mobile*.
f. *Media Support*, merupakan alat pelengkap yang dapat mendukung format *multimedia*, seperti *audio*(MP3,MIDI,OGG,WAV),*text*(WORDX,TXT), *video*(MP4, 3GP, WMV, AVI), dan *images*(JPG,GIF,BMP,PNG).
g. *Hardware,* merupakan perangkat keras yang selalu ada pada media elektronik. Perangkat keras yang tersedia di smartphone antara lain, Kamera, GPS(*Global Positioning System*), *magnetometer,* GPU(*Graphic Processing Unit*), RAM(*Random Access Memory*), ROM(*Read Only Memory*), *storage, sound card, Wifi adapter, bluetooth adapter, simcard slot.*

h. *Touchscreen,* merupakan sistem yang sudah terintergrasi dengan android, yang memungkinkan pengguna untuk dapat menyentuh layar dan tidak memerlukan alat bantuan *controling*, seperti *keyboard* dan *mouse.*
i. Lingkungan pengembangan, seperti *debugging system* untuk membantu dalam mengembangkan sebuah aplikasi.

2.2.1. Arsitektur Android

Android memiliki beberapa arsitektur dan berikut penjelasan arsitektur android.

a. Linux Kernel, Fondasi platform android adalah kernel linux. Menggunakan kernel linux memungkinkan android untuk memanfaatkan fitur keamanan inti dan memungkinkan produsen perangkat untuk mengembangkan *driver* perangkat keras yang cukup dikenal.
b. *Hardware Abstraction Layer*(*HAL*), menyediakan antarmuka standar yang mengekspos kemampuan perangkat keras dari perangkat ke kerangka keja Java API yang lebih tinggi. HAL terdiri atas beberapa modul pustaka, masing-masing mengimplementasikan antarmuka untuk komponen perangkat keras tertentu, seperti modul kamera atau bluetooth. Bila API kerangka kerja melakukan panggilan untuk mengakses perangkat keras, sistem android memuat modul pustaka untuk komponen perangkat keras tersebut.
c. *Android Runtime*(ART), ART berguna untuk menjalankan beberapa mesin virtual pada perangkat bermemori rendah dengan mengeksekusi file DEX, format bytecode yang didesain khusus untuk Android yang dioptimalkan untuk footprint memori minimal. Buat rantai aplikasi, misalnya Jack, mengumpulkan sumber Java ke bytecode DEX, yang dapat berjalan pada platform Android.

Beberapa fitur utama ART mencakup:

1. Kompilasi mendahului waktu (AOT) dan tepat waktu (JIT)

2. Pengumpulan sampah (GC) yang dioptimalkan
3. Dukungan debug yang lebih baik, mencakup profiler sampling terpisah, pengecualian diagnostik mendetail dan laporan kerusakan dan kemampuan untuk mengatur titik pantau guna memantau bidang tertentu.

Sebelum ke Android versi 5.0 (API level 21), Dalvik adalah waktu proses Android. Jika aplikasi Anda berjalan baik pada ART, semestinya berfungsi baik juga pada Dalvik, tetapi mungkin tidak sebaliknya. Android juga menyertakan serangkaian pustaka waktu proses inti yang menyediakan sebagian besar fungsionalitas bahasa pemrograman Java, termasuk beberapa fitur bahasa Java 8, yang digunakan kerangka kerja Java API.

d. Pustaka C/C++ Asli, Banyak komponen dan layanan sistem Android inti seperti ART dan HAL dibuat dari kode asli yang memerlukan pustaka asli yang tertulis dalam C dan C++. Platform Android memungkinkan kerangka kerja Java API mengekspos fungsionalitas beberapa pustaka asli pada aplikasi. Misalnya, Anda bisa mengakses OpenGL ES melalui kerangka kerja Java OpenGL API Android guna menambahkan dukungan untuk menggambar dan memanipulasi grafik 2D dan 3D pada aplikasi Anda.
e. Kerangka Kerja Java API, Keseluruhan rangkaian fitur pada Android OS tersedia melalui API yang ditulis dalam bahasa Java. API ini membentuk elemen dasar yang di perlukan untuk membuat aplikasi Android dengan menyederhanakan penggunaan kembali inti, komponen dan layanan sistem modular, yang menyertakan berikut ini:
    1. Tampilan Sistem yang kaya dan luas bisa di gunakan untuk membuat UI aplikasi, termasuk daftar, kisi, kotak teks, tombol, dan bahkan browser web yang dapat disematkan.
    2. Pengelola Sumber Daya, memberikan akses ke sumber daya bukan kode seperti string yang dilokalkan, grafik, dan file

layout. Pengelola Notifikasi yang mengaktifkan semua aplikasi guna menampilkan lansiran khusus pada bilah status.

3. Pengelola Aktivitas yang mengelola daur hidup aplikasi dan memberikan back-stack navigasi yang umum.
4. Penyedia Materi yang memungkinkan aplikasi mengakses data dari aplikasi lainnya, seperti aplikasi Kontak, atau untuk berbagi data milik sendiri.
5. Developer memiliki akses penuh ke API kerangka kerja yang sama dengan yang digunakan oleh aplikasi sistem Android.

f. Aplikasi Sistem. Android dilengkapi dengan serangkaian aplikasi inti untuk email, perpesanan SMS, kalender, menjelajahi internet, kontak, dll. Aplikasi yang disertakan bersama platform tidak memiliki status khusus pada aplikasi yang ingin dipasang pengguna. Jadi, aplikasi pihak ketiga dapat menjadi browser web utama, pengolah pesan SMS atau bahkan keyboard utama (beberapa pengecualian berlaku, seperti aplikasi Settings sistem).

Aplikasi sistem berfungsi sebagai aplikasi untuk pengguna dan memberikan kemampuan kunci yang dapat diakses oleh developer dari aplikasi mereka sendiri. Misalnya, jika aplikasi Anda ingin mengirimkan pesan SMS, Anda tidak perlu membangun fungsionalitas tersebut sendiri—sebagai gantinya Anda bisa menjalankan aplikasi SMS mana saja yang telah dipasang guna mengirimkan pesan kepada penerima yang Anda tetapkan.[1][9].

2.3. Multimedia

Multimedia adalah penggunaan komputer untuk menyajikan dan menggabungkan teks, suara, gambar, animasi, *audio* dan *video* dengan alat bantu (*tool*) dan koneksi (*link*) sehingga pengguna dapat melakukan navigasi, berinteraksi, berkarya dan berkomunikasi. *Multimedia* sering digunakan dalam dunia informatika. Selain dari dunia informatika, *multimedia* juga diadopsi oleh dunia *game*, dan juga untuk membuat *website*.

Multimedia dimanfaatkan juga dalam dunia pendidikan dan bisnis. Di dunia pendidikan, *multimedia* digunakan sebagai media pengajaran, baik dalam kelas maupun secara sendiri-sendiri atau otodidak. Di dunia bisnis, *multimedia* digunakan sebagai media profil perusahaan, profil produk, media promosi, bahkan sebagai media kios informasi dan pelatihan dalam sistem *e-learning*.

2.4. SolidWork

Solidwork merupakaan program yang dirancang untuk keperluan bidang teknik seperti desain produk, desain mesin, desain *Mold*, dan keperluan teknik lainnya. *SolidWork* merupakan perangkat lunak milik CAD yang memiliki featured-based, pametric solid modeling dan bergerak pada pemodelan 3 dimensi.[10]

2.5. Animasi

Animasi adalah gambar begerak berbentuk dari sekumpulan objek (gambar) yang disusun secara beraturan mengikuti alur pergerakan yang telah ditentukan pada setiap pertambahan hitungan waktu yang terjadi. Gambar atau objek yang dimaksud dalam definisi di atas bisa berupa gambar manusia, hewan, maupun tulisan. Pada proses pembuatannyam sang pembuat animasi atau yang lebih dikenal dengan animator harus menggunakan logika berfikir untuk menentukan alur gerak suatu objek dari keadaan awal hingga keadaan akhir objek tersebut. Perencanaan yang matang dalam perumusan alur gerak berdasarkan logika yang tepat akan menghasilkan animasi yang menarik untuk disaksikan.

Animasi terdiri jadi beberapa bagian yaitu animasi 3D dan animasi 2D:

1. Animasi 3D. Objek animasi yang berada pada ruang tiga dimensi(3D). Objek animasi ini dapat dirotasi dan berpindah seperti objek riil. Proses pembuatan grafis komputer 3D dapat dibagi secara sekuens menjadi 3 fase dasar:
   a. *Modeling* : Proses menyusun bentuk sebuah objek dengan membuat kontur yang luas dan struktur dari objek 3D dan adegan.

b. *Animation* : Mendefinisikan perpindahan objek.
c. *Rendering* : Proses mengeneralisasi sebuah gambar dari sebuah model dengan memberikan atribut objek seperti warna, tekstur permukaan dan kadar transparansi dengan menggunakan program komputer.

Modelling. Hampir seluruh model 3D dapat dibagi menjadi 2 kategori :

a. *Solid* : Model ini mendefinisikan volume dari objek yang mereka tampilkan (seperti batu). Model ini lebih realistis, namun lebih sulit untuk dibangun, Model solid kebanyakan digunakan untuk simulasi nonvisual seperti simulasi medis dan teknik, untuk CAD dan aplikasi visual khusus seperti ray tracing dan konstruksi geometri solid.
b. *Shell / Boundary* :Modeil ini menggambarkan permukaan seperti batas objek, bukan volumenya (seperti cangkang yang sangat tipis). Model ini lebih mudah dikerjakan daripada model solid. Kebanyakan model visual yang digunakan dalam game dan film adalah model shell.

Animation. Ada beberapa teknik untuk membuat animasi :

a. *Traditional Animation***:** Proses yang digunakan untuk sebagian besar film animasi pada abad ke-20. Setiap frame dari film animasi tradisional adalah foto dari gambar, yang terlebih dahulu digambar di kertas. Untuk membuat ilusi gerakan, masing-masing gambar dibuat sedikit berbeda dengan gambar yang sebelumnya.
b. *Full Animation***:** Merujuk pada proses produksi film animasi tradisional berkualitas tinggi, dimana secara teratur menggunakan gambar yang detail dan gerakan yang masuk akal. Film animasi penuh dapat dibuat dengan berbagai cara, dari animasi yang realistis hingga yang lebih mengarah ke kartun.
c. *Limited Animation***:** Menggunakan detail yang kurang rinci dan/atau gambar yang lebih stylist dan metode perpindahan. Penggunaannya telah menghasilkan animasi berbiaya efektif untuk media seperti televise dan kemudian internet.

d. *Rotoscoping***:** Merupakan teknik dimana animator melacak gerakan live-action, frame demi frame. Sumber film dapat secara langsung digandakan dari aktor ke gambar animasi.
e. *Live-action/animation***:** Teknik yang mengkombinasikan karakter yang digambar tangan menjadi gambar live-action.

Rendering. Beberapa teknik berhubungan langsung dengan algoritma, ketika digunakan bersama-sama.

a. *Shading* : Bagaimana warna dan tingkat kecerahan permukaan dipengaruhi oleh pencahayaan.
b. *Texture-mapping* : Metode untuk mengaplikasikan detail pada permukaan.
c. *Bump-mapping* : Metode simulasi bump skala kecil pada permukaan.
d. *Fogging/participating medium* : Bagaimana cahaya meredup ketika melewati atmosfir yang tidak bersih atau udara.
e. *Shadows* : Pengaruh menghalangi cahaya.
f. *Soft Shadows* : Variasi tingkat gelap yang disebabkan oleh terhalanginya sumber cahaya.
g. *Reflection* : Seperti cermin atau mengkilap.
h. *Transparency* (*optics*), *transparency* (*graphic*) atau *opacity* : Transmisi yang tajam dari cahaya yang melalui objek solid.
i. *Transluency* : Transmisi cahaya melalui objek solid yang sangat menyebar.
j. *Refraction* : Pembengkokan cahaya yang terkait dengan transparansi.
k. *Diffraction* : Pembengkokan, penyebaran dan interferensi cahaya yang melewati suatu objek.
l. *Indirect illumination* : Permukaan diterangi cahaya yang dipantulkan oleh permukaan lain, bukan langsung daru sumber cahaya.
m. *Caustics* : Bentuk pencahayaan tidak langsung, pantulan cahaya dari sebuah objek mengkilat atau memfokuskan cahaya melalui objek transparan untuk menghasilkan cahaya ke objek lain.

n. *Depth of field* : Objek tampil buram atau tidak focus ketika terlalu jauh di depan atau di belakang objek dalam focus.
o. *Motion blur* : Objek yang tampil buram karena gerak kecepatan tinggi atau gerakan kamera.
p. *Non-photorealistic rendering* : Rendering gambar dengan gaya artistic, agar terlihat seperti lukisan atau gambar.

2. Animasi 2D. Penciptaan gambar bergerak dalam lingkungan dua dimensi(2D). Hal ini dilakukan dengan urutan gambar berturut-turut, atau "frame", yang mensimulasikan gerak oleh setiap gambar menunjukkan berikutnya dalam perkembangan bertahap langkah-langkah.
   2 tipe dari 2D animation :
   a. *Cel* animasi
   b. *Path* animasi

a. *Cel* animasi berasal dari kata "*celluloid*", yaitu bahan dasar dalam pembuatan animasi jenis ini ketika tahun-tahun awal adanya animasi. Animasi *cel* merupakan lembaran-lembaran yang membentuk animasi tunggal, masing-masing *cel* merupakan bagian yang terpisah sebagai objek animasi. misalnya ada tiga buah animasi *cel*, *cel* pertama berisi satu animasi karakter, *cel* kedua berisi animasi karakter lain, dan *cel* terakhir berisi latar animasi. Ketiga animasi *cel* ini akan disusun berjajar, sehingga ketika dijalankan animasinya secara bersamaan, terlihat seperti satu kesatuan. Contoh animasi jenis ini adalah *film* kartun seperti Tom and Jerry, Mickey Mouse dan Detectif Conan.
b. *Path* animasi adalah animasi dari objek yang gerakannya mengikuti garis lintasan yang sudah ditentukan. Contoh animasi jenis ini adalah animasi kereta api yang bergerak mengikuti lintasan rel. Biasanya dalam animasi *path* diberi perulangan animasi, sehingga animasi terus berulang hingga mencapai kondisi tertentu. Dalam Macromedia Flash, animasi jenis ini didapatkan dengan teknik animasi *path*, teknik ini menggunakan layer tersendiri yang didefinisikan sebagai lintasan gerakan objek.

2.6. Pengujian Black Box

Pengujian *black box* yaitu pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian *black box* dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Di kasus ini pengujian yang diambil dari salah satu bagian pengujian *black box* adalah alpha testing dan beta testing.[8]

1. Alpha Testing

Pengujian *Alpha* adalah salah satu strategi pengujian perangkat lunak yang paling umum digunakan dalam pengembangan perangkat lunak, hal ini khusus digunakan oleh organisasi pengembangan produk dengan tujuan agar system yang dikembangkan terhindar dari cacat atau kegagalan penggunaan.

Pengujian *alpha* berlangsung di situs pengembang oleh tim internal, sebelum rilis kepada pelanggan eksternal. Agar nantinya ketika pelanggan menggunakan system ini tidak kecewa karena masalah cacat atau kegagalan aplikasi. Pengujian ini dilakukan tanpa keterlibatan tim pengembangan.

1. Tes ini berlangsung di situs pengembang. Pengembang mengamati penggunaan aplikasi oleh pengguna selanjutnya pengguna mencatat temuan yang terjadi dari kecacatan aplikasi.
2. Pengujian *Alpha* adalah pengujian dari aplikasi saat pembangunan adalah tentang untuk menyelesaikan. perubahan desain kecil masih dapat dibuat sebagai hasil dari pengujian *alpha.*
3. Pengujian *Alpha* biasanya dilakukan oleh kelompok yang independen dari tim desain, tim pengembang tapi masih dalam perusahaan, misalnya di-rumah insinyur pengujian perangkat lunak, atau insinyur perangkat lunak QA.

Pengujian *Alpha* adalah pengujian akhir sebelum perangkat lunak ini diluncurkan untuk pengguna secara umum. Ini memiliki dua fase:

1. Pada tahap pertama dari pengujian alpha, perangkat lunak diuji oleh pengembang di lingkungan *internal developer*. Mereka

menggunakan perangkat lunak *debugger*, atau *debugger hardware-assisted*. Tujuannya adalah untuk menangkap bug dengan cepat.

2. Pada tahap kedua pengujian *alpha*, *software* ini diserahkan kepada *staff* QA *software*, untuk pengujian tambahan dalam lingkungan yang mirip dengan penggunaan yang dimaksudkan. Hal ini untuk mensimulasikan suasana atau lingkungan pengujian yang sebenarnya sehingga ketika *system* tersebut dipasang, sudah tidak terjadi kegagalan maupun cacat *system* secara *real.*

Pengujian *Alpha* disimulasikan atau pengujian operasional yang sebenarnya dengan potensi pengguna / pelanggan atau tim uji independen di situs pengembang. pengujian *alpha* sering digunakan untuk perangkat lunak *off-the-rak* sebagai bentuk pengujian penerimaan *internal* sebelum s*oftware* pergi ke pengujian *beta*.[5]

2. Beta Testing

Pengujian *beta* juga dikenal sebagai pengujian pengguna berlangsung di lokasi pengguna akhir oleh pengguna akhir untuk memvalidasi kegunaan, fungsi, kompatibilitas, dan uji reliabilitas dari *software* yang dibuat.

Aktifitas pengujian beta menambah nilai siklus hidup pengembangan perangkat lunak karena memungkinkan pelanggan sebenarnya kesempatan untuk memberikan masukan ke dalam desain, fungsi, dan kegunaan dari produk. Masukan ini tidak hanya penting untuk keberhasilan produk tetapi juga investasi ke produk masa depan ketika data yang dikumpulkan dikelola secara efektif.

Hal ini juga dikenal sebagai uji lapangan. Ini terjadi di lokasi pelanggan. Ini mengirimkan sistem untuk pengguna yang menginstal dan menggunakannya di bawah kondisi kerja dunia nyata.

Tes *beta* merupakan tahap kedua dari pengujian perangkat lunak di mana pengguna mencoba produk. Awalnya, tes *alpha* berarti tahap pertama pengujian dalam proses pengembangan perangkat lunak. Tahap pertama

meliputi unit *testing*, pengujian komponen, dan pengujian sistem. pengujian *beta* dapat dianggap "pengujian pra-rilis artinya sebelum produk tersebut dilempar ke pasaran maka harus dipastikan dari sisi pelanggan bahwa perangkat lunak tersebut terbebas dari cacat atau kegagalan .

Tujuan dari pengujian *beta* adalah untuk menempatkan aplikasi Anda di tangan pengguna yang sebenarnya yang berada di luar tim teknik Anda untuk menemukan setiap kekurangan atau masalah dari perspektif pengguna akhir.

Ada sejumlah faktor yang tergantung pada keberhasilan pengujian *beta*:

1. Uji Biaya
2. Jumlah Peserta Uji
3. Pengiriman
4. Durasi Uji
5. cakupan Demografi

Pengembang merilis baik beta tertutup atau beta terbuka;

1. Versi beta tertutup dilepaskan untuk kelompok memilih individu untuk uji pengguna dan sesuai undangan saja, sedangkan
2. Pengujian *beta* terbuka berasal dari kelompok yang lebih besar dari masyarakat umum dan siapapun yang tertarik. Penguji melaporkan bug yang mereka temukan, dan kadang-kadang menyarankan fitur tambahan yang mereka pikir harus tersedia dalam versi *final*.[4]

2.7. Unity 3D

Unity 3D merupakan sebuah *game engine*.*Game engine* ini di bangun atas dasar kepedulian terhadap *indie developer* yang tidak bisa membeli game engine karena terlalu mahal. Unity memungkinkan seseorang untuk membuat sebuah game 3D dengan mudah dan cepat. Secara awal, Unity telah di atur untuk pembuatan permainan bergenre *First Person Shooting*(FPS), namun unity juga bisa di gunakan untuk membuat game bergenre Role Playing Game(RPG), dan Real Time Strategy(RTS). Selain itu, unity merupakan

sebuah engine multiplatform untuk berbagai platform , seperti Windows, Mac, Android, iOS, PS3 dan Wii.[2]

2.8. Photoshop

Adobe Photoshop, atau biasa disebut Photoshop, adalah perangkat lunak editor citra buatan Adobe *Systems* yang dikhususkan untuk pengeditan foto/gambar dan pembuatan efek.. Versi kedelapan aplikasi ini disebut dengan nama Photoshop CS (Creative Suite), versi sembilan disebut Adobe Photoshop CS2, versi sepuluh disebut Adobe Photoshop CS3 , versi kesebelas adalah Adobe Photoshop CS4 , versi keduabelas adalah Adobe Photoshop CS5 , dan versi terbaru adalah Adobe Photoshop CC[6].

2.9. Blender 3D

Blender adalah perangkat penciptaan 3D gratis dan open source. Blender mendukung keseluruhan pemodelan pipa 3D, *rigging*, animasi, simulasi, *rendering*, compositing dan *motion tracking*, bahkan *editing video* dan pembuatan game. Blender memiliki beberapa *toolbar* yang umum dan harus di ketahui, berikut adalah *toolbar* blender yang umum.

1. *Tool Self*. Berisi *tools* yang dikelompokan berdasarkan jenisnya. *Tools* ini bisa disembunyikan(*Keyboard* T).
2. *Properties.* Berisi parameter umum objek, *grid*, dan tampilan *viewport.*
3. *Info Window*. Berisi menu, jenis *layout* antar muka, informasi *scane*, dan *render engine.*
4. *Outliner Window*. Menampilkan semua objek berikut settingannya.
5. *Editor Window*. Kadang disebut sebagai *buttons window* atau *properties window.*
6. 3D *View Window*. Disebut juga sebagai *viewport*
7. *Timeline Window* dan *Playback Window* digunakan saat pembuatan animasi.[7]

Blender memiliki beberapa fitur, seperti :

a. Rendering Photorealistic

   Blender sekarang memiliki mesin *rendering* yang baru dan tidak bias yang disebut Siklus yang menawarkan rendering ultra-realistis yang menakjubkan.

   Rows built-in menawarkan mesin rendering:

   1. GPU & *rendering* CPU
   2. Pratinjau *viewport* waktu nyata
   3. Dukungan pencahayaan HDR
   4. Lisensi Permisif untuk menghubungkan dengan perangkat lunak eksternal

b. Pemodelan Cepat

   Kumpulan alat pemodelan komprehensif dari Blender membuat, mengubah dan mengedit model Anda menjadi mudah.

   Alat pemodelan Blender meliputi:

   1. Pintasan *keyboard* untuk alur kerja yang cepat
   2. Dukungan N-Gon
   3. Tepi meluncur, runtuh dan membubarkan
   4. *Grid* dan *Bridge* terisi
   5. Python *scripting* untuk *custom tools* dan add-on

c. Bahan Realistis

   Dengan mesin *rendering* baru milik Blender, kemungkinan material tidak ada habisnya.

2.10. UML (*Unified Modeling Language*)

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek.

2.10.1. Diagram UML

UML memiliki 13 jenis diagram, yaitu :

Structural UML diagrams

1. *Class diagram*
2. *Package diagram*
3. *Object diagram*
4. *Component diagram*
5. *Composite structure diagram*
6. *Deployment diagram*

Behavior UML diagrams

1. *Activity diagram*
2. *Sequence diagram*
3. *Usecase diagram*
4. *State diagram*
5. *Communication diagram*
6. *Interaction overview diagram*
7. *Timing diagram*

Berikut Penjelasan satu persatu dari *structural* UML diagram.

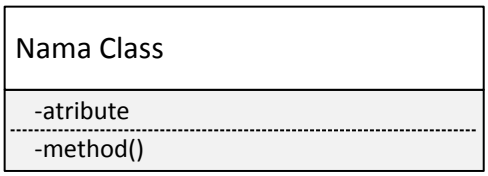
1. *Class Diagram*

Class diagram merupakan model struktur statis dari sebuah sistem. Menunjukan hubungan antara *class*, objek, atribut.dan method.

Beberapa *class* diagram simbol umum dan Notasi,yaitu :

a. *Class*

Class mepresentasikan abstrak dari entitas dengan karakteristik. Atribute menerangkan atribute apa saja yang ada dalam *class* dan method menjelaskan operasi apa saja yang ada pada *class.*



Gambar 2.3 Class

b. Visibilitas

Gunakan penanda visibilitas untuk menandakan siapa yang dapat mengakses informasi yang terdapat dalam kelas. Visibilitas pribadi, dilambangkan dengan tanda -, menyembunyikan informasi dari partisi luar kelas. Visibilitas publik, dilambangkan dengan tanda +, memungkinkan semua kelas lain melihat informasi yang ditandai. Visibilitas terlindungi, dilambangkan dengan tanda #, memungkinkan kelas anak mengakses informasi yang mereka warisi dari kelas induk.

Tabel 2.1 Visibilitas

Tanda	Keterangan
+	Public
-	Private
#	Protect
~	Package

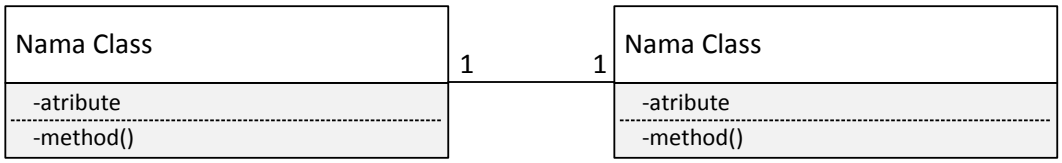
c. Asosiasi

Asosiasi mewakili hubungan statis antar kelas. Tempatkan nama asosiasi di atas, pada, atau di bawah garis asosiasi. Gunakan panah yang terisi untuk menunjukkan arah hubungan. Tempatkan peran di dekat akhir sebuah asosiasi. Peran mewakili cara dua kelas saling melihat.

Gambar 2.4 Simbol Asosiasi

d. Kardinalits

Tempatkan notasi kardinalitas di dekat ujung asosiasi. Simbol-simbol ini menunjukkan jumlah instance dari satu kelas yang terhubung ke satu *instance* kelas lainnya. Misalnya, satu perusahaan akan memiliki satu atau lebih karyawan, namun setiap karyawan hanya bekerja untuk satu perusahaan.



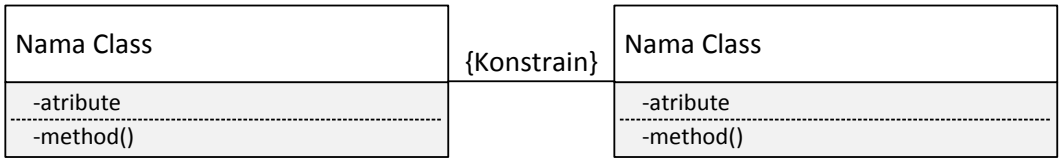
Gambar 2.5 Kardinalitas

Tabel 2.2 Jenis Multiplicity

Indikator	Keterangan
0..1	Kosong atau satu
1	Satu
0..*	Kosong atau banyak
1..*	Satu atau banyak
n	Sebanyak n (n bernilai lebih dari satu)
0..n	Kosong ke n (n bernilai lebih dari satu)
1..n	Satu ke n (n bernilai lebih dari satu)

e. Konstrain

Tempatkan hambatan di dalam kurung kurawal {}.

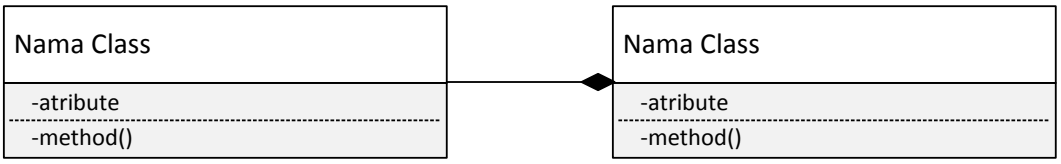


Gambar 2.6 Konstrain

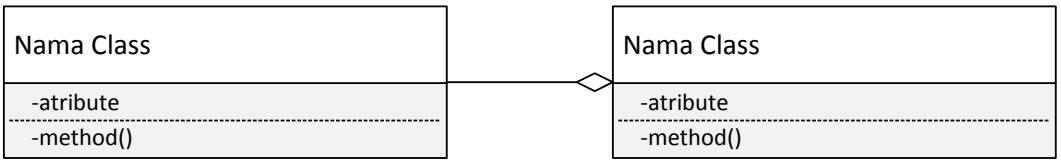
f. Komposisi dan Aggregasi

Komposisi adalah jenis agregasi khusus yang menunjukkan kepemilikan yang kuat antara Kelas A, keseluruhan, dan Kelas B, bagiannya. Ilustrasikan komposisi dengan berlian yang terisi.

Gunakan berlian berongga untuk mewakili hubungan agregasi sederhana, di mana kelas "keseluruhan" memainkan peran yang lebih penting daripada kelas "bagian", namun kedua kelas tidak saling bergantung satu sama lain. Berlian berakhir pada kedua komposisi dan hubungan agregasi mengarah ke kelas "keseluruhan" (yaitu agregasi).



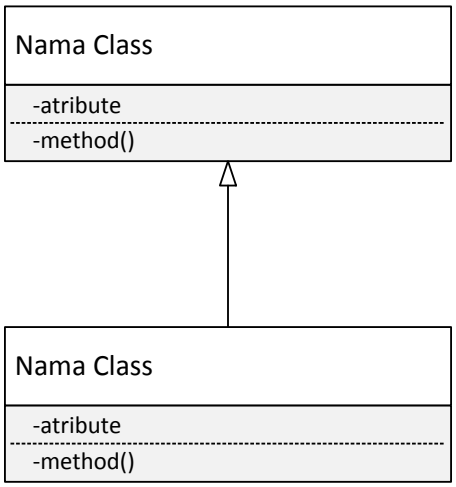
Gambar 2.7 Komposisi



Gambar 2.8 Aggregasi

g. Generalisasi

Generalisasi adalah nama lain untuk warisan atau hubungan "adalah". Ini mengacu pada hubungan antara dua kelas di mana satu kelas adalah versi khusus yang lain. Misalnya, Honda adalah jenis mobil. Maka kelas Honda akan memiliki hubungan generalisasi dengan mobil kelas.

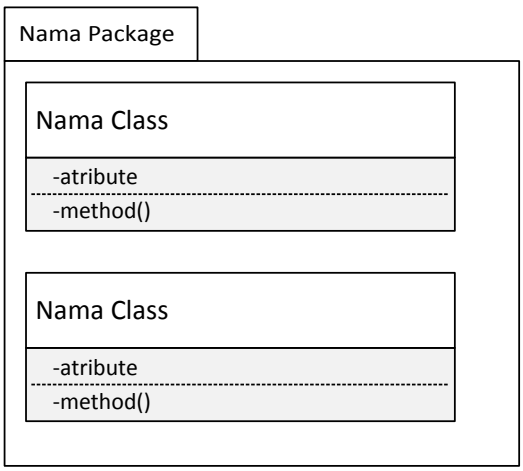


Gambar 2.9 Generalisasi

Dalam contoh pengkodean kehidupan nyata, perbedaan antara warisan dan agregasi bisa membingungkan. Jika Anda memiliki hubungan agregasi, keseluruhan (keseluruhan) hanya dapat mengakses fungsi publik kelas bagian. Di sisi lain, warisan memungkinkan kelas mewarisi untuk mengakses fungsi publik dan protect dari superclass.

2. *Package Diagram*

Diagram paket adalah subset dari diagram kelas, namun pengembang terkadang memperlakukannya sebagai teknik yang terpisah. Diagram paket mengatur elemen suatu sistem ke dalam kelompok yang terkait untuk meminimalkan ketergantungan antar paket.



Gambar 2.10 Package Diagram

3. *Object Diagram*

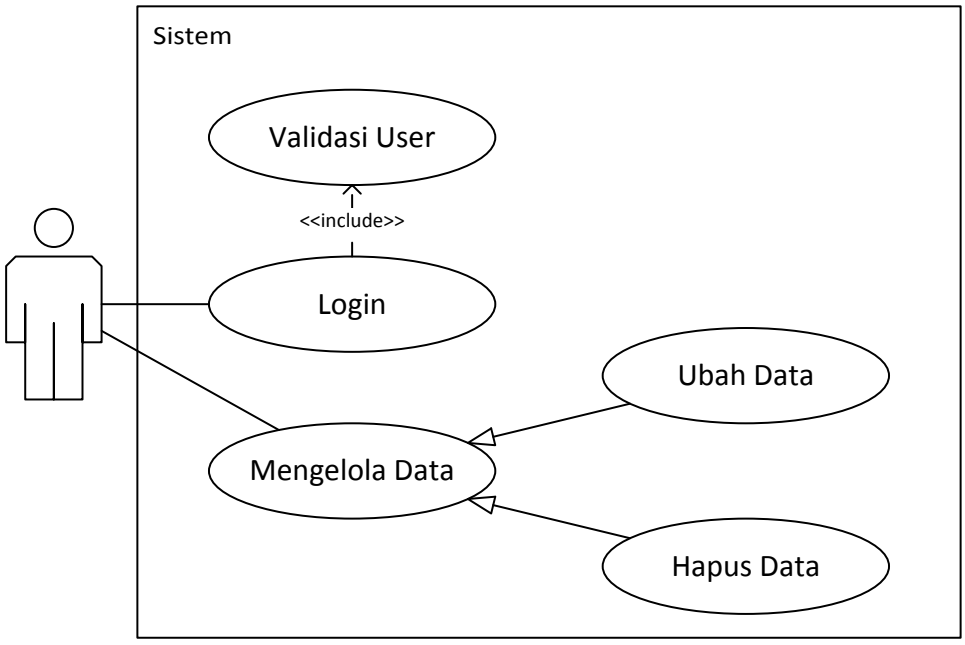
   Diagram objek menggambarkan struktur statis suatu sistem pada waktu tertentu. Mereka dapat digunakan untuk menguji diagram kelas untuk akurasi.

4. *Composite Structure Diagram*

   Diagram struktur komposit menunjukkan bagian dalam sebuah kelas.

5. *Use Case Diagram*

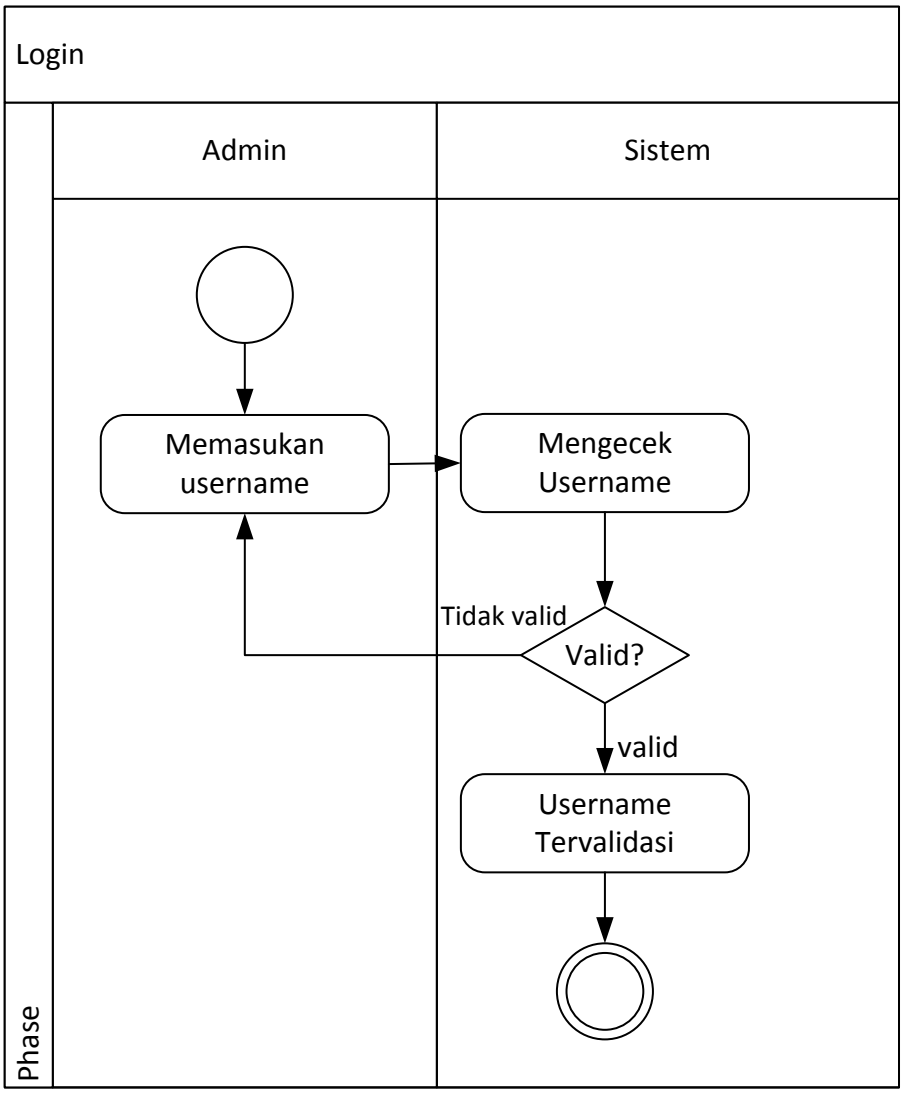
   Use case diagram memodelkan fungsionalitas suatu sistem dengan menggunakan aktor dan use case.



Gambar 2.11 Usecase Diagram

6. *Actifity Diagram*

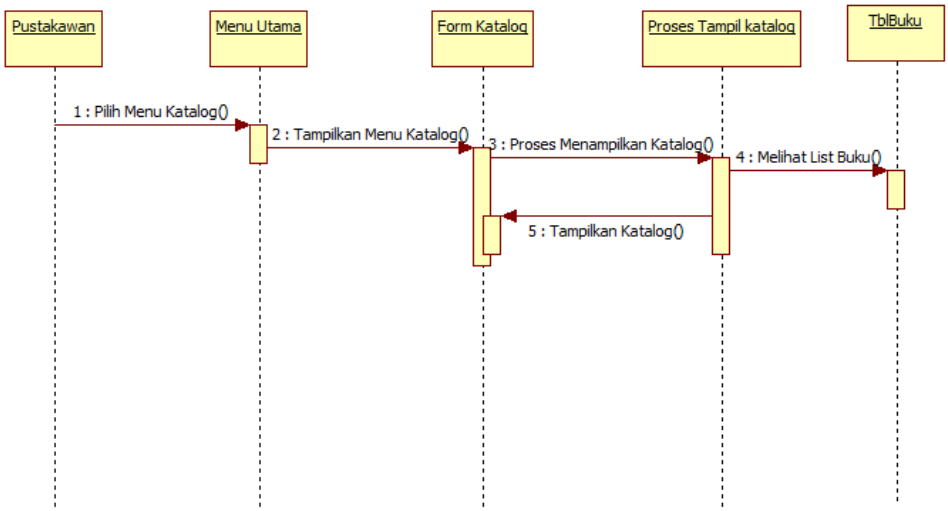
   Diagram aktivitas menggambarkan sifat dinamik suatu sistem dengan memodelkan aliran kontrol dari aktivitas ke aktivitas. Aktivitas merupakan operasi pada beberapa kelas dalam sistem yang menghasilkan perubahan keadaan sistem. Biasanya, diagram aktivitas digunakan untuk memodelkan alur kerja atau proses bisnis dan operasi internal.



Gambar 2.12 Activity Diagram

7. *Sequence Diagram*

   Diagram urutan menggambarkan interaksi antar kelas dalam hal pertukaran pesan dari waktu ke waktu.



Gambar 2.13 Sequence Diagram

8. *Interaction Overview Diagram*

   Diagram interaksi *overview* adalah kombinasi diagram aktivitas dan urutan. Mereka memodelkan urutan tindakan dan membiarkan Anda mendekonstruksi interaksi yang lebih kompleks menjadi kejadian yang mudah dikendalikan. Anda harus menggunakan notasi yang sama pada diagram interaksi *overview* yang akan Anda lihat pada diagram aktivitas.

9. *Timing Diagram*

   Diagram waktu adalah jenis diagram interaksi perilaku atau interaksi UML yang berfokus pada proses yang terjadi selama periode waktu tertentu. Mereka adalah contoh khusus dari diagram urutan, kecuali waktu yang ditunjukkan meningkat dari kiri ke kanan, bukan di bagian atas.

10. Communication Diagram

Diagram komunikasi memodelkan interaksi antar objek secara berurutan. Mereka menggambarkan struktur statis dan perilaku dinamis sebuah sistem. Dalam banyak hal, diagram komunikasi adalah versi sederhana dari diagram kolaborasi yang diperkenalkan di UML 2.0.

11. State Diagram

Diagram Statechart, sekarang dikenal sebagai diagram mesin negara dan diagram keadaan menggambarkan perilaku dinamis suatu sistem dalam menanggapi rangsangan eksternal. Diagram keadaan sangat berguna dalam memodelkan objek reaktif yang keadaannya dipicu oleh kejadian tertentu.

12. Component Diagram

Diagram komponen menggambarkan pengorganisasian komponen perangkat lunak fisik, termasuk kode sumber, kode *run-time* (*biner*), dan *file executable*.

Beberapa hal umum komponen diagram :

h. Komponen

Komponen adalah blok unit logis dari sistem, abstraksi yang sedikit lebih tinggi daripada kelas. Ini direpresentasikan sebagai persegi panjang dengan persegi panjang yang lebih kecil di sudut kanan atas dengan tab atau kata yang tertulis di atas nama komponen untuk membantu membedakannya dari kelas.

i. Interface

Antarmuka (lingkaran kecil atau semi lingkaran pada tongkat) menggambarkan sekelompok operasi yang digunakan (wajib) atau dibuat (disediakan) oleh komponen. Lingkaran penuh mewakili antarmuka yang dibuat atau disediakan oleh komponen. Semi-lingkaran mewakili antarmuka yang dibutuhkan, seperti masukan seseorang.

j. Dependensi

Gambarkan dependensi antar komponen dengan menggunakan panah putus-putus.

k. Port

Port diwakili menggunakan bujur sangkar di sepanjang tepi sistem atau komponen. Port sering digunakan untuk membantu mengekspos interface interface yang dibutuhkan dan disediakan.

13. Deployment Diagram

Diagram penyebaran menggambarkan sumber daya fisik dalam sistem, termasuk node, komponen, dan koneksi.[3][8].