BAB II

LANDASAN TEORI

2.1 Koperasi Koperasi

Koperasi adalah badan usaha yang beranggotakan orang atau badan hukum koperasi dengan berlandaskan kegiatannya sebagai gerakan ekonomi rakyat yang berdasarkan atas asas kekeluargaan. Pengertian tersebut sesuai dengan UU Koperasi No. 25 thun 1992 Bab I. Selain itu, tujuan utama dibentuk koperasi juga telah dijelaskan dalam Undang-Undang Koperasi No. 25 Tahun 1992 Bab II pasal 3, bahwa tujuan utama koperasi adalah memajukan kesejahteraan anggota pada khususnya dan masyarakat pada umumnya serta ikut membangun tatanan perekonomian nasional dalam rangka mewujudkan masyarakat yang maju, adil, dan makmur berlandaskan Pancasila dan UUD 1945. Menurut Undang-Undang Koperasi No. 25 Tahun 1992 Bab II pasal 4, Fungsi dan peran koperasi adalah sebagai berikut:

a. Membangun dan mengembangkan potensi dan kemampuan ekonomi anggota pada khususnya dan masyarakat pada umumnya serta meningkatkan kesejahteraan ekonomi dan sosial mereka.
b. Berperan serta secara aktif dalam mempertinggi kualitas kehidupan manusia dan masyarakat.
c. Memperkokoh perekonomian rakyat sebagai dasar kekuatan dan ketahanan perekonomian nasional dengan koperasi sebagai soko gurunya.

d. Berusaha mewujudkan dan mengembangkan perekonomian nasional yang merupakan usaha bersama berdasarkan atas asas kekeluargaan dan demokrasi ekonomi.

2.1.1 Simpan Pinjam

Simpan pinjam merupakan salah satu unit kerja yang terdapat pada Koperasi Wanita Setia Bhakti Wanita. Simpan pinjam yang terdapat pada koperasi ini ada dua jenis yaitu simpan pinjam anggota dan simpan pinjam UKM (Unit Kegiatan Masyarakat). Anggota dalam koperasi ini harus membentuk kelompok tanggung renteng.

A. Simpan Pinjam Anggota

Simpan pinjam anggota hanya dapat dilakukan oleh anggota koperasi. Setiap kelompok tanggung renteng memiliki satu PJ (Penanggung Jawab) kelompok. PJ kelompok tersebut yang bertanggung jawab atas transaksi-transaksi yang dilakukan oleh kelompoknya. Transaksi-tansaksi yang dapat dilakukan oleh anggota adalah simpanan dan pinjam. Berikut jenis-jenis transaksi yang berlangsung pada Koperasi Wanita Setia Bhakti Wanita:

1. Simpan

   Transaksi simpanan dapat dilakukan oleh anggota tanggung renteng, baik diwakilkan oleh PJ kelompok ataupun dilakukan oleh perseorangan.

   a. Simpanan Pokok

      Simpanan pokok, dibayar sekali oleh calon anggota untuk syarat menjadi anggota yaitu sebesar Rp. 500.000.

b. Simpanan Wajib

Simpanan wajib, dibayar oleh anggota setiap bulannya paling sedikit sebesar Rp. 5.000 dan paling banyak Rp. 300.000.

c. Simpanan Sukarela

Simpanan sukarela, dapat dibayar oleh anggota tanpa ada jumlah ataupun jangka waktu tertentu. Jasa dari simpanan ini sebesar 0, 8% per bulan.

2. Pinjaman

Pengajuan pinjaman harus melalui musyawarah pada pertemuan masing-masing kelompok, dan disetujui oleh paling sedikit 50% ditambah satu orang dari jumlah anggota kelompok. Anggota juga harus meminta pesetujuan pinjaman dari PPL (Pembimbing Penyuluh Lapangan) dengan meminta tandatangan SPP (Surat Permohonan Pinjaman). Anggota dapat mengajukan pinjaman baru apabila telah mengangsur pinjamannya sebanyak 50%. Jasa pinjaman ini sebesar 1,8% perbulan. Sedangkan plafon merupakan batas pinjaman yang dapat dilakukan oleh anggota. Plafon dihitung berdasarkan dari jumlah simpanan yang dilakukan oleh anggota dan kelompok. Plafon kelompok ditetapkan sebesar 3, 50 kali jumlah simpanan wajib seluruh anggota kelompok, sedangkan plafon pribadi tergantung dengan jenis simpanan yang dipilih sesuai dengan ketentuan di atas.

a. Pinjaman SP I

Pinjaman jenis ini terkait dengan plafon kelompok. Pada jenis pinjaman ini ada 3 tahap pinjman yaitu:

1) Tahap I besar pinjaman sebesar ≥ Rp. 750.000 baru boleh dilunasi pada bulan ke-5.
2) Tahap II besar pinjaman sebesar ≥ Rp. 1.000.000.
3) Tahap III besar pinjaman sebesar 4 kali simpanan wajib yang sudah dibayar oleh anggota dan maksimal ≥ Rp. 11.000.000 dengan masa tenggang pembayaran yang ditentukan.

b. Pinjaman SP II

Pinjaman ini dapat dipinjam anggota yang sudah membayar 2 kali simpanan wajib maksimal ≥ Rp. 5.000.000 dan tidak berlaku masa tenggang pembayaran.

c. Pinjaman SP III

Pinjaman ini dapat dipinjam oleh anggota yang sudah menjadi anggota selama satu tahun. Besar pinjaman maksimal ≥ Rp. 2.100.000 dan dapat diangsur sebanyak 20 kali. Pinjaman ini terkait dengan plafon kelompok.

B. Simpan Pinjam UKM

Untuk melakukan transaksi simpan pinjam pada unit UKM (Unit Kecil Menengah) harus mendaftar menjadi ALB (Anggota Luar Biasa). Untuk mendaftar menjadi ALB, calon anggota harus membayar simpananpokok sebesar Rp. 250.000, dapat dibayar langsung atau diangsur maksimal 5 bulan. ALB juga harus membayar simpanan wajib sebesar RP. 2.500 perbulan, simpanan ini akan dibayarkan dimuka selama satu tahun yang dipotong dari realisasi pinjaman. Untuk meminjam ALB harus mengisi form SPP (Surat Pengajuan Pinjaman) dan menyerahkan jaminan. Petugas koperasi akan

memeriksa kelayakan jaminan tersebut, dan menyatakan apakah pinjaman tersebut diperbolehkan atau tidak. Selain simpanan pokok dan simpanan wajib, pada unit UKM juga menyediakan tabungan (simpanan pribadi). Untuk ALB yang ingin menabung, ALB harus mendaftar terlebih dahulu. Pihak koperasi akan membuatkan buku tabungan yang nantinya digunakan ALB untuk melakukan transaksi simpan tabungan.

2.1.2 Kenggotaan

Sistem anggota di koperasi ini memakai sistem tanggung renteng. Setiap satu kelompok tanggung renteng beranggotakan minimal 10 orang dan maksimal 30 orang. Dalam masing-masing kelompok terdapat satu orang PJ (Penanggung Jawab) kelompok. Untuk menjadi anggota koperasi, masing-masing calon anggota harus membayar simpanan pokok.

2.1.3 Griya Tamu

Griya tamu merupakan unit usaha koperasi yang menyediakan tempat singgah sementara. Griya tamu yang terdapat pada Koperasi Wanita Setia Bhakti Wanita ini tidak dikhususkan untuk anggota koperasi saja, yang bukan merupakan anggota koperasi juga dapat menginap di griya tamu ini. Griya tamu ini

2.1.4 Swalayan

Swalayan adalah unit kerja pada koperasi yang menjual kebutuhan sehari-hari. Pada swalayan ini tidak hanya melayani pembeli yang merupakan anggota koperasi, tetapi juga melayani pembeli yang bukan merupakan koperasi.

2.2 *Unified Modeling Langguage* (UML)

UML adalah sebuah bahasa pemodelan untuk menentukan, visualisasi, konstruksi dan mendokumentasikan artifacts (sepotong informasi yang digumakan atau dihasilkan dalam suatu proses rekayasa *software*, dapat berupa model, deskripsi, atau *software*) yang terdapat dalam sistem *software* (Suhendar, 2002). UML merupakan salah satu alat bantu yang sangat handal dalam dunia pengembangan sistem yang berorientasi objek (Munawar, 2005). Hal ini dikarenakan UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atau visi mereka dalam bentuk baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi dan mengkomunikasikan rancangan mereka dengan yang lain

UML bisa berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem, dengan demikian semua anggota tim akan mempunyai gambaran yang sama tentang suatu sistem. UML berfungsi sebagai cetak biru karena sangat lengkap dan detil, dengan cetak biru ini maka akan bisa diketahui informasi detil tentang coding program (*forward engineering*) atau bahkan membaca program dan mengintepretasikannya kembali di dalam diagram (*reverse engineering*). Tujuan UML diantaranya adalah:

1. Memberikan model siap pakai, bahasa pemodelan visual yang ekpresif untuk mengembangkan dan saling tukar menukar model dengan mudah dan dimengerti secara umum.
2. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemograman dan proses rekayasa.
3. Menyatukan praktek-praktek terbaikyang terdapat dalam pemodelan.

2.2.1 *Bussines Use case Diagram*

Bussines Use case Diagram digunakan untuk menunjukkan interaksi antara *use case* bisnis, aktor bisnis, dan pekerja bisnis (Sholiq, 2006). Diagram ini menjawab pertanyaan "apa yang bisnis lakukan dan mengapa harus membangun sistem". Diagram ini menunjukkan antara *use case* bisnis, aktor bisnis, dan pekerja bisnis dalam organisasi.

1. Aktor bisnis (*bussines actor*) adalah seseorang atau sesuatu yang ada di luar organisasi dan berinteraksi dengan organisasi yang terlibat dalam kegiatan bisnis organisasi. Aktor bisnis bisa berupa kelompok orang atau perusahaan.
2. Pekerja bisnis (*bussines worker*) adalah suatu peranan di dalam organisasi, bukan posisi. Seorang dapat memainkan banyak peranan tetapi memegang hanya satu posisi. Keuntungan peranan dibanding posisi adalah adalah bahwa posisi cenderung berubah, sementara peranan cenderung tetap.
3. *Use case* bisnis (*use case bussines*) adalah model yang digunakan menggambarkan proses bisnis organisasi. Dengan kata lain *use case* bisnis menceritakan kepada pembaca apa yang organisasi lakukan. Diagram ini tidak membedakan antara proses manual atau proses terkomputerisasi.

2.2.2 *Use case Diagram*

Diagram *use case* menyajikan interaksi antara *use case* dan aktor. Dimana aktor dapat berupa orang, peralatan, atau sistem yang lain yang berinteraksi denagn sistem yang sedang dibangun. *Use case* menggambarkan fungsionalitas sistem atau persyaratan-peryaratanyang harus dipenuhi sistem dari pandangan pemakai (Sholiq, 2006). Jika diagram *use case* bisnis tidak memperhatikan apakah proses

dilakukan secara otomatis terkomputerisasi, maka diagram *use case* berfokus hanya pada proses otomatisasi saja.

Use case dan aktor menggambarkan ruang lingkup sistem yang sedang dibangun.

1. Aktor (*actor*) adalah sesorang atau apa saja yang berhubungan dengan sistem yang berhubuangan dengan sistem yang sedang dibangun.
2. *Use case* adalah bagian tingkat tinggi dari fungsionalitas yang sedang disediakan oleh sistem. Dengan kata lain, *use case* menggambarkan bagaimana seseorang menggunakan sistem.

Pada *use case* diagram terdapat relasi antar *use case* untuk mendapatkan sistem secara utuh. Relasi asosiasi digunakan untuk menunjukkan relasi antar *use case* dan aktor. Sedangkan asosiasi arah panah mengindikasikan siapa yang mengawali komunikasi. Karena setiap *use case* harus diinialisasi oleh aktor. Relasi generalisai dilakukan untuk menunjukkan bahwa beberapa aktor mempunyai beberapa persamaan. Misalnya ada dua tipe pelanggan, yaitu pelanggan perusahaan dan pelanggan individu.

Selain relasi asosiasi dan generalisasi ada dua relasi yang lain, relasi include dan extend. Relasi include memungkinkan satu *use case* menggunakan fungsionalitas yang disediakan oleh *use case* yang lain. Sedangkan relasi *extend* memungkinkan satu *use case* secara optimal menggunakan funsionalitas yang disediakan oleh *use case* lainnya.

2.2.3 *Class Diagram*

Diagram kelas menunjukkan interaksi antar kelas dalam sistem (Sholiq, 2006). Kelas mengandung informasi dan perilaku (behavior) yang berkaitan denagn

informasi tersebut. Sebuah kelas pada diagram kelas dibuat untuk setiap objek pada Diagram Sekuensial atau Diagram Kolaborasi. Kelas memiliki tiga area kelompok yaitu:

1. Nama *(stereotype)*
2. Atribut *(attribute)*
3. Metode *(operation)*

Ada empat visibilitas untuk sebuah kelas, yaitu:

1. *Private*, suatu kelas tidak kelihatan oleh kelas lainnya atau tidak dapat dipanggil dari luar kelas yang bersangkutan.
2. *Protected*, hanya dapat dipanggil oleh kelas yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public*, dapat dipanggil oleh siapa saja yang terlibat kesemua kelas dan lainnya dalam sistem.
4. *Package* atau *implementation*, mengindiksikan bahwa kelas dapat dilihat hanya oleh kelas yang lain dalam paket yang sama.

2.2.4 *Behavior Diagram*

Behavior diagram berguna untuk memodelkan perilaku dinamis satu kelas atau objek, terdiri dari:

A. *Statechart Diagram*, menyediakan sebuah cara untuk memodelakan bermacam-macam keadaan yang mungkin dialami oleh sebuah objek (Sholiq, 2006). Jika dalam kelas diagram menunjukkan gambaran untuk memodelkan tingkah laku dinamik sistem. Diagram keadaan tidak dibuat untuk setiap kelas, bahkan kadang-kadang untuk suatu proyek sistem informasi tidak menggunakannya sama sekali.

B. *Activity Diagram* menggambarkan aliran fungsionalitas sistem (Sholiq, 2006). Diagram aktivitas tidak perlu dibuat untuk setiap aliran kerja, tetapi diagram ini akan sangat berguna untuk aliran kerja yang komplek dan luas.

C. *Interaction Diagram* menunjukkan langkah demi langkah di dalam *use case*. Obyek apa saja yang dibutuhkan untuk aliran, pesan apa saja yang objek kirimkan ke objek lainnya, dan urutan pesan-pesan yang dikirimkan. Diagram interaksi terdiri dari:

   1. *Sequence Diagram,* digunakan untuk menunjukkan aliran fungsionalitas dalam *use case* (Sholiq, 2006). Diagram sekuensial adalah diagram interaksi yang disusun berdasarkan waktu.
   2. *Collaboration Diagram* menunjukkan informasi yang sama persis dengan diagram sekuensial, tetapi lebih dalam bentuk dan tujuan yang berbeda. Diagram ini lebih menekankan pada hubuangan (*relationship*) antar objek-objek.

D. *Implementation Diagram* terdiri dari:

   1. *Component Diagram* menunjukkan model secara fisik komponen perangkat lunak pada sistem dan hubungannya antar mereka (Sholiq, 2006). Komponen hanya terdapat satu tipe relasi antar komponen yaitu dependensi. Dependensi menyatakan bahwa satu komponen bergantung pada lainnya, dan digambarkan seperti panah terputus-putus.
   2. *Deployment Diagram,* menampilakan rancangan fisik jaringan dimana berbagai komponen akan terdapat di sana (Sholiq, 2006). Dalam setiap sistem hanya ada satu *deployment diagram*, sehingga hanya ada satu *deployment diagram* dalam setiap model. Suatu *deployment diagram*

menampilkan semua titik (*node*) dalam suatu jaringan, hubungan antar mereka, dan proses-proses yang dijalankan pada masing-masing titik.

2.3 *Object oriented Analysis and Design* (OOAD)

Analisa berbasis objek (*Object oriented Analysis*) adalah metode analisis yang memeriksa *requirements* (syarat/keperluan yang harus dipenuhi sistem) dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup permasalahan. Sedangkan Desain orientasi Objek (*Object oriented Design*) adalah metode untuk mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau sub sistem (Suhendar, 2002). Terdapat berbagai konsep dalam OOAD yaitu objek, kelas, abstraksi, pewarisan, pembungkusan, asosiasi, dan agregasi.

2.3.1 Objek (*Object*)

Objek adalah "sesuatu", secara fisik atau konseptual, yang dapat kita temui disekeliling kita. *Hardware, software*, dokumen, bahkan konsep semuanya adalah contoh objek (Suhendar, 2002). Sebuah objek memiliki keadaan sesaat (*state*) dan perilaku (*behavior*). *State* dari sebuah objek adalah kondisi objek tersebut atau himpunan dari keadaan yang menggambarkan objek tersebut. Sebagai contoh bola lampu adalah objek, dan salah satu keadaan hidup atau mati adalah *state* dari objek bola lampu tersebut. *State* dinyatakan dengan nilai dan atribut objeknya. Atribut adalah nilai internal yang dimiliki sebah objek, yang meliputi karakteristik dari objek tersebut, kondisi sesaat, koneksi dengan objek lain, dan identitas. Perilaku suatu objek mendefinisikan bagaimana sebuah objek bereaksi dan memberi reaksi, yang ditentukan oleh himpunan semua atau beberapa operasi yang dapat dilakukan

dalam objek itu sendiri. *Behavior* dari sebuah objek dicerminkan oleh *interface, service*, dan *method* dari objek tersebut. *Interface* adalah pintu untuk mengakses *service* objek. *Service* adalah fungsi yang bisa diemban oleh objek. *Method* adalah mekanisme internal objek yang mencerminkan perilaku objek tersebut.

2.3.2 Kelas (*Class*)

Kelas adalah definisi umum (pola, *template*, atau cetak biru) untuk himpunan objek sejenis. Kelas menetapkan spesifikasi perilaku dan atribut objek-objek tersebut (Suhendar, 2002). Jadi kelas adalah sesuatu yang membungkus informasi dan perilaku. Sebuah objek merupakan contoh dari sebuah kelas. Kucing, harimau dan singa adalah contoh objek dari kelas binatang. Objek mempunyai atribut dan operasi. Atribut dari operasi di atas adalah berkaki empat, memiliki ekor dan lain sebagainya. Sedangkan operasi dari objek tersebut adalah makanan, minum, tidur dan lain sebagainya.

2.3.3 Abstraksi (*Abstraction*)

Abstraksi secara sederhana dikatakan sebagai *filter-out* atribut objek dan operasi objek hanya sampai pada benar-benar diperlukan saja. Tipe yang berbeda dari persoalan memerlukan nilai informasi yang berbeda, jika persoalan itu ada di dalam area global data yang sama. Sehingga atribut-atribut dan operasi-operasi yang diperlukan saja didefinisikan. Metode ini dikenal dengan istilah abstraksi dari suatu objek.

2.3.4 Pewarisan (*Inheritance*)

Pewarisan didefinisikan sebagai sebuah *relationship* antara kelas di mana satu kelas dapat membagi struktur dan operasi-operasi untuk satu atau beberapa

kelas. Objek adalah instan suatu kelas, maka objek mempunyai semua karakteristik dari suatu kelas. Atribut dan operasi yang ditentukan dalam kelas akan tewariskan ke masing-masing objek dalam kelas tersebut. Kelas dapat pula mewarisi sifat-sifat kelas lainnya. Sebagai contoh *washing machine, microwave, oven,* radio, televisi dan sebagainya adalah kelas peralatan misalnya tipe, dan mewarisi operasi misalnya *turn-on* dan *trun-off*.

2.3.5 Banyak Bentuk (*Polymorphism*)

Kadang-kadang sebuah operasi mempunyai nama yang sama pada kelas yang berbeda. Sebagai contoh membuka jendela, membuka pintu, membuka surat kabar, membuka percakapan. Dalam masing-masing persoalan dapat dilakukan operasi berbeda-beda walaupun dengan nama yang sama.

Konsep di atas dikenal dengan istilah banyak bentuk, yaitu suatu operasi dengan nama yang sama, tetapi jika dikenalkan pada objek yang berbeda akan menghasilkan operasi yang berbeda.

2.3.6 Pembungkusan (*Encapsulation*)

Ketika seorang menonton televisi, biasanya orang tersebut tidak mempedulikan kompleksitas rangkaian elektronika yang ada di dalamnya. Mereka tidak memperdulikan bagaimana rangkaian elektronika tersebut bekerja, mereka hanya memperdulikan tombol-tombol apa saja yang bisa digunakan untuk mengoperasikannya. Konsep ini dikenal dengan istilah pembungkusan, yaitu proses menyembunyikan detail implementasi sebuah objek. Satu-satunya jalan untuk mengakses objek tersebut adalah melalui interface. Interface melindungi internal state sebuah objek dari ”campur tangan” pihak luar. Oleh karena itu objek sering

digambarkan sebagai kotak hitam (*black box*) yang menerima dan mengirim pesan-pesan (*messages*).

Dalam *object oriented programming* kotak hitam tersebut berisi kode (himpunan instruksi dengan bahasa yang dipahami komputer) dan data (informasi dimana intruksi tersebut beropasi dengannya). Kode dan data tersebut disatukan dalam sebuah "benda" yang tersembunyi isinya, yaitu objek. Pengguna objek tidak perlu mengetahui isi dalam kotak tersebut. Untuk dapat berkomunikasi dengan objek, diperlukan pesan.

2.3.7 Asosiasi (*Association*) dan Agresiasi (*Aggregation*)

Asosiasi adalah hubungan antar objek yang saling membutuhkan (Suhendar, 2002). Sebagai contoh, saat seseorang menyalakan sebuah televisi maka menurut terminologi berorientasi objek, seseorang tersebut sedang berasosiasi dengan televisi.

Agresiasi adalah bentuk khusus dari asosiasi yang menggambarkan seluruh bagian suatu objek merupakan bagian dari objek lainya (Suhendar, 2002). Sebagai contoh, komputer terdiri dari CPU, *keyboard*, *mouse*, monitor, printer, dan lain sebagainya. Di dalam kotak CPU mungkin terdapat *grafikcard, soundcard*, dan peralatan lainya. Komputer adalah sebuah agresiasi, komputer terdiri dari sejumlah komponen-komponen berbeda sebagai penyusunya.

2.4 *Rational Unified Process* (RUP)

Dalam membangun suatu sistem informasi diperlukan metode Siklus Hidup dan Pengembangan Sistem (*System Development Life Cycle* atau SDLC). SDLC terdiri dari sejumlah tahapan yang dilaksanakan secara berurutan. Secara

umum tahapan dari SDLC adalah perencanaan, analisis, rancangan, penerapan dan penggunaan. Seringkali terdapat kesalahan dalam satu tahapan sehingga menyebabkan siklus ini harus diulangi dari tahapan yang salah. Bisa terjadi bahwa siklus ini dilakukan sampai berulang-ulang. Maka dari itu perlu digunakannya metode untuk membangun sistem informasi.

RUP merupakan suatu metode SDLC yang dikembangkan dengan mengumpulkan berbagai *best practices* yang terdapat dalam industri pengembangan perangkat lunak. Ciri utama metode ini adalah menggunakan *Use case Driven* dan pendekatan *iterative* untuk siklus pengembangan perangkat lunak (Taryana, 2007). Sedangkan menurut sholiq (2006) penggunaan RUP dilakukan melalui 4 fase yaitu :

A. Insepsi

Fase insepsi adalah permulaan proyek. Fase ini dimulai dengan membuat pemodelan bisnis, dengan menganalisis bisnis sekitar sistem yang akan dibangun. Kita berusaha menemukan fitur-fitur level atas sistem dan mendokumentasikan secara baik. Dengan menggunakan UML, fase ini dapat kita lakukan dengan membuat *use case* bisnis, aktor bisnis, pekerja bisnis, dan diagram *use case* bisnis. Mungkin saja juga membuat diagram aktivitas untuk memodelkan *workflow*.

B. Elaborasi

Tujuan fase *elaborasi* adalah untuk menganalisis domain masalah, menguatkan arsitektur sistem, menggembangkan rencana proyek, dan mengurangi unsur-unsur resiko tertinggi proyek. Fase elaborasi meliputi beberapa kegiatan antara lain analisis dan desain arsitektur. Dengan mengikuti perencanaan iterasi,

elaborasi dilakukan untuk setiap *use case*. Kebutuhan level operasional pada setiap *use case* meliputi alur proses di dalam *use case*, aktor apa saja yang berinteraksi denagn *use case*, diagram sekuensial dan diagram kolaborasi untuk menunjukkan alur proses secara grafik, dan diagram *statechart* untuk menunjukkan perubahan kondisi yang mungkin terjadi di dalam *use case*.

C. Konstruksi

Fase konstruksi menyempurnakan hasil-hasil yang telah dicapai di fase *elaborasi*. Fase konstruksi dimulai dengan membuat diagram komponen untuk sistem, kemudian dengan bantuan tool tertentu seperti *Rational Rose* dapat dibangkitkan struktur kode ke bahasa pemograman tertentu seperti Java, Visual Basic, dan lain-lain. Struktur kode yang dibangkitkan dalam format berorientasi obyek yang terdiri dari deklarasi kelas, deklarasi atribut, deklarasi skope (*private*, *protected*, dan *public*), *prototype* fungsi, dan pernyataan perwarisan. Kode yang telah dibangkitkan tentu saja terbatas pada struktur kode bahasa pemograman tertentu, sehingga diperlukan penyempurnaan oleh para *programmer*. Jika ada penambahan atribut baru, fungsi-fungsi baru, atau relasi baru yang sifatnya adalah penambahan minor pada fase konstruksi maka dapat langsung ditambahkan. Kemudian model yang telah dibuat diperbarui dengan menggunakan fasilitas *reverse engineering* pada tool seperti *Rational Rose*. *Reverse engineering* adalah salah satu fasilitas di aplikasi pemodelan yang memungkinkan proses pembaruan model yang telah dibuat dilakukan otomatis dari kose sumber.

D. Transisi

Fase transisi dilakukan ketika perangkat lunak yang diproduksi telah selesai dan *delivery* ke pemakaian dilakukan. Tugas-tugas di fase ini meliputi: melengkapi dan menyempurnakan perangkat lunak, melengkapi *acceptance* testing akhir, melengkapi manual *user guide*, dan menyiapkan pelatihan pemakaian.