BAB III

OBJEK DAN METODE PENELITIAN

3.1. Objek Penelitian

Objek yang diteliti ini adalah pembelajaran mata pelajaran yang ada sangkut pautnya dengan keagamaan dengan menggunakan fasilitas internet atau dikenal dengan pembelajaran elektronik (*e-learning*)

Dalam artian objek penelitian merupakan variable yang akan dikaji, yang menjadi focus penelitian disertai keberadaan/tempat dilakukannya objek penelitian tersebut".

Objek penelitian penting diungkapkan karena berperan dalam menunjang keberhasilan kegiatan selama penelitian, sehingga hal-hal yang diperlukan dalam penelitian akan mudah dicapai. Dengan demikian yang menjadi objek penelitian ini adalah "Sistem Informasi Pembelajaran Jarak Jauh Prodi Sistem Informasi Universitas Komputer Indonesia**".** Selanjutnya untuk melengkapi objek penelitian ini, akan diuraikan tentang sejarah singkat, visi dan misi, struktur organisasi dan deskripsi tugasnya di bawah ini.

3.1.1. Sejarah Singkat

Universitas Komputer Indonesia (UNIKOM) secara resmi berdiri pada hari Selasa, tanggal 8 Agustus 2000 berdasarkan Surat Keputusan Menteri Pendidikan Nasional nomor 126/D/0/2000.

Awalnya dimulai pada bulan Juli tahun 1994 ketika didirikan Lembaga Pendidikan Komputer Indonesia Jerman, disingkat LPKIG, bertempat di jalan Dipati Ukur 102 Bandung . Dengan 1 ruang kelas berkapasitas 50 orang dan 1 laboratorium komputer dengan 25 unit komputer, Lembaga ini membuka program pendidikan 1 tahun dengan 5 program studi yaitu Ahli Komputer Aplikasi Bisnis, Ahli Komputer Keuangan & Perbankan, Ahli Komputer Akuntansi & Perpajakan, Ahli Komputer Manajemen & Pemasaran dan Sekretaris Eksekutif. Jumlah peserta pendidikan pada tahun pertama ini sebanyak 233 siswa.

Pada tahun kedua, 1995, dibuka jenjang pendidikan 3 tahun untuk memenuhi animo siswa tahun pertama yang ingin memperdalam ilmunya, disamping pemikiran jangka panjang pengembangan institusi. Pada tahun ini juga dibuka program studi baru, meliputi : Ahli Komputer Teknik Informatika, Ahli Komputer Manajemen Informatika dan Sekretaris Eksekutif. Ruang kelas ditambah menjadi 2 buah dan laboratorium komputer menjadi 2 buah dengan jumlah siswa sebanyak 457 orang.

Pada tahun ketiga, 1996, dilakukan penambahan gedung kuliah baru bertempat di jalan Dipati Ukur 116 (gedung FISIP sekarang), sekaligus pemindahan pusat administrasi dan perkantoran. Digedung baru ini dilakukan penambahan 1(satu) Lab. Komputer, 5(lima) Ruang Kuliah, Ruang Dosen dan

Ruang Kemahasiswaan. Jumlah siswa dari tahun 1996 hingga tahun 1998 bertambah dari 632 orang menjadi 1184 orang.

Pada tahun kelima, 1998, dimulai pembangunan Kampus baru (Gedung Rektorat /Kampus-1 sekarang) berlantai 6(enam) di jalan Dipati Ukur 114. Pembangunan Kampus baru ini dapat diselesaikan pada bulan Agustus 1999, sehingga pada awal perkuliahan bulan September 1999 telah dapat digunakan.

Mencermati dinamika peserta didik dan pengembangan Institusi kedepan, pada tanggal 24 Desember 1998 dibentuklah Yayasan Science dan Teknologi dan dilanjutkan dengan pengajuan pendirian STIMIK IGI dan STIE IGI ke DIKTI.

Pada bulan Juli 1999 STIE IGI diresmikan dengan keluarnya SK Mendiknas no. 119/D/O/1999 dengan 5 program studi : Akuntansi S1, Manajemen S1, Manajemen Pemasaran D3, Keuangan Perbankan D3 serta Akuntansi D3.

Pada bulan Agustus 1999 STIMIK IGI diresmikan dengan keluarnya SK Mendiknas no. 143/D/O/1999 dengan 5 program studi : Teknik Informatika S1, Manajemen Informatika D3, Teknik Komputer D3, Komputerisasi Akuntansi D3 serta Teknik Informatika D3.

Agar Sistem Pendidikan lebih Efisien, Efektif, Produktif dengan Struktur Organisasi yang lebih baik, enam bulan kemudian dilakukan usulan ke DIKTI untuk melakukan Merger kedua Sekolah Tinggi diatas menjadi Universitas.

Pada hari Selasa, tgl. 8 Agustus 2000 keluarlah SK MENDIKNAS no. 126/D/O/2000 atas Universitas Komputer Indonesia yang disingkat dengan nama UNIKOM.Pada SK tersebut sekaligus diijinkan dibukanya 11 program studi baru

: Teknik Komputer S1, Manajemen Informatika S1, Teknik Industri S1, Teknik Arsitektur S1, Perencanaan Wilayah dan Kota S1, Ilmu Hukum S1, Ilmu Komunikasi S1, Ilmu Pemerintahan S1, Desain Interior D3, Desain Komunikasi Visual S1 dan Desain Komunikasi Visual D3.

Sejak berdirinya pada tahun 2000, setiap tahunnya UNIKOM menerima ± 2.000 mahasiswa baru. Terakhir pada tahun 2009 yang lalu diterima sebanyak 3.108 mahasiswa baru. Hingga tahun akademik 2009/2010 terdapat 6 Fakultas dan 23 Program Studi di UNIKOM dengan jumlah mahasiswa sebanyak 15.000 orang yang berasal dari berbagai pelosok tanah air dan dari luar negeri yang sedang menempuh pendidikan di UNIKOM.

3.1.2. Visi dan Misi Universitas Komputer Indonesia

3.1.2.1. Visi

Menjadi Universitas terdepan dibidang Teknologi Informasi & Komputer, berwawasan Global dan menjadi pusat Unggulan dibidang ilmu pengetahuan Teknologi dan seni yang mendukung pembangunan nasional serta berorientasi pada kepentingan masyarakat, bangsa dan Negara

3.1.2.2. Misi

Menyelenggarakan Pendidikan tinggi kearah masyarakat Industri maju dengan sistem pendidikan yang kondusif, tenaga pengajar berkualitas dan program-program studi berbasis pada teknologi informasi & komputer dengan mengoptimalkan sumber daya yang ada, kualitas dan manajemen mutu berdasarkan prinsip Quality Is Our Tradition.

3.2. Metode Penelitian

3.2.1. Desain Penelitian

Dalam melakukan penelitian salah satu hal yang penting ialah membuat desain penelitian. Desain penelitian bagaikan sebuah peta jalan bagi peneliti yang menuntun serta menentukan arah berlangsungnya proses penelitian secara benar dan tepat sesuai dengan tujuan yang telah ditetapkan. Tanpa desain yang benar seorang peneliti tidak akan dapat melakukan penelitian dengan baik karena yang bersangkutan tidak mempunyai pedoman arah yang jelas.

3.2.2. Jenis dan Metode Pengumpulan Data

3.2.2.1. Jenis Pengumpulan Data

Berdasarkan sumbernya, data dibagi menjadi:

a. Data Primer: Data yang diusahakan/didapat oleh peneliti dengan cara observasi langsung dan melakukan wawancara terhadap intansi yang sedang di teliti
b. Data Sekunder: Data yang didapat dari orang/instansi lain Data Sekunder cenderung siap “pakai”, artinya siap diolah dan dianalisis oleh penelitian.

3.2.2.2. Metode Pengumpulan Data

Penelitian mendalam, menyeluruh, atas objek tertentu yang biasanya relatif kecil selama kurun waktu tertentu, termasuk lingkungannya, maka informasi penelitian secara khusus dipilih berdasarkan purposive sampling (sampel bertujuan) dengan pertimbangan informan yang dipilih dipandang

mengetahui secara jelas terhadap permasalahan yang akan diteliti untuk keperluan penelitian pengumpulan data, yaitu :

Dalam penelitian ini dilakukan pengumpulan data melalui:

Observasi, yaitu pengamatan lansung terhadap jalannya kegiatan sistem data,proses pengolahan data, dan proses pencetakan laporan, yang kualitatif menghendaki suatu kajian yang rinci, Adapun bukti atau data peneliti menggunakan beberapa teknik informasi akademik yaitu meliputi proses penginput 15 hasilnya dalam bentuk informasi akademik yang akan di gunakan sebagai bahan pengambilan keputusan.

Dokumentasi, mengumpulkan data melalui dokumen-dokumen, buku referensi dan laporan-laporan, serta hasil pendataan yang dianggap mendukung serta melengkapi hasil penelitian yang dilakukan.

Wawancara, yaitu peneliti melakukan tanya jawab langsung kepada informan guna menggali informasi yang relevan dengan tujuan penelitian. Maksud mengadakan wawancara, seperti ditegaskan oleh Lincoln dan Guba (1985 : 266) dalam Lexi J. Moleong, yaitu antaranlain mengkonstruksi mengenai orang, kejadian, kegiatan, organisasi, perasaan, merekonstruksi kebulatan-kebulatan demikian sebagai yang dialami di masa lalu; memproyeksikan kebulatan-kebulatan sebagai yang telah diharapkan untuk dialami pada masa yang akan datang; memverivikasi, mengubah dan memperluas informasi yang diperoleh dari orang lain, baik manusia maupun bukan manusia (triangulasi); dan memverifikasi,

mengubah dan memperluas konstruksi yang dikembangkan oleh peneliti sebagai pengecekan anggota.

3.2.3. Metode Pendekatan dan Pengembangan Sistem

Dalam pembuatan sistem informasi, perlu digunakan suatu metodologi yang dapat digunakan sebagai pedoman bagaimana dan apa yang harus dikerjakan selama pembuatan sistem antara lain, metode pendekatan sistem dan pengembangan sistem. Dalam penelitian ini metode pendekatan sistem yang digunakan ialah metode pendekatan metode objek dan untuk mengembangkan sistem informasinya menggunakan metode pengembangan waterfall.

3.2.3.1. Metode Pendekatan Sistem

Karena banyak terjadi permasalahan-permasalahan di pendekatan klasik, maka kebutuhan akan pendekatan pengembangan sistem yang lebih baik mulai terasa dibutuhkan. Sayangnya sampai sekarang masih banyak orang yang tidak menyadari bahwa hanya dengan mengikuti tahapan di life cycle saja tidak akan membuat pengembangan sistem informasi menjadi berhasil. Oleh karena itu diperlukan suatu pendekatan pengembangan sistem yang baru yang dilengkapi dengan beberapa alat dan teknik supaya membuatnya berhasil.

Pendekatan ini yang dimulai dari awal tahun 1970 disebut dengan pendekatan terstruktur (structured approach). Pendekatan terstruktur dilengkapi dengan alat-alat (tools) dan teknik-teknik (techniques) yang dibutuhkan dalam pengembangan sistem, sehingga hasil akhir dari sistem yang dikembangkan akan didapatkan sistem yang strukturnya didefinisikan dengan baik dan jelas.

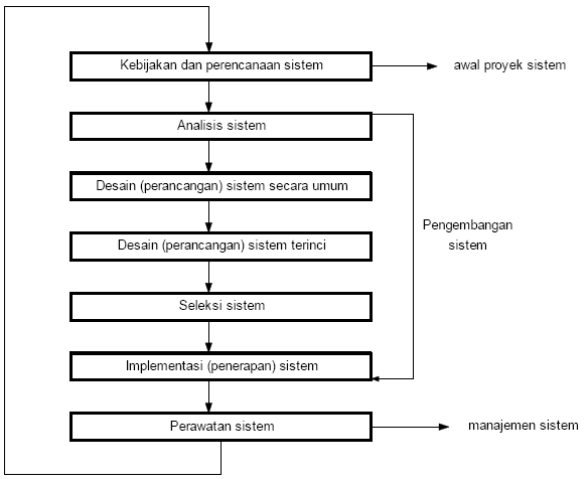
Konsep pengembangan sistem terstruktur bukan merupakan konsep yang baru. Teknik perakitan di pabrik-pabrik dan perancangan sirkuit untuk alat-alat elektronik adalah dua contoh dari konsep ini yang banyak digunakan di industri-industri. Konsep ini memang relatif masih baru digunakan dalam mengembangkan sistem informasi untuk dihasilkan produk sistem yang memuaskan pemakainya.

3.2.3.2. Pengembangan Sistem

Pengembangan Sistem dapat berarti menyusun suatu sistem yg baru untuk menggantikan sistem yg lama secara keseluruhan atau memperbaiki sistem yg telah ada.

Metodologi adalah : Kesatuan metode-metode, prosedur-prosedur, konsep-konsep pekerjaan, aturan-aturan dan postulat-postulat yang digunakan oleh suatu ilmu pengetahuan, seni atau disiplin lainnya

Metode adalah : Suatu cara/teknik yang sistematik untuk mengerjakan sesuatu.



Gambar 3.1 Konsep Pengembangan Sistem

a. Kebijakan dan Perencanaan Sistem
b. Top Management Mengkaji, mengkoordinasi pelaksanaan, memonitor, menilai kinerja, dan memberi saran terhadap kelangsungan proyek yang dikembangkan
c. Analisis Sistem Penelitian system
d. Desain Sistem Secara umum Memberikan gambaran umum system
e. Seleksi Sistem (Penilaian Sistem) Apakah rancangan akan dilanjutkan
f. Desain dan Implementasi Terinci

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 :

g. Manusia adalah obyek yang memiliki atribut(nama, pekerjaan, rumah, dll.)
h. 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. 2 contoh *Use case* kegiatan pasien yang membuat janji.

Diagram Use Case berguna dalam tiga hal :

a. Menjelaskan fasilitas yang ada (requirements)

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

b. Komunikas dengan klien

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

c. Membuat test dari kasus-kasus secara umum

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

1. **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

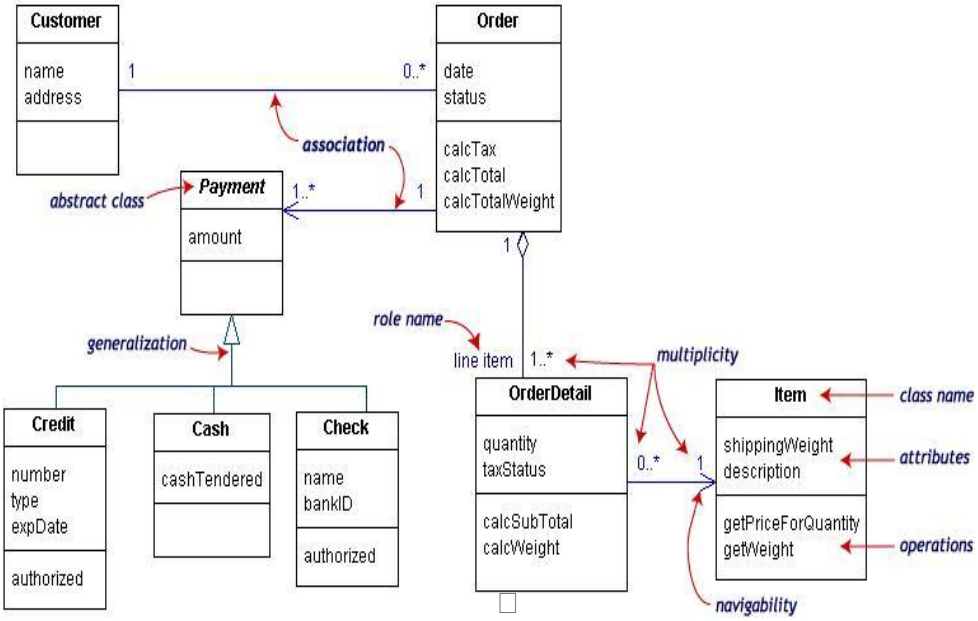
  Suatu hubungan antara bagian dari dua kelas. Terjadi *association* antara dua kelas jika salah satu bagian dari kelas mengetahui yang lainnya dalam melakukan suatu kegiatan. Di dalam diagram, sebuah *association* adalah penghubung yang menghubungkan dua kelas.

- 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.

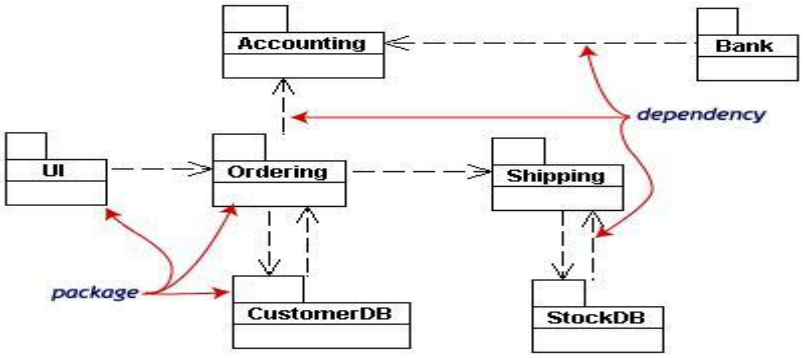


Gambar 3. 3 Contoh *Diagram Class* transaksi Pembelian barang.

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 :



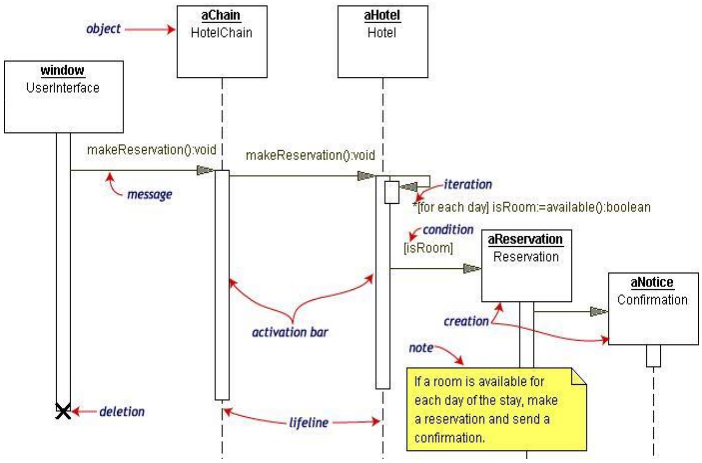
Gambar 3. 4 Contoh *Diagram Package*.

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 make Reservation ke Hotel Chain. Kemudian Hotel Chain, mengirim pesan yang sama ke Hotel. Bila Hotel, punya kamar kosong, maka dibuat Reservation dan Confirmation.

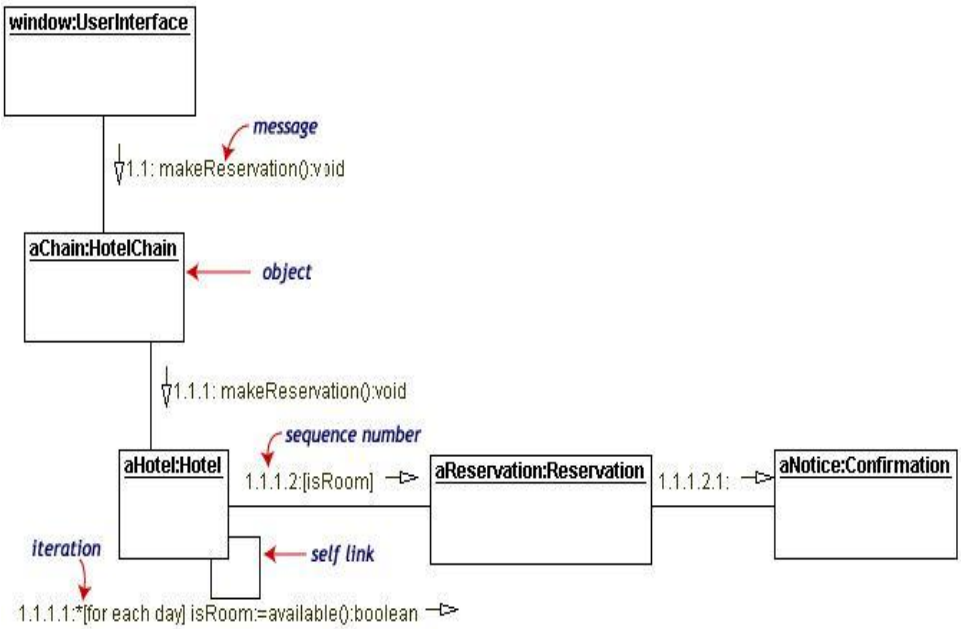
Lifeline adalah garis dot (putus-putus) vertikal pada gambar, menerangkan waktu terjadinya suatu obyek. Setiap panah yang ada adalah pemanggilan suatu pesan. Panah berasal dari pengirim ke bagian paling atas dari batang kegiatan (*activation bar*) dari suatu pesan pada *lifeline* penerima. *Activation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

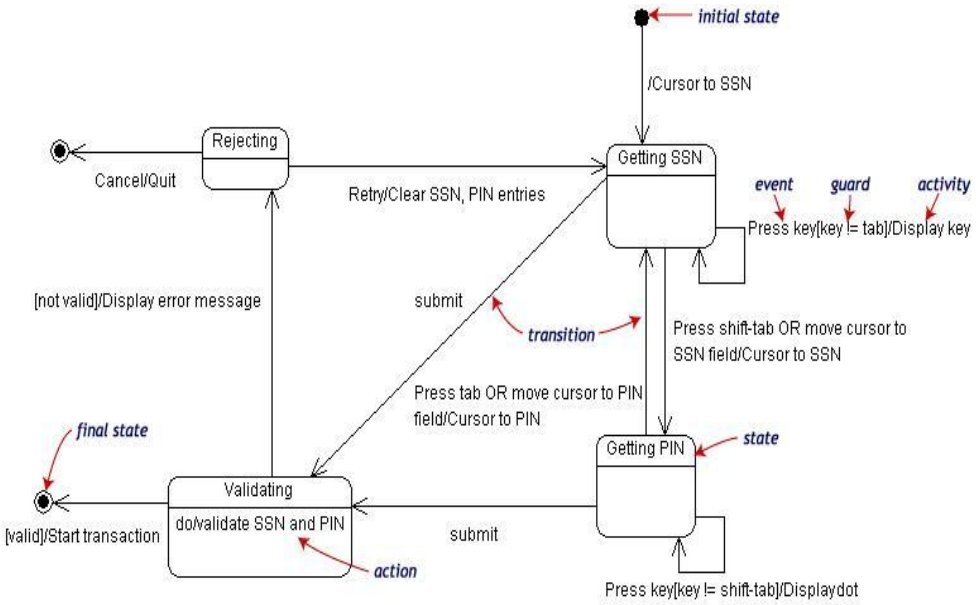
Kotak kegiatan obyek diberi label dengan nama kelas atau obyek (atau keduanya). Nama kelas dibatasi dengan *colons /titik dua* (:).

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 :

a. Getting SSN (masukkan SSN),
b. Getting PIN (masukkan PIN),
c. Validating (periksa kesahannya), dan
d. 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.

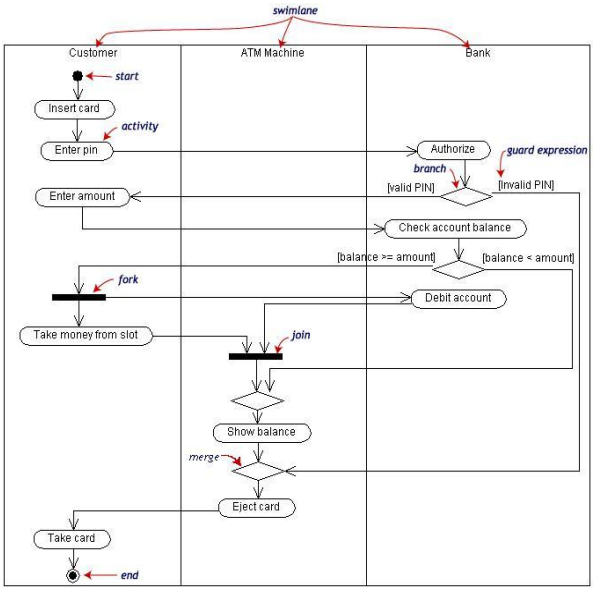
Aksi yang terjadi sebagai hasil dari suatu peristiwa atau keadaan ditandai dalam bentuk /action. Pada Validating State, obyek tidak menunggu peristiwa dari luar untuk menyebabkan suatu perubahan. Sebagai gantinya melakukan suatu *activity* (aktifitas). Hasil dari aktifitas tersebut menentukan keadaan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

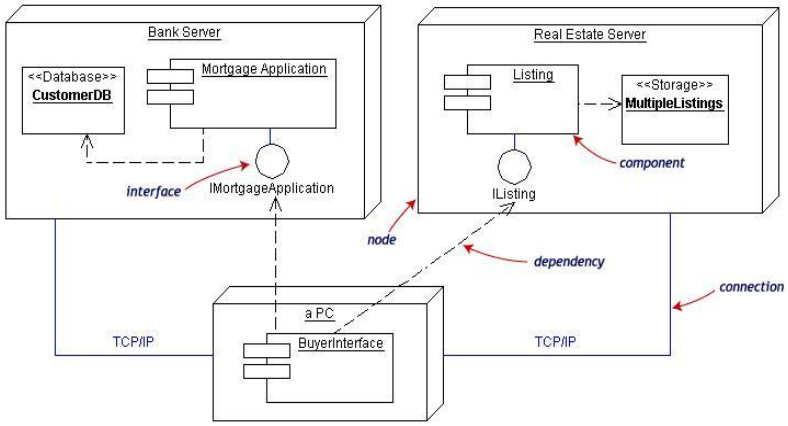
Diagram Activity dapat dibagi menjadi beberapa jalur kelompok yang menunjukkan obyek yang mana yang bertanggung jawab untuk suatu aktifitas. Peralihan tunggal (*single transition*) timbul dari setiap adanya *activity* (aktifitas), yang saling menghubungi pada aktifitas berikutnya.

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 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.