BAB III

OBJEK DAN METODE PENELITIAN

3.1 Objek Penelitian

Objek penelitian dari penulisan skripsi ini adalah tentang Sistem pembelajran jarak jauh yang ada di SMA N 1Ciwidey

3.1.1 Sejarah Singkat SMAN 1 CIWIDEY

SMA Negeri 1 Ciwidey mulai menerima siswa baru pada Tahun Pelajaran 1987-1988 sebagai FILIAL (Kelas Jauh) dari SMA Negeri 1 Soreang, yang pada saat itu dipimpin oleh Bapak Lili Mulya Setia BA, selaku kepala SMA Negeri 1 Soreang sekaligus sebagai pengelola SMA Negeri 1 Ciwidey.

Pada Tahun Pertama, SMA Negeri 1 Ciwidey menerima siswa baru kelas 1 sebanyak 4 kelas (186 siswa) dan menumpang di SD Ciwidey III, sedangkan guru-gurunya 90% guru SMAN Soreang.

Tahun kedua, yakni tahun pelajaran 1988-1989 lokasi sekolah pindah ke SMPN 1 Ciwidey dan menerima siswa baru kelas 1 sebanyak 4 kelas (187 siswa) sedangkan siswa yang naik dan melanjutkan di kelas 2 tinggal 3 kelas (141 siswa). Pada tahun ini bapak Lili Mulya Setia BA meninggal dunia, dan Bapak DRS. Tadjudin Noor Rosadi menggantikannya sebagai kepala SMAN 1 Soreang sekaligus sebagai pengelola SMA Negeri 1 Ciwidey. Jumlah Guru dan TU sebanyak 30 orang.

Tahun ketiga, yaitu tahun pelajaran 1989-1990 lokasi sekolah berpindah lagi ke SD Panundaan dan menerima siswa baru kelas 1 sebanyak 4 kelas (161

siswa). Sedangkan siswa yang naik dan melanjutka di kelas 2. 4 kelas (139 siswa) dan dikelas 3 sebanyak 127 siswa.

Pada tahun ini dirintis pemebebasan tanah untuk lokasi SMA Negeri 1 Ciwidey Di Jalan Babakan tiga dengan status hak milik (± 8000 m2) jumlah guru dan tata usaha sebanyak 37 orang.

Tahun Pelajaran 1993-1994 SMA Negeri 1 Ciwidey diresmikan sebagai SMAN 1 Ciwidey pada tanggal 23 Desember 1993 dan mandiri dengan Kepala Sekolah Bapak Amaj Rachman sebagai pejabat yang mangku tugas (PYMT), tahun ini menerima siswa baru kelas 1 sebanyak 5 kelas (215 siswa) yang naik dan melanjutkan di kelas 2 sebanyak 5 kelas (186 siswa) dan dikelas 3 sebanyak 3 kelas (109 siswa). Jumlah guru dan tata usaha 40 orang. Tahun ini dibangun kelas baru bertingkat sebanyak 4 lokal.

Bapak Amaj Rachman pensiun dan diserah terimakan jabatan kepala sekolah kepada Bapak. Drs.Odji Setiajdi AR dan didampingi oleh Ibu Dra.Hj.Jajah Rumilah sebagai PLH. Tahun Pelajaran 1994-1995 dilakukan serah terima jabatan Kepala Sekolah dari Bpk. Drs.Odji Setiadji AR kepada Ibu Dra.Hj. Jajah Rumilah yang selama ini menjabat PLH. Setelah Dra.Hj. Jajah Rumilah Kepala Sekolah berikutnya adalah Drs. Asep Ikhsan kemudian digantikan oleh Hj. Nani P Samja,S.Pd dan setelah itu Ibu Hj.Nani digantikan oleh Drs. Suhendan Suhala.

Pada pertengahan tahun pelajaran 2006-2007 Drs. Suhendan Suhala digantikan oleh H. Totong Syamsudin,SP.d.M.MPd. Tahun Pelajaran baru 2007-

2008 SMA Negeri 1 Ciwidey Menerima siswa baru 340 siswa (8 kelas). Jumlah guru dan TU tahun ini sebanyak 64 orang.

3.1.2 Visi dan Misi

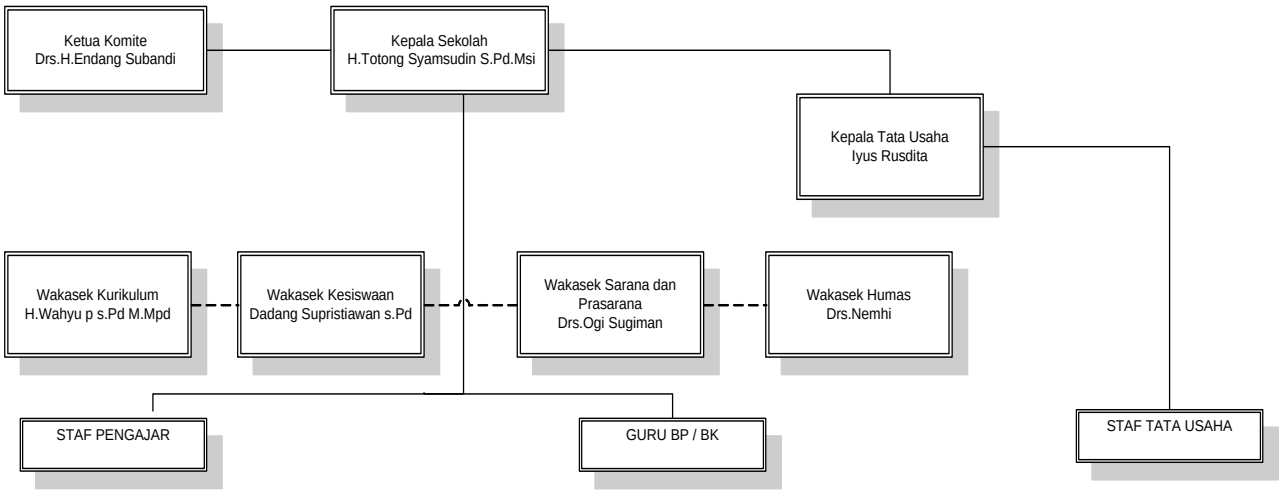
V i s i

Terwujudnya peserta didik yang beriman, bertaqwa, berprestasi, berbudaya, dan berwawasan lingkungan.

M i s i

1. Membentuk pesetra didik yang memiliki ketakwaan terhadap Tuhan Yang Maha Esa
2. Mengembangkan sikap dan kepribadian yang santun, beretika dan berestetika tinggi
3. Membentuk peserta didik yang senantiasa menjunjung tinggi kebudayaan nasional dan daerah
4. Mewujudkan kegiatan pembelajaran yang inovatif , kreatif, dan menyenangkan
5. Mempersiapkan peserta didik untuk melanjutkan ke jenjang perguruan tinggi
6. Membentuk sikap dan watak kompeteitif peserta didik sehingga senantiasa siap berkompetisi baik di tingkat lokal, nasional hingga global
7. Membangun kemitraan yang biak dengan segenap stoke holder lingkungan sekolah.

3.1.3 Struktur Organisasi



Gambar 3.1 Struktur Organisasi SMAN 1 CIWIDEY

3.1.4 Deskripsi Tugas

1. **Komite Sekolah**

   Bertugas memantau keadaan sekolah agar sesuai dengan visi dan misi yang akan dijalankan sekolah

   Susunan Pengurus Komite Sekolah SMA Negeri 1 Ciwidey
   Periode 2006 - 2009

   Ketua : Drs. H.Endang Subandi

2. **Kepala Sekolah** : H.Totong syamsudin S.pd .Msi

   Mengelola seluruh unit usaha sesuai visi dan misi, memimpin dan mengolah aset SMA Negeri1 Ciwidey serta senantiasa berusaha meningkatkan efisiensi dan poduktivitas.

3. **Wakasek Bidang Kurikulum : H.Wahyu P .S.pd,MPAPD**

   Bagian ini merencanakan melakukan atau menentukan kurikulum yang digunakan pada saat sekarang dan yang akan datang

4. **Wakasek Kesiswaan : Dadang Supristiawan Spd**

Pada bagian ini berfungsi untuk mengatur semua kegiatan siswa

5. **Wakasek Prasarana : Drs.Ogi Sugiman**
   Bagian ini bertugas menyediakan sarana dan prasarana untuk segala kebutuhan di SMA Negeri 1 Ciwidey.
6. **Wakasek Humas : Drs.Nemhi**
   Pada bagian humas yaitu melayani apabila ada tamu dari luar yang berkunjung ke SMA Negeri 1 Ciwidey.
7. **Urusan Tata Usaha : Iyus Rusdita**
   Bagian ini bertugas melakukan penerimaan laporan data siswa yang masuk dan melakukan pencatatan laporan pembayaran.
8. **Koordinator BP / BK : Drs. Taryat**
   Bagian ini bertugas untuk menyalurkan minat dan bakat siswa.

3.2 Metode Penelitian

3.2.1 Desain penelitian

Suatu penelitian tidak akan berjalan dengan baik apabila tidak dilakukan dalam suatu proses yang teratur dan terarah. Oleh karena itu diperlukan suatu metode pendekatan/penyelesaian untuk melaksanakan suatu penelitian. Metode pendekatan/penyelesaian yang digunakan oleh penulis pada penelitian ini adalah metode *deskriptif*, yaitu akumulasi data dasar dalam cara *deskriptif* semata-mata tidak perlu mencari atau menerangkan saling hubungan, membuat ramalan, dan mendapatkan makna. Para ahli dalam bidang penelitian tidak ada kesepakatan mengenai apa sebenarnya penelitian *deskriptif* itu, tetapi ada sebagian para ahli mendefinisikan penelitian *deskriptif* dalam arti yang lebih luas, dan penelitian *deskriptif* disebut juga penelitian *survei*.

Langkah-langkah pokok metode *deskriptif,* mendefinisikan dengan dan spesifik tujuan yang akan dicapai, merencanakan bagaimana cara pendekatannya, mengumpulkan semua hasil yang telah terdata, dan terakhir adalah menyusun laporan.

3.2.2 Jenis dan Metoda Pengumpulan Data

Untuk mencapai maksud dan tujuan pada penelitian ini penulis menggunakan beberapa metode pengumpulan data, yaitu mengumpulkan data-data yang berkaitan dengan pembuatan *Program*.

3.2.2.1 Sumber data Primer

Sumber data primer yang dilakukan oleh penulis yaitu dengan cara wawancara langsung dengan bagian staff pengembang IT di SMAN 1 CIWIDEY Data yang diperoleh berupa kesepakatan rancangan/hasil akhir yang akan dicapai dalam pembangunan sistem pembelajaran berbasis web ini, sehingga tidak melenceng dari apa yang inginkan pihak staff pengembang IT di SMAN 1 CIWIDEY. Teknik pengumpulan data yang digunakan oleh penulis adalah :

1. Studi Observasi

   Yaitu peninjauan secara langsung ke objek penelitian dalam hal ini adalah SMAN 1 CIWIDEY untuk kemudian dianalisis dan digunakan untuk pembentukan deskripsi masalah.

2. Interview (Wawancara)

   Yaitu dengan melakukan wawancara langsung dengan staff pengembang teknologi informasi di SMAN 1 CIWIDEY

3. Dokumentasi

   Yaitu mengumpulkan, membaca dan mempelajari laporan-laporan dengan tujuan untuk mencari pemecahan masalah, konsep, dan generalisasi yang dapat digunakan sebagai landasan teori / kerangka dalam proses pembuatan laporan ini.

3.2.2.2 Sumber data Sekunder

Data sekunder merupakan data yang sudah ada, data tersebut sudah dikumpulkan sebelumnya untuk tujuan-tujuan yang tidak mendesak. Keuntungan data sekunder ialah sudah tersedia, ekonomis, dan cepat didapat. Kelemahan data sekunder ialah tidak dapat menjawab secara keseluruhan masalah yang sedang diteliti. Kelemahan lainnya ialah kurangnya akurasi karena data sekunder dikumpulkan oleh orang lain untuk tujuan tertentu dengan menggunakan metode yang tidak kita ketahui, sehingga memugkinkan terjadinya perbedaan unit pengukuran dan umur data.

Sumber data sekunder yang didapatkan oleh penulis berupa data-data sekolah, data-data guru beserta staff nya, dan data-data siswa, yang sudah berbentuk data digital (sudah terkomputerisasi).

3.2.3 Metode Pendekatan dan Pengembangan Sistem

3.2.3.1 Metode Pendekatan Sistem

Metode ini menekankan pada karakteristik dari data yang akan diproses, dimana metodologi ini diklasifikasikan kedalam dua kelompok, yaitu :

a. Data-flow Oriented Methodologies

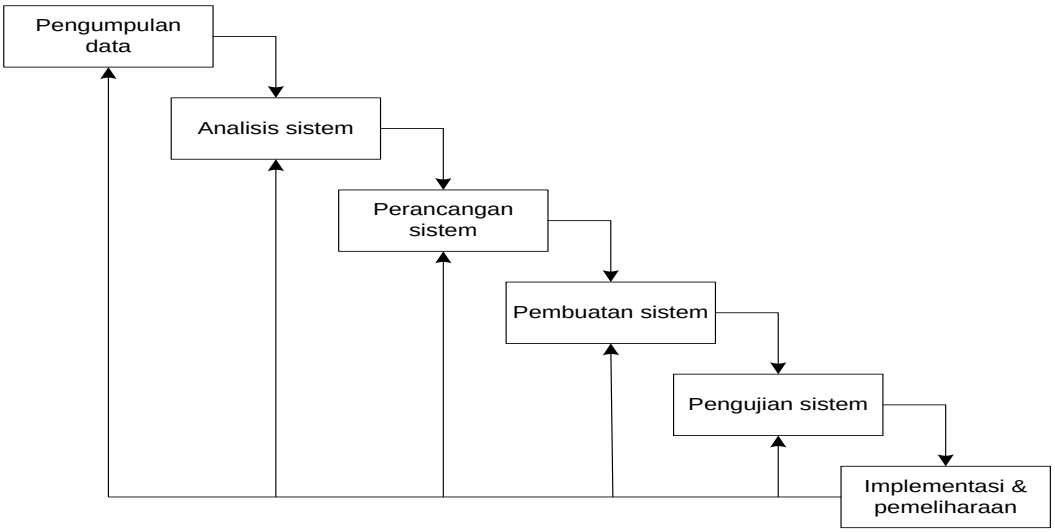
Metoda ini menekankan pada pemecahan dari sistem kedalam modul-modul berdasarkan tipe elemen data dan tingkah laku logik dari elemen data tersebut kedalam sistem. Dengan metodologi ini, sistem dapat digambarkan dari arus data dan berhubungan antar fungsi di dalam modul-modul di sistem.

b. Data Structure Oriented Methodologies

Metodologi ini menekankan struktur dari input dan output di sistem. Struktur ini kemudian akan digunakan sebagai dasar struktur dari sistemnya. Hubungan fungsi antar modul atau elemen-elemen sistem kemudian dijelaskan dari struktur sistemnya.

3.2.3.2 Metode Pengembangan Sistem

Metode Penelitian yang digunakan dalam perancangan Sistem Pembelajaran ini menggunakan ***Model Waterfall***, yang merupakan paradigma daur hidup dari sebuah permintaan yang sistematis, pengembangan perangkat lunak yang dimulai pada tingkat permasalahan sistem sampai ke tingkat analisis, perencanaan, pengadaan, pengujian dan pemeliharaan, inilah alasan mengapa penulis menggunakan Model Waterfall. Selain itu, karena model ini menawarkan cara pembuatan perangkat lunak secara lebih nyata dengan langkah-langkah sebagai berikut :



Gambar 3.2 *Waterfall* Model

Keterangan :

1. *Pengumpulan data*

   Pada tahap ini, penulis mencari dan mengumpulkan data–data dan informasi secara lengkap dan akurat.

2. *Analisis Sistem*

   Yaitu untuk megetahui kendala-kendala dan aspek-aspek sistem yang dibutuhkan.

3. *Perancangan Sistem*

   Yaitu merancang system sesuai dengan teknik dan metode perancangan system yang digunakan.

4. *Pembuatan Sistem*

   Pada tahap ini, penulis membuat suatu aplikasi web berdasarkan rancangan sistem.

5. *Pengujian Sistem*

Pada tahap ini, penulis melakukan pengujian sistem yang telah dibuat, secara black box ataupun white box.

6. *Implementasi dan Pemeliharaan*

yaitu perangkat lunak yang telah dibuat dapat mengalami perubahan sesuai permintaan pemakai. Pemeliharaan dapat dilakukan jika ada permintaan tambahan fungsi sesuai dengan keinginan admin ataupun adanya pertumbuhan dan perkembangan baik perangkat lunak maupun perangkat keras.

3.2.3.3 Alat Bantu Analisis dan Perancangan

1. Pemrograman Berorientasi Objek

Merupakan paradigma baru dalam rekayasa software yang didasarkan obyek dan kelas. Teknik object oriented memandang software bagian per bagian dan menggambarkannya dalam satu obyek. Obyek dapat beraksi dan bereaksi. Teknologi obyek menganalogikan sistem aplikasi seperti kehidupan nyata yang didominasi obyek contoh :

- Manusia adalah obyek yang memiliki atribut(nama, pekerjaan, rumah, dll.)
- Manusia dapat berjalan, bicara, makan, minum, dll.

Empat prinsip dasar dari pemograman berorientasi obyek yaitu :

1. **Abstraksi** : memfokuskan pada karakteristik obyek
2. **Enkapsulasi** : menyembunyikan banyak hal yang terdapat dalam obyek yang tidak perlu diketahui obyek lain
3. **Modularitas** : membagi sistem yang rumit menjadi bagian-bagian yang lebih kecil

4. **Hirarki** : berhubungan dengan abstraksi dan modularitas yaitu pembagian berdasarkan urutan dan pengelompokan tertentu.

2. Objek Management Group (OMG)

Objek Management Group (OMG) adalah sebuah nama konsorium UML yang dimapankan oleh **Ivar Jacobson** yang aggota-anggotanya adalah beberapa organisasi berdedikasi tinggi yang bertujuan melengkapi/memperluas konsep-konsep yang melandasi UML**.** Kolaborasi ini menghasilkan UML 1.0 yang dapat digunakan secara luas dalam hampir segala bentuk perancangan perangkat lunak dan sistem informasi.

Demikian perkembangn berlanjut terus degan masuknya organisasi-organisasi besar kedalam OMG (*Object Management Group)* yang merupakan konsorium yang berusaha membuat standarisasi metoda pengembangan perangkat lunak berorientasi objek tadi.

3. Semantik dalam UML

OMG telah menetapkan semantik (makna istilah) semua notasi UML dalam model structural dan model behavioral. Model structural (disebut juga sebagai model statis) menekankan struktur objek dalam sebuah sistem, menyangkut kelas-kelas, interface, attribute, dan hubungan antarkomponen. Model behavioral (juga disebut sebagai model dinamis) menekankan perilaku objek dalam sebuah sistem, termasuk metode, interaksi, kolaborasi, dan state history. Lebih lengkapnya tentang semantik dalam UML bisa dilihat di http://www.rational.com/uml.

4. Faktor yang Mendorong dibuatnya UML

Pendekatan analisa dan perancangan dengan menggunakan model *Object-Oriented* mulai diperkenalkan sekitar pertengahan tahun 1970 sampai akhir 1980 dikarenakan pada saat itu aplikasi software sudah sangat meningkat dan mulai kompleks. Jumlah yang menggunakan *OO* mulai diujicobakan dan diaplikasikan antara 1989 hingga 1994, seperti halnya oleh Grady Brooch dari *Rational Software Co.*, dikenal dengan OOSE (*Object-Oriented Software Engineering*), serta James Rumbaugh dari *General Electric*, dikenal dengan OMT (*Object Modeling Technique*).

Faktor yang mendorong dibuatnya UML adalah mulai disadarinya kelemahan-kelemahan oleh Booch maupun Rumbaugh mengenai tidak adanya standar penggunaan model yang berbasis *OO,* dan ketika mereka bertemu ditemani rekan lainya yaitu Ivar Jacobson dari *Objectory* mulai mendiskusikan untuk mengadopsi masing-masing pendekatan metode *OO* untuk membuat suatu model bahasa yang *uniform/seragam* yang disebut UML (*Unified Modeling Language)* dan dapat digunakan oleh seluruh dunia.

5. Tujuan UML

Tujuan dari UML, sebagaimana yang dinyatakan oleh perancangnya adalah:

1. Memodelkan sistem (dan bukan hanya perangkat lunak) menggunakan konsep object-oriented.
2. Membuat sebuah bahasa pemodelan yang dapat digunakan baik oleh manusia maupun oleh mesin.

3. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemrograman dan proses rekayasa.
4. Menyatukan praktek-praktek terbaik yang terdapat dalam pemodelan.

6. Cakupan UML

Unified Modeling Language (UML) merupakan sistem arsitektur yang bekerja dalam OOAD (*Object-Oriented Analysis/Design*) dengan satu bahasa yang konsisten untuk menentukan, visualisasi, mengkontruksi, dan mendokumentasikan *artifact* (sepotong informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa software, dapat berupa model, deskripsi, atau software) yang terdapat dalam sistem software.

UML menggabungkan konsep Booch, OMT (*Object Modeling Technique*), dan OOSE (*Object-Oriented Software Engineering*). UML menekankan pada apa yang dapat dikerjakan dengan metode-metode tsb. UML berfokus pada bahasa Pemodelan Standar, bahkan pada Proses Standar.

UML merupakan kesatuan dari dari ketiga pemodelan tersebut dan ditambah kemampuan lebih karena mengandung metode tambahan untuk mengatasi masalah pemodelan yang tidak dapat ditangani ketiga metode tersebut. UML dikeluarkan oleh OMG (Object Management Group, Inc) yaitu organisasi internasional yang dibentuk pada 1989, terdiri dari perusahaan sistem informasi, *software developer*, dan para *user* sistem komputer.

7. Perbedaan UML dengan Bahasa Pemodelan Lain

Ada tiga aspek utama dalam pemodelan sistem yang mampu didukung oleh UML :

1. Functional Model, untuk menunjukkan fungsionalitas dari suatu sistem dari sudut pandang user atau pengguna. Ini dicapai dengan menggunakan Use Case Diagram.
2. Object Model, untuk menunjukkan struktur dan substruktur dari suatu sistem dengan menggunakan object, atribut, operasi dan juga asosiasi. Ini dicapai dengan menggunakan Class Diagram.
3. Dynamic Model, menunjukkan internal behavior dan suatu sistem. Ini dicapai dengan menggunakan Sequence Diagram, Activity Diagram dan juga Statechart Diagram.

8. Alat Bantu UML

Alat bantu UML didefinisikan oleh beberapa diagram, diantaranya sebagai berikut :

1. **Diagram Use Case**

Diagram Use Case menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar yang menjadi persoalan itu *apa yang dilakukan* bukan *bagaimana melakukannya*.

Diagram Use Case dekat kaitannya dengan kejadian-kejadian. kejadian (scenario) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem. Untuk lebih memperjelas lihat gambaran suatu peristiwa untuk sebuah klinik kesehatan di bawah ini :

> "Pasien menghubungi klinik untuk membuat janji (appointment) dalam pemeriksaan tahunan. Receptionist mendapatkan waktu yang luang pada buku jadwal dan memasukkan janji tersebut ke dalam waktu luang itu."



Gambar 3. 3 contoh *Use case* kegiatan pasien yang membuat janji.

Diagram Use Case berguna dalam tiga hal :

☒ Menjelaskan fasilitas yang ada (requirements)

Use Case baru selalu menghasilkan fasilitas baru ketika sistem di analisa, dan design menjadi lebih jelas.

☒ Komunikas dengan klien

Penggunaan notasi dan simbol dalam diagram Use Case membuat pengembang lebih mudah berkomunikasi dengan klien-kliennya.

☒ Membuat test dari kasus-kasus secara umum

Kumpulan dari kejadian-kejadian untuk Use Case bisa dilakukan test kasus layak untuk kejadian-kejadian tersebut.

2. **Diagram Class**

Diagram Class memberikan pandangan secara luas dari suatu sistem dengan menunjukan kelas-kelasnya dan hubungan mereka. Diagram Class bersifat statis; *menggambarkan hubungan apa yang terjadi bukan apa yang terjadi jika mereka berhubungan*.

Diagram Class mempunyai 3 macam relationalships (hubungan), sebagai berikut :

☒ Association

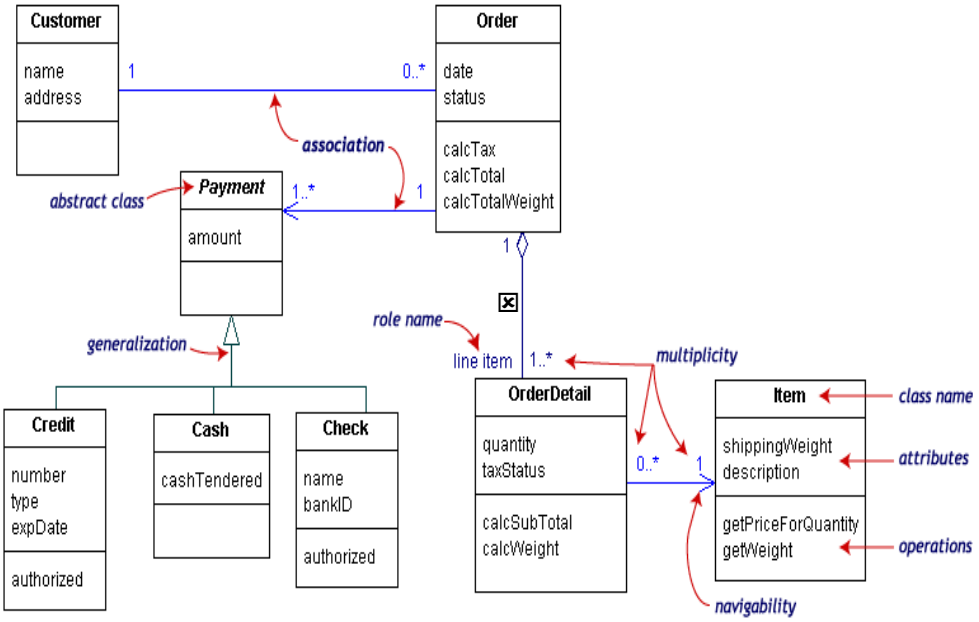
Suatu hubungan antara bagian dari dua kelas. Terjadi *association* antara dua kelas jika salah satu bagian dari kelas mengetahui yang lainnya dalam melakukan suatu kegiatan. Di dalam diagram, sebuah *association* adalah penghubung yang menghubungkan dua kelas.

☒ Aggregation

Suatu *association* dimana salah satu kelasnya merupakan bagian dari suatu kumpulan. *Aggregation* memiliki titik pusat yang mencakup keseluruhan bagian. Sebagai contoh : OrderDetail merupakan kumpulan dari Order.

☒ Generalization

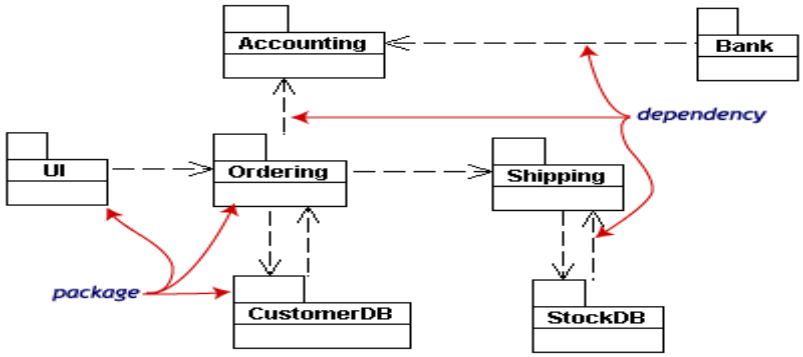
Suatu hubungan turunan dengan mengasumsikan satu kelas merupakan suatu *superClass* (kelas super) dari kelas yang lain. *Generalization* memiliki tingkatan yang berpusat pada *superClass*. Contoh : Payment adalah *superClass* dari Cash, Check, dan Credit.



Gambar 3. 4 Contoh *Diagram Class* transaksi Pembelian barang.

3. **Package dan Object**

Untuk mengatur pengorganisasian diagram Class yang *kompleks*, dapat dilakukan pengelompokan kelas-kelas berupa *packag*e (paket-paket). *Package* adalah kumpulan elemen-elemen logika UML. Gambar di bawah ini mengenai model bisnis dengan pengelompokan kelas-kelas dalam bentuk paket-paket :



Gambar 3. 5 Contoh *Diagram Package*.

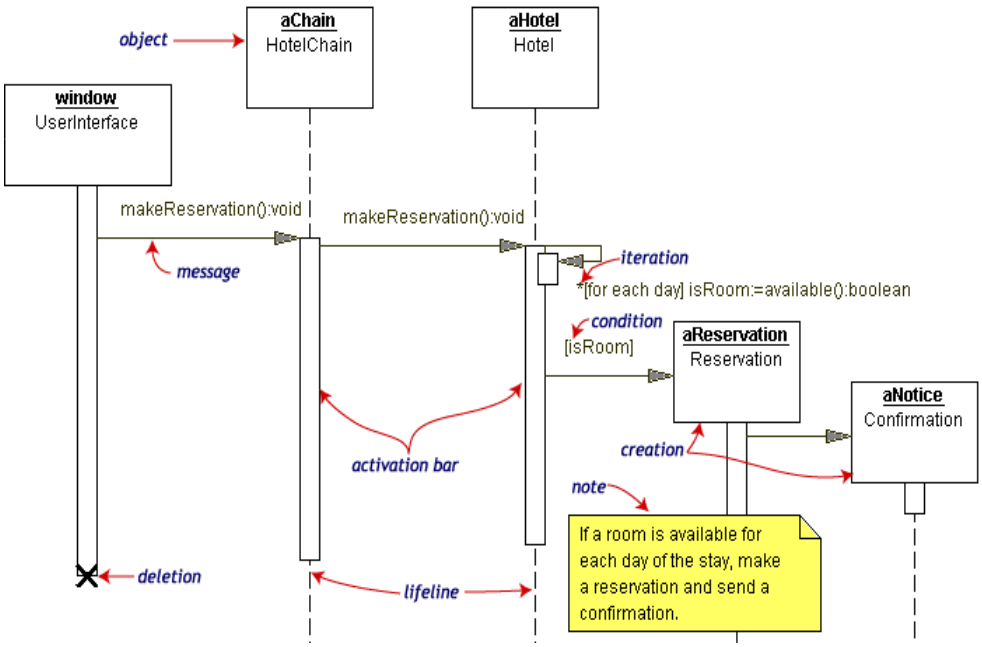
Ada jenis khusus dari diagram Class yaitu diagram Object. Kegunaannya untuk penjelasan yang sedikit dengan relasi yang sulit, khususnya relasi rekursif.

4. **Diagram Sequence**

Diagram Class dan diagram Object merupakan suatu gambaran *model statis*. Namun ada juga yang bersifat *dinamis*, seperti **Diagram Interaction**.

Diagram sequence merupakan salah satu diagram Interaction yang menjelaskan bagaimana suatu operasi itu dilakukan; *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Obyek-obyek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.

Di bawah ini adalah diagram Sequence untuk pembuatan Hotel Reservation. Obyek yang mengawali urutan *message* adalah ‘aReservation Window’.



Gambar 3. 6 Contoh *Diagram Sequence* ‘Pemesanan kamar di Hotel’.

‘Reservation window’ mengirim pesan `makeReservation()` ke ‘HotelChain’. Kemudian ‘HotelChain’ mengirim pesan yang sama ke ‘Hotel’. Bila ‘Hotel’ punya kamar kosong, maka dibuat ‘Reservation’ dan ‘Confirmation’.

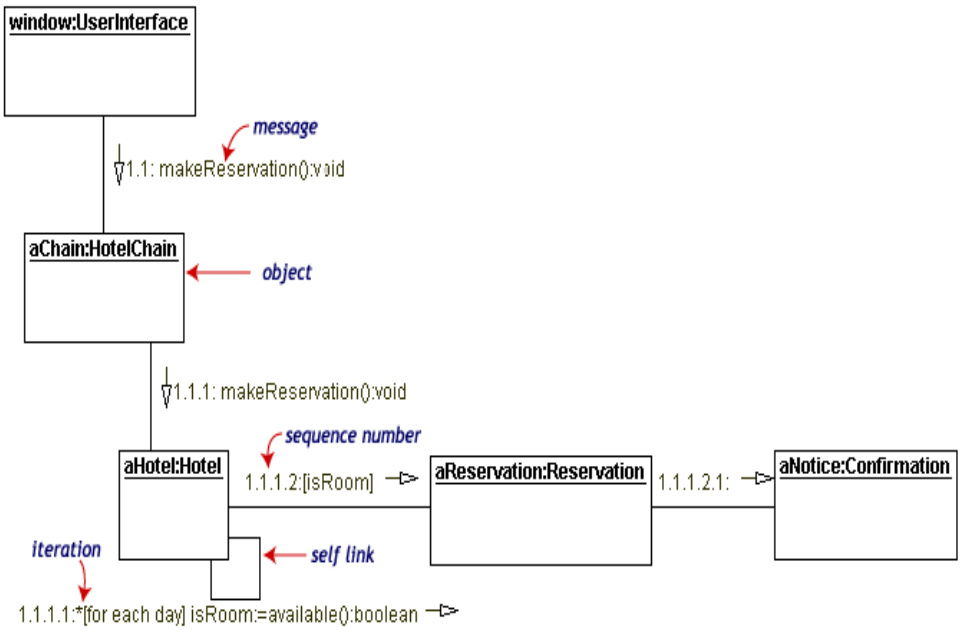
Lifeline adalah garis dot (putus-putus) vertikal pada gambar, menerangkan waktu terjadinya suatu obyek. Setiap panah yang ada adalah pemanggilan suatu pesan. Panah berasal dari pengirim ke bagian paling atas dari batang kegiatan (*activation bar*) dari suatu pesan pada *lifeline* penerima. *Activation bar* menerangkan lamanya suatu pesan diproses.

Pada gambar diagram , terlihat bahwa ‘Hotel’ telah melakukan pemanggilan diri sendiri untuk pemeriksaan jika ada kamar kosong. Bila benar, maka ‘Hotel’ membuat ‘Reservation’ dan ‘Confirmation’. Pemanggilan diri sendiri disebut dengan *iterasi*. *Expression* yeng dikurung dengan “[ ]”, adalah *condition* (keadaan kondisi).

Pada diagram dapat dibuat *note* (catatan). Pada gambar, terlihat seperti selembar kertas yang berisikan teks. *Note* bisa diletakan dimana saja pada diagram UML.

5. **Diagram Collaboration**

Diagram Collaboration juga merupakan *diagram interaction*. Diagram membawa informasi yang sama dengan diagram Sequence, tetapi lebih memusatkan atau memfokuskan pada kegiatan obyek dari waktu pesan itu dikirimkan.



Gambar 3. 7 Contoh *Diagram Collaboration* 'Pemesanan kamar di Hotel'

Kotak kegiatan obyek diberi label dengan nama kelas atau obyek (atau keduanya). Nama kelas dibatasi dengan *colons /titik dua* (:).

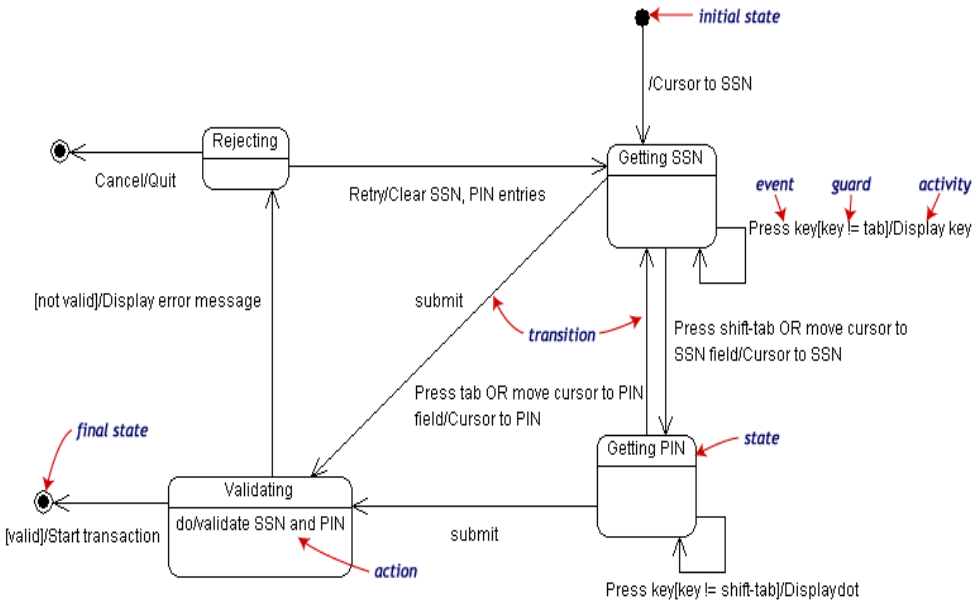
Setiap pesan pada diagram Collaboration mempunyai angka yang terurut. Pesan yang tingkatannya tertinggi adalah angka 1. Pesan yang berada pada tingkat yang sama memiliki *prefix* yang sama, namun *suffix* berbeda bergantung pada posisinya; hanya untuk angka 1, 2, dan seterusnya.

6. **Diagram StateChart**

Behaviors dan *state* dimiliki oleh obyek. Keadaan dari suatu obyek bergantung pada kegiatan dan keadaan yang berlaku pada saat itu. Diagram StateChart menunjukan kemungkinan dari keadaan obyek dan proses yang menyebabkan perubahan pada keadaannya.

Untuk lebih jelas, contoh yang digunakan model diagram untuk login yang merupakan bagian dari Online Banking System. Logging in terdiri atas masukan

input Social Security Number dan Personal Id Number yang berlaku, lalu memutuskan kesahan dari informasi tersebut.



Gambar 3.8 Contoh *Diagram StateChart* 'Sistem Perbankkan secara Online'.

Logging in dapat dibagi menjadi empat tahapan proses, yaitu :

- ☒ Getting SSN (masukkan SSN),
- ☒ Getting PIN (masukkan PIN),
- ☒ Validating (periksa kesahannya), dan
- ☒ Rejecting (keluar).

Proses peralihan digambarkan dengan panah dari satu state ke yang lainnya. *Event* (peristiwa) atau *condition* (keadaan) yang menyebabkan perubahan dituliskan pada samping panah. Diagram ini mengandung dua *self-transition* (transisi sendiri), satu pada getting SSN dan lainnya pada getting PIN.

Keadaan awal *Start* (*black circle* /lingkar hitam) adalah *dummy* (model) untuk memulai action (kegiatan). Keadaan akhir juga keadaan model yang menghentikan kegiatan.

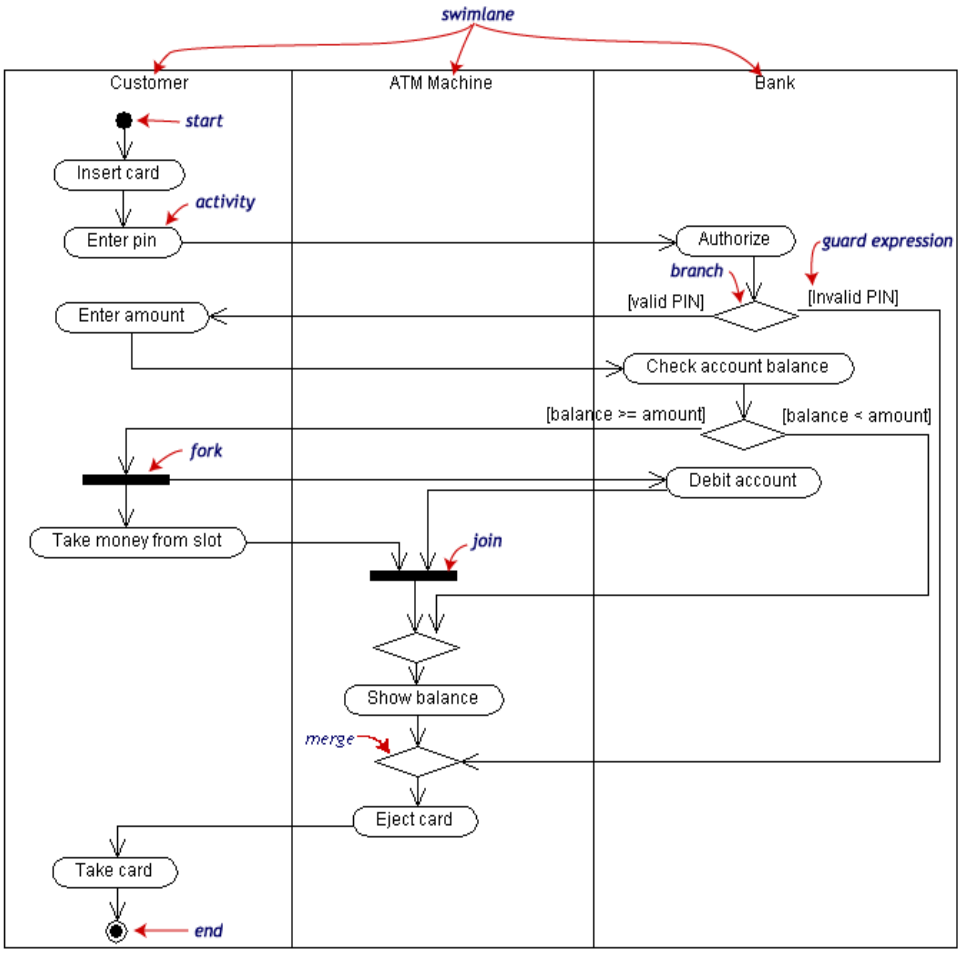
Aksi yang terjadi sebagai hasil dari suatu peristiwa atau keadaan ditandai dalam bentuk `/action`. Pada Validating State, obyek tidak menunggu peristiwa dari luar untuk menyebabkan suatu perubahan. Sebagai gantinya melakukan suatu *activity* (aktifitas). Hasil dari aktifitas tersebut menentukan keadaan berikutnya dari obyek tersebut.

7. **Diagram Activity**

Pada dasarnya diagram Activity sering digunakan oleh *flowchart*. Diagram ini berhubungan dengan diagram Statechart. Diagram Statechart berfokus pada *obyek yang dalam suatu proses* (atau proses menjadi suatu obyek), diagram Activity berfokus pada *aktifitas-aktifitas yang terjadi yang terkait dalam suatu proses tunggal*. Jadi dengan kata lain, diagram ini menunjukkan bagaimana aktifitas-aktifitas tersebut bergantung satu sama lain.

Sebagai contoh, perhatikan proses yang terjadi. “Pengambilan uang dari bank melalui ATM.”

Ada tiga aktifitas kelas (orang, dan lainnya) yang terkait yaitu : Customer, ATM, and Bank. Proses berawal dari lingkaran start hitam pada bagian atas dan berakhir di pusat lingkaran stop hitam/putih pada bagian bawah. Aktivitas digambarkan dalam bentuk kotak persegi. Lihat gambar di bawah ini, agar lebih jelas :



Gambar 3. 9 Contoh *Diagram Activity* 'Pengambilan Uang melalui ATM'.

Diagram Activity dapat dibagi menjadi beberapa jalur kelompok yang menunjukkan obyek yang mana yang bertanggung jawab untuk suatu aktifitas. Peralihan tunggal (*single transition*) timbul dari setiap adanya *activity* (aktifitas), yang saling menghubungi pada aktifitas berikutnya.

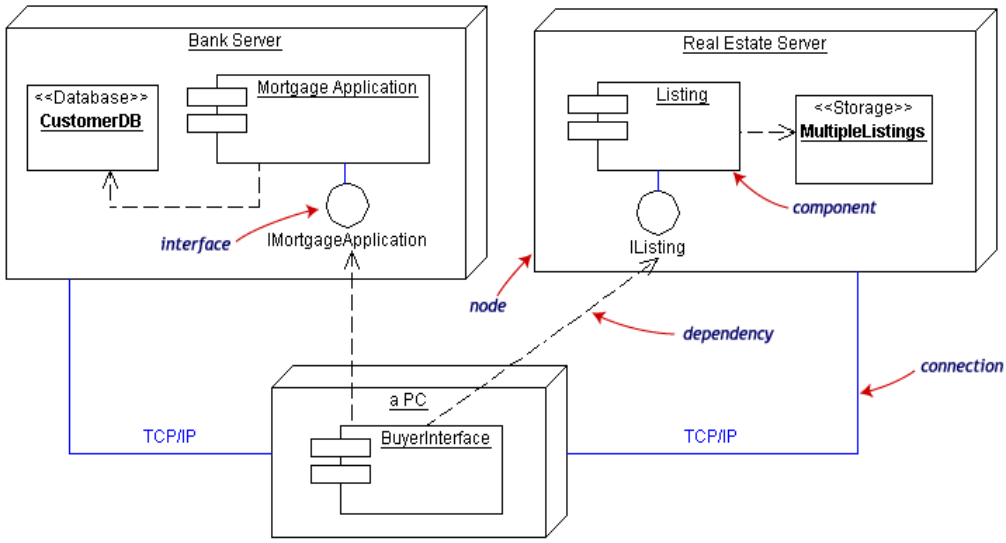
Sebuah *transition* (transisi) dapat membuat cabang ke dua atau lebih percabangan *exclusive transition* (transisi eksklusif). Label *Guard Expression* (ada di dalam [ ]) yang menerangkan output (keluaran) dari percabangan. percabangan akan menghasilkan bentuk menyerupai bentuk intan. *transition* bisa bercabang

menjadi beberapa aktifitas paralel yang disebut **Fork.** *Fork* beserta *join* (gabungan dari hasil output *fork*) dalam diagram berbentuk *solid bar* (batang penuh).

8. **Diagram Component dan Deployment**

Component adalah sebuah *code module* (kode-kode module). Diagram Component merupakan *fisik sebenarnya dari diagram Class*. Diagram Deployment menerangkan bahwa *konfigurasi fisik software* dan *hardware*.

Gambar 3.10 menerangkan hubungan sekitar komponen software dan hardware yang berperan dalam ruang lingkup real estate.



Gambar 3. 10 Contoh *Diagram Deployment* 'Sistem Real Estate'.

Fisik hardware berbentuk seperti *node-node*. Setiap komponen merupakan bagian dari *node*. Pada gambar komponen berbentuk dua kotak tersusun yang terletak di sebelah kiri atas.

3.2.4 Faktor Pengujian sofware

Metode pengujian adalah cara atau teknik untuk menguji perangkat lunak, mempunyai mekanisme untuk menentukan data uji yang dapat menguji perangkat lunak secara lengkap dan mempunyai kemungkinan tinggi untuk menemukan kesalahan. Perangkat lunak dapat diuji dengan dua cara, yaitu :

1. Pengujian dengan menggunakan data uji untuk menguji semua elemen program (data internal, loop, logika, keputusan dan jalur). Data uji dibangkitkan dengan mengetahui struktur internal (kode sumber) dari perangkat lunak.
2. Pengujian dilakukan dengan mengeksekusi data uji dan mengecek apakah fungsional perangkat lunak bekerja dengan baik. Data uji dibangkitkan dari spesifikasi perangkat lunak.

Pengujian *black box* adalah pengujian aspek fundamental sistem tanpa memperhatikan struktur logika internal perangkat lunak. Metode ini digunakan untuk mengetahui apakah perangkat lunak berfungsi dengan benar. Pengujian black box merupakan metode peracangan data uji yang didasarkan pada spesifikasi perangkat lunak. Data uji dibangkitkan, dieksekusi pada perangkat lunak dan kemudian keluaran dari perangkat lunak dicek apakah telah sesuai dengan yang diharapkan

Pengujian *black box* berusaha menemukan kesalahan dalam kategori :

1) fungsi-fungsi yang tidak benar atau hilang

2) kesalahan interface

3) kesalahan dalam struktur data atau akses database eksternal

4) kesalahan kinerja

5) inisialisasi dan kesalahan terminasi.