

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/303840922>

OBJECT ORIENTED MODELLING WITH UNIFIED MODELING LANGUAGE (UML)

Research · June 2016

DOI: 10.13140/RG.2.1.3464.4088

CITATION

1

READS

12,783

1 author:



Henderi Henderi

Universitas Raharja

75 PUBLICATIONS 71 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



OBJECT ORIENTED MODELLING WITH UNIFIED MODELING LANGUAGE (UML) [View project](#)

OBJECT ORIENTED MODELLING WITH UNIFIED MODELING LANGUAGE (UML)



Oleh: Henderi, M. Kom.

DIKTAT PERANCANGAN SISTEM INFORMASI (SI-111/3 SKS)

STMIK RAHARJA - TANGERANG

TAHUN 2009/2010



KATA PENGANTAR

Atas rahmat, petunjuk dan izin Allah SWT penulis dapat menyusun dan menyelesaikan Diktat Kuliah Perancangan Sistem Informasi dengan judul OBJECT ORIENTED DESIGN WITH UNIFIED MODELING LANGUAGE (UML) yang ada ditangan pembaca saat ini.

Diktat Kuliah Perancangan Sistem menggunakan UML ini disusun dengan tujuan agar pada pembelajar tentang Perancangan Sistem Informasi dan UML khususnya Pribadi Raharja dapat lebih memahami konsep UNIFIED MODELING LANGUAGE (UML) dan mempunyai kemampuan menggunakan UML dalam merancang sistem informasi berorientasi objek dan berbasis visual.

Seperti telah diketahui bahwa metodologi rancang bangun sistem informasi saat ini sudah beralih dari metode terstruktur menjadi berorientasi objek dan berbasis visual. Perkembangan ini sejalan dengan perkembangan dan penerapan analisis sistem dan pemrograman berorientasi objek dan visual. Karenanya Diktat ini diharapkan dapat menjadi suplemen dalam memahami dan menerapkan perancangan sistem informasi berorientasi objek dan visual menggunakan UNIFIED MODELING LANGUAGE (UML).

Diktat kuliah Perancangan Sistem Informasi ini disusun dan memuat materi: *Sejarah UML, Definisi dan Konsep Pemodelan Menggunakan UML yang terdiri dari: Use Case Diagram, Class Diagram, Sequence Diagram, Statechart/State Machine Diagram, dan Activity Diagram.*

Pada kesempatan ini, penulis mengucapkan terimakasih dan penghargaan yang tak terhingga kepada isteri tercinta W. Lestari, putra dan putri tercinta: Amir Acalapati Henry dan Carissa Azarine Henry yang telah menghiasi hari-hari penulis selama proses penyusunan dan editing. Terimakasih juga penulis tujukan kepada manajemen dan staf akademik Perguruan Tinggi dan STMIK Raharja, Sekretaris Asdir Raharja (Ibu Euis Melinda), dan seluruh mahasiswa yang mengikuti matakuliah Perancangan Sistem yang penulis ampuh yang telah menjadi motivasi bagi penulis dalam menyusun Diktat ini. Penulis juga menyampaikan terimakasih kepada mahasiswa terbaik penulis dalam memahami UML yaitu: Arie dan Evana, sekaligus sebagai partner penulis dalam mendalami dan memperkenalkan UML kepada civitas akademika Perguruan Tinggi Raharja Tangerang.

Tak ada gading yang tak retak sehingga penyusun menyadari bahwa penyusunan dan penulisan materi dan isi dari pada Diktat ini belum sempurna. Pada akhirnya saran dan kritik yang membangun dari pembaca akan penulis terima dengan lapang dada untuk perbaikan penulisan dimasa yang akan datang. Semoga Diktat ini dapat bermanfaat, amin.

Tangerang, 5 November 2009

Penyusun

Henderi



DAFTAR ISI

	hal
KATA PENGANTAR	2
DAFTAR ISI	3
BAB I PENDAHULUAN	5
1.1. Sejarah Unified Modeling Language (UML).....	5
1.2 Definisi Unified Modeling Language (UML).....	5
1.3 Konsep Pemodelan Menggunakan Unified Modeling Language (UML)	6
1.3.1 Diagram Dasar dalam Unified Modeling Language (UML)	6
BAB II USE CASE DIAGRAM	8
2.1 Definisi Use Case Diagram	8
2.2 Simbol dan Notasi Dasar Use Case Diagram	8
2.3 Contoh Use Case Diagram	9
2.4 Membuat Use Case Diagram Menggunakan Visual Paradigm	11
Soal Latihan Use Case Diagram	15
BAB III CLASS DIAGRAM	16
3.1 Definisi Class Diagram	16
3.2 Simbol dan Notasi Dasar Class Diagram	16
3.3 Contoh Class Diagram	17
3.4 Membuat Class Diagram Menggunakan Visual Paradigm.....	19
3.5 Membuat Association Antara Class	26
3.6 Melengkapi Class dengan Operations	28
3.7 Melengkapi Class Diagram dengan Multiplicity	30
Soal Latihan	32
BAB IV SEQUENCE DIAGRAM	33
4.1 Definisi Sequence Diagram	33
4.2 Simbol dan Notasi Dasar Sequence Diagram	33
4.3 Contoh Sequence Diagram	35
4.4 Membuat Sequence Diagram Menggunakan Visual Paradigm	36
4.4.1 Membuat Actor dan Object pada Sequence Diagram	36
4.4.2 Membuat Message pada Sequence Diagram	39
Soal Latihan	41
BAB V STATECHART DIAGRAM	42
5.1 Definisi Statechart Diagram	42
5.2 Simbol dan Notasi Dasar Statechart Diagram	42
5.3 Contoh Statechart Diagram	42
5.4 Membuat Statechart Diagram Menggunakan Visual Paradigm	44
5.4.1 Membuat Initial Pseudo State Awal State dan State Pertama.....	44
5.4.2 Melengkapi State dengan Event/Action	47
5.4.3 Menghubungkan State dengan State.....	49
5.4.4 Membuat Akhir State.....	50
Soal Latihan	51



BAB VI	ACTIVITY DIAGRAM	53
6.1	Definisi Activity Diagram	53
6.2	Simbol dan Notasi Dasar Activity Diagram	53
6.3	Contoh Activity Diagram	54
6.4	Membuat Activity Diagram Menggunakan Visual Paradigm	55
6.4.1	Mengaktifkan Worksheet dan Membuat Awal Activity	55
6.4.2	Memberi Nama Action State dan Action Flow	58
6.4.3	Membuat Join Node	61
6.4.4	Membuat Decision Node	64
6.4.5	Membuat Flow Final Node	66
6.4.6	Membuat Activity Final Node	68
	Soal Latihan	69
CASE STUDY 1		70
CASE DTUDY 2		71
CASE DTUDY 3		72
DAFTAR SIMBOL YANG DIGUNAKAN DALAM UML		73
PUSTAKA		



BAB I

PENDAHULUAN

1.1 Sejarah Unified Modeling Language (UML)

Tahun 1994, Grady Boch dan James Rumbaugh bergabung untuk menggunakan metode berorientasi objek. Ivan Jacobson bergabung pada tahun 1995, dan mereka bertiga fokus membuat suatu bahasa pemodelan objek standar sebagai ganti dari pendekatan atau metode objek standar. Berdasarkan kerja mereka dan hasil kerja lainnya pada industri, **Unified Modeling Language (UML)** versi 1.0 dirilis pada tahun 1997.

Unified Modeling Language (UML) tidak menentukan metode untuk sistem-sistem pengembangan, tetapi sudah diterima luas sebagai standar untuk pemodelan objek. Object Management Group/OMG, badan standar industri, mengadopsi UML pada bulan November 1997 dan terus bekerja sama untuk meningkatkannya berdasarkan kebutuhan industri. Pada saat ini, salah satu industri telah merilis sebuah software yang mendukung UML yaitu Visual Paradigm 6.4 Enterprise edition. Berbagai industri juga bermunculan dan mendukung penggunaan UML dengan berbagai produk, diantaranya Rational Rose, SmartDraw, dan lain-lain.

1.2 Definisi Unified Modeling Language (UML)

Unified Modeling Language (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek (Whitten L. Jeffery et al, 2004). Sementara menurut Henderi (2007: 4) **Unified Modeling Language (UML)** adalah sebuah bahasa pemodelan yang telah menjadi standar dalam industri software untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak. Bahasa Pemodelan UML lebih cocok untuk pembuatan perangkat lunak dalam bahasa pemrograman berorientasi objek (C++, Java, VB.NET), namun demikian tetap dapat digunakan pada bahasa pemrograman prosedural (Ziga Turck, 2007)

Unified Modeling Language (UML) biasa digunakan untuk (Henderi, 2007 :11)

1. Menggambarkan batasan sistem dan fungsi-fungsi sistem secara umum, dibuat dengan use case dan actor
2. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan interaction diagrams
3. Menggambarkan representasi struktur statik sebuah sistem dalam bentuk class diagrams
4. Membuat model behavior "yang menggambarkan kebiasaan atau sifat sebuah sistem" dengan state transition diagrams
5. Menyatakan arsitektur implementasi fisik menggunakan component and development diagrams
6. Menyampaikan atau memperluas functionality dengan stereotypes (Ziga Turck, 2007)

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa

aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. UML mempunyai banyak diagram yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun. Diagram-diagram tersebut digunakan untuk (Henderi et al, 2008:71):

1. Mengkomunikasikan ide
2. Melahirkan ide-ide baru dan peluang-peluang baru
3. Menguji ide dan membuat prediksi
4. Memahami struktur dan relasi-relasinya

1.3 Konsep Pemodelan Menggunakan Unified Modeling Language (UML)

Pemodelan menggunakan Unified Modeling Language merupakan metode pemodelan berorientasi objek dan berbasis visual. Karenanya pemodelan menggunakan UML merupakan pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML menawarkan diagram yang dikelompokkan menjadi lima perspektif berbeda untuk memodelkan suatu sistem. Seperti satu set *blue print* yang digunakan untuk membangun sebuah rumah.

1.3.1 Diagram Dasar dalam Unified Modeling Language (UML)

Berikut ini adalah penjelasan mengenai berbagai diagram UML serta tujuannya:

1. Model Use Case Diagram

Use Case Diagram secara grafis menggambarkan interaksi antara sistem, sistem eksternal, dan pengguna. Dengan kata lain Use Case diagram secara grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (user) mengharapkan interaksi dengan sistem itu. Use Case secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi.

2. Diagram Struktur Statis

UML menawarkan dua diagram untuk memodelkan struktur statis sistem informasi, yaitu:

a. Class Diagram: menggambarkan struktur object sistem. Diagram ini menunjukkan class object yang menyusun sistem dan juga hubungan antara class object tersebut

b. Object Diagram: serupa dengan class diagram, tetapi object diagram memodelkan instance object actual dengan menunjukan nilai-nilai saat ini dari atribut instance. Object Diagram menyajikan “snapshot/potret” tentang objek sistem pada point waktu tertentu. Diagram ini tidak digunakan sesering Class Diagram, tetapi saat digunakan dapat membantu seorang developer memahami struktur sistem secara lebih baik.

3. Diagram Interaksi

Diagram interaksi memodelkan sebuah interaksi, terdiri dari satu set objek, hubungan-hubungannya, dan pesan yang terkirim di antara objek. Model diagram ini memodelkan behavior (kelakuan) sistem yang dinamis dan UML memiliki dua diagram untuk tujuan ini, yaitu:

a. Diagram rangkaian/Sequence Diagram: secara grafis menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada sekuensi sebuah use case atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima di antara objek dan dalam sekuensi atau timing apa.

b. Diagram kolaborasi/Collaboration Diagram: serupa dengan diagram rangkaian/sekuensi, tetapi tidak fokus pada timing atau sekuensi pesan. Diagram ini

justru menggambarkan interaksi (atau kolaborasi) antara objek dalam sebuah format jaringan.

Diagram rangkaian maupun diagram kolaborasi merupakan isomorphic artinya kita dapat mengubah dari satu diagram ke diagram lain.

4. Diagram State/State Diagram

UML memiliki sebuah diagram untuk memodelkan behavior objek khusus yang kompleks (statechart) dan sebuah diagram untuk memodelkan behavior dari sebuah use case atau sebuah metode, yaitu:

a.Diagram statechart: digunakan untuk memodelkan behavior objek khusus yang dinamis. Diagram ini mengilustrasikan siklus hidup objek-berbagai keadaan yang dapat diasumsikan oleh objek dan event-event (kejadian) yang menyebabkan objek beralih dari satu state ke state lain.

b.Diagram aktivitas/Activity Diagram: secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun use case. Activity diagram dapat juga digunakan untuk memodelkan action yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari action tersebut.

5. Diagram Implementasi

Diagram implementasi juga memodelkan struktur sistem informasi, yaitu:

a.Diagram komponen/Component Diagram: digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen software sistem. Komponen diagram dapat digunakan untuk menunjukan bagaimana kode pemrograman dibagi menjadi modul-modul (atau komponen).

b.Diagram penguraian/Deployment: digunakan untuk mendeskripsikan arsitektur fisik dalam istilah "node" untuk hardware dan software dalam sistem. Diagram ini menggambarkan konfigurasi komponen-komponen software *real-time*, prosesor, dan peralatan yang membentuk arsitektur sistem.

Tidak semua jenis diagram di atas dibahas dDalam Buku I ini melainkan hanya lima diagram saja yaitu: Use Case Diagram, Class Diagram, Sequence Diagram, Statechart Diagram, dan Activity Diagram. Sementara itu diagram implementasi akan dibahas lebih lanjut dalam Buku II (Advance Modeling With Unified Modeling Language/UML).

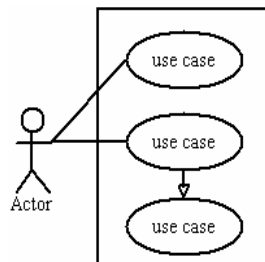
BAB II

USE CASE DIAGRAM

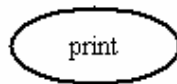
2.1 Definisi Use Case Diagram

Use Case diagram adalah model fungsional sebuah sistem yang menggunakan *actor* dan *use case*. *Use Case* adalah layanan (*services*) atau fungsi-fungsi yang disediakan oleh sistem untuk pengguna-penggunanya (Henderi et al, 2008). *Use Case* adalah suatu pola atau gambaran yang menunjukkan kelakuan atau kebiasaan sistem. Setiap *Use Case* adalah suatu urutan (*sequence*) transaksi yang saling berhubungan dan dilakukan oleh sebuah actor dan sistem dalam bentuk sebuah dialog (Henderi, 2007). *Use Case Diagram* dibuat untuk memvisualisasikan/ menggambarkan hubungan antara Actor dan *Use Case*. *Use Case diagram* mempresentasikan kegunaan atau fungsi-fungsi sistem dari perspektif pengguna.

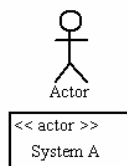
2.2 Simbol dan Notasi Dasar Use Case Diagram



Gambar 2.1



Gambar 2.2



Gambar 2.3

Sistem

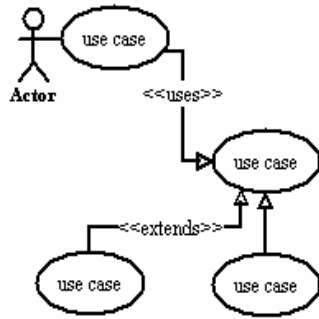
Gambar batasan (boundries) sebuah sistem menggunakan empat persegi panjang yang berisi use case-use case. Tempatkan actor-actor yang terlibat pada setiap use case pada bagian luar boundaries sistem (gambar 2.1)

Use Case

Gambar use cases menggunakan lingkaran berbentuk bulat telur (ovals) Beri nama ovals tersebut dengan kata kerja (verbs) yang menggambarkan fungsi-fungsi sistem (gambar 2.2)

Actors

Actors adalah para pengguna (users) dari sebuah sistem. Kadangkala sebuah sistem adalah merupakan actors bagi sistem yang lain, beri nama actors sistem tersebut dengan stereotipe (bentuk klise/tiruan) actor (gambar 2.3). Actor adalah seseorang atau sesuatu yang harus berinteraksi dengan sistem atau sistem yang dibangun/dikembangkan

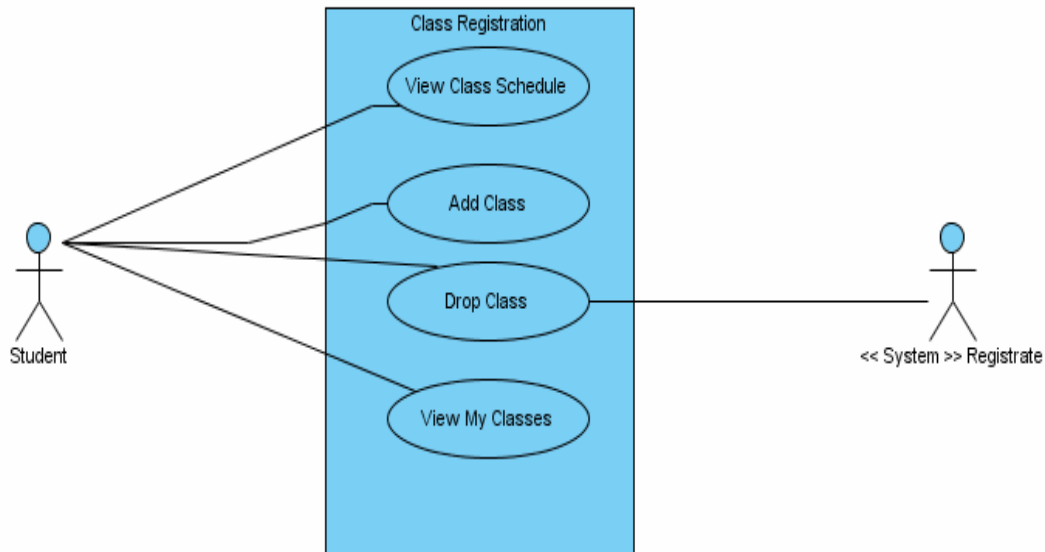


Gambar 2.4

Relationships

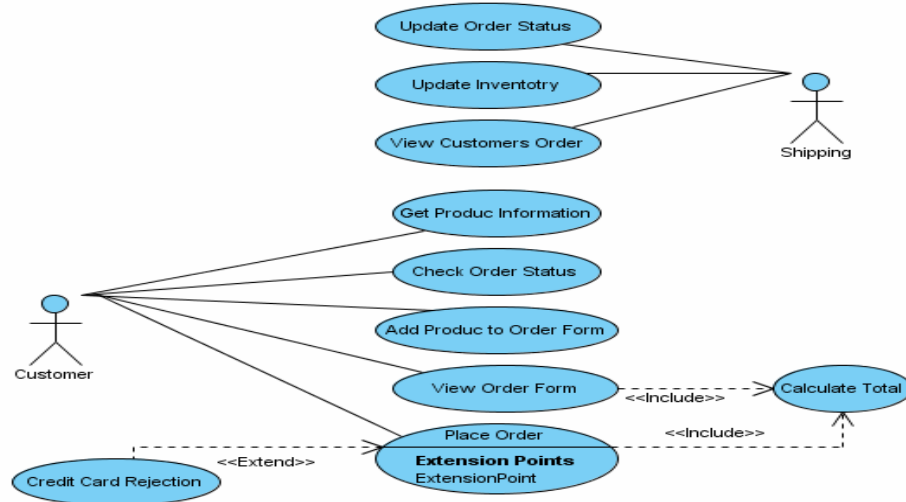
Ilustrasikan atau gambaran relasi/hubungan antara sebuah actor dan use case dengan sebuah garis sederhana. Untuk relasi-relasi antara use cases, gunakan tanda anak panah-anak panah pada "uses" yang lainnya atau gunakan "extends". Suatu relasi "uses" mengindikasikan bahwa ada use case yang dibutuhkan oleh use case yang lain untuk melakukan sebuah permintaan (task). Sementara suatu relasi "extends" mengindikasikan beberapa alternatif opsi (pilihan) tertentu pada tingkatan yang lebih bawah yang ada pada use case (gambar 2.4)

2.3 Contoh Use Case Diagram



Gambar 2.5 Use Case Diagram Class Registration (Henderi, 2008)

Berdasarkan gambar 2.5 Use Case Diagram Class Registration di atas diketahui bahwa sistem pendaftaran kelas mempunyai empat use case yang menunjukkan fungsi sistem. Sementara itu actor atau para pengguna (*users*) yang terlibat pada sistem class registration ini terdiri dari dua yaitu pelajar (*student*) dan sistem registrasi sebagai sistem yang lain.

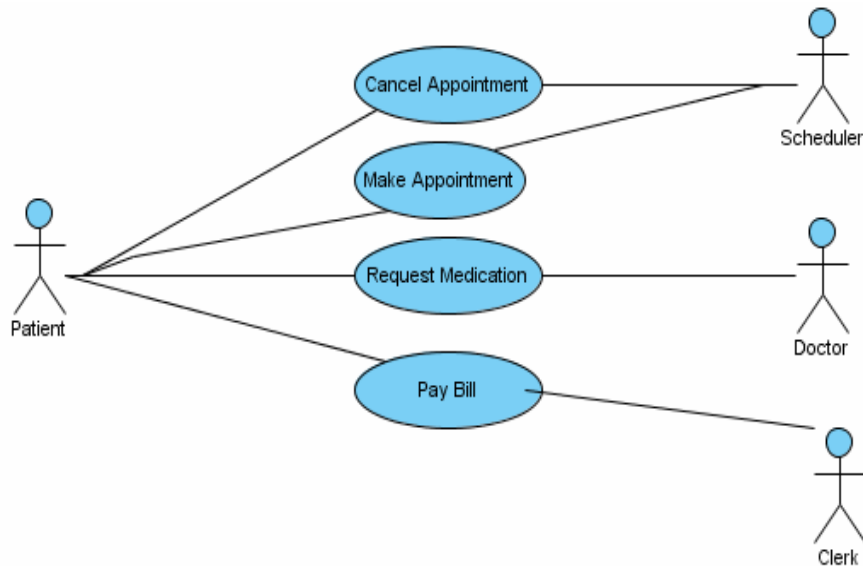


Gambar 2.6 Use Case Diagram Credit Card Processing (Henderi, 2008)

Gambar 2.6 di atas merupakan Use Case Diagram Credit Card Processing yang mempunyai sepuluh fungsional sistem, dan melibatkan dua pengguna (*actor*). Use Case diagram ini juga memiliki satu fungsional extend dan satu fungsional include. Fungsional

extend dan include akan dibahas lebih lanjut dalam buku Advance Unified Modeling Language: Konsep dan Penerapannya dalam Perspektif yang Lebih Luas (Book II).

Berikut ini adalah contoh Use Case Diagram Fungsional sistem yang ada disebuah rumah obat. Use Case ini memvisualisasikan hubungan antara actort patient, Scheduler, Dokter, dan Clerk dan keempat Use Case yang menggambarkan fungsi dan jelas services yang ada pada sistem.



Gambar 2.7 Use Case Diagram Periksa ke Dokter (Henderi, 2008)

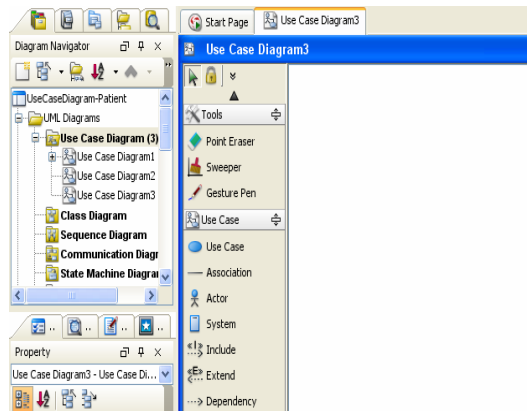
2.4 Membuat Use Case Diagram dengan Software Visual Paradigm (VP)

1. Aktifkan software VP hingga tampil dialog utama berikut



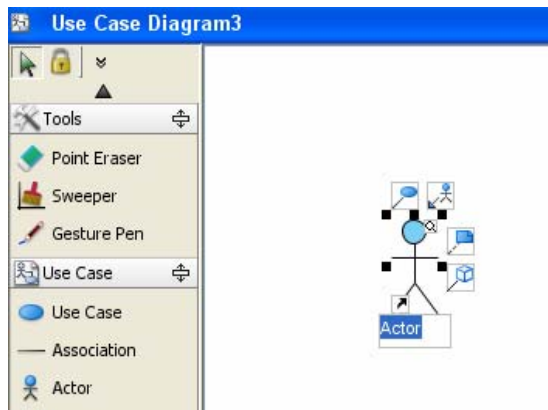
Gambar 2.8

2. Tempatkan kursor pada fungsi New Use Case Diagram pada layar dialog
3. Klik New Use Case Diagram hingga sistem menampilkan gambar berikut



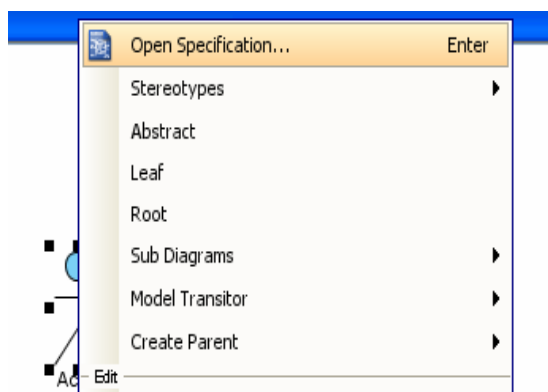
Gambar 2.9

4. Klik simbol actor yang ada pada daftar simbol Use Case
5. Tempatkan kursor pada worksheet Use Case diagram
6. Klik mouse satu kali pada worksheet Use Case diagram hingga tampil simbol actor seperti gambar berikut



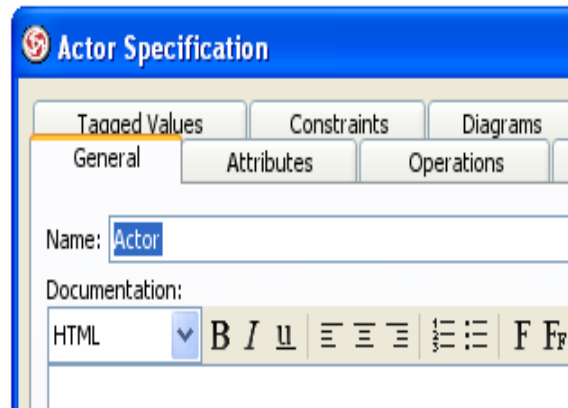
Gambar 2.10

7. Klik kanan mouse pada simbol actor yang aktif hingga sistem menampilkan dialog berikut:



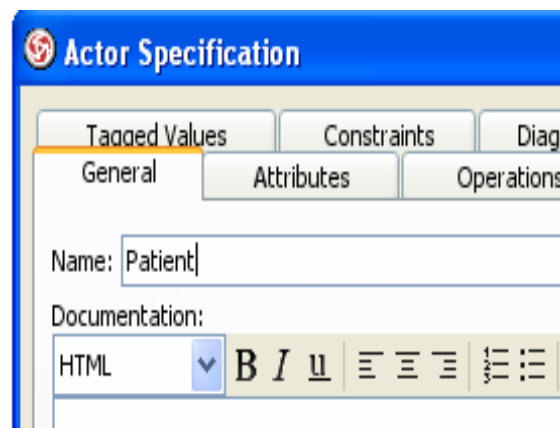
Gambar 2.11

8. Klik dialog **Open Specification** pada layar dialog di atas
9. Ketika tampil dialog berikut, ketikkan nama **actor** (pengganti actor pada dialog name) sesuai kebutuhan



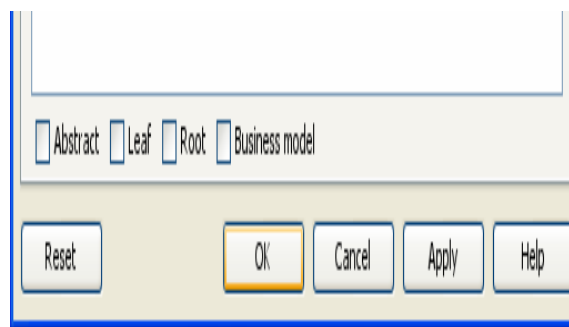
Gambar 2.12

10. Ketikkan nama actor sesuai kebutuhan pada bagian Name, misalnya **Patient**



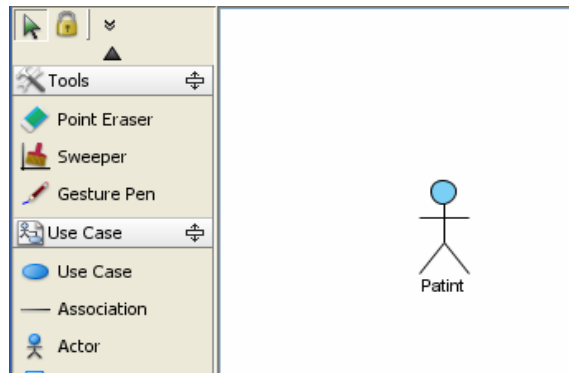
Gambar 2.13

11. Klik **Apply** dan **OK** pada layar dialog worksheet seperti pada gambar berikut



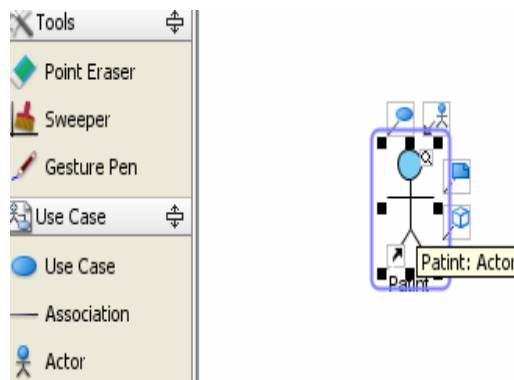
Gambar 2.14

12. Sistem akan menampilkan simbol **actor** yang sudah diberi nama **Patient**



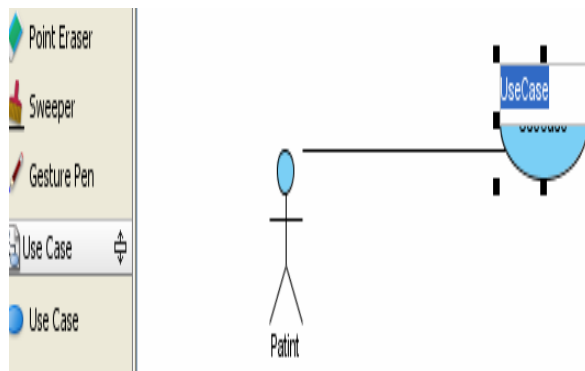
Gambar 2.15

13. Aktifkan simbol actor dengan cara meletakkan kursor pada simbol actor hingga tampilannya seperti berikut



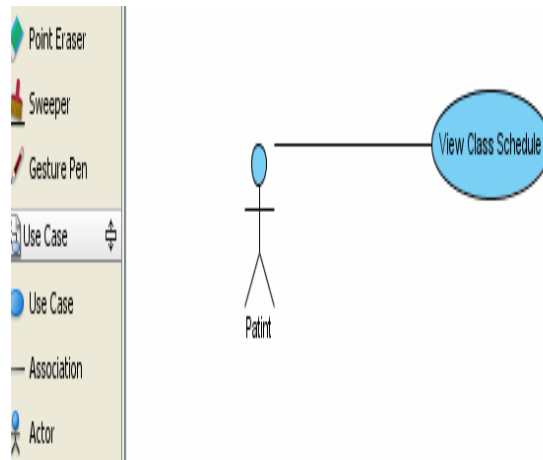
Gambar 2.16

14. Letakan kursor pada simbol Use Case yang ada pada bagian atas simbol actor
15. Tekan mouse pada simbol Use Case dan tarik simbol Use Case pada worksheet hingga sistem menampilkan gambar seperti berikut



Gambar 2.17

16. Ketik nama Use Case sesuai kebutuhan, misalnya **view class schedule**
17. Letakan kursor dan klik pada sembarang tempat diluar simbol Use Case hingga sistem menampilkan Use Case Diagram sebagai berikut



Gambar 2.18

Soal Latihan

1. Buatlah use case diagram untuk menggambarkan sistem detik.com yang mempunyai behaviors (kebiasaan) dan fungsi-fungsi sebagai berikut:
 - a. Sistem menyediakan fungsi untuk meng-upload berita bagi wartawan
 - b. Sistem mempersilahkan pengunjung/guest untuk melihat berita yang ditampilkan oleh sistem dan di-upload oleh wartawan
 - c. Wartawan harus melakukan login terlebih dahulu untuk bisa melakukan upload berita
 - d. Bagi pengunjung/guest yang ingin memberikan komentar terhadap suatu berita yang ada pada sistem wajib melakukan registrasi dan login terlebih dahulu
2. Buatlah use case diagram credit card processing seperti tampak pada gambar enam menggunakan software yang mendukung Unified Modeling Language (misalnya Visual Paradigm atau NetBend)

BAB III

CLASS DIAGRAM

3.1 Definisi Class Diagram

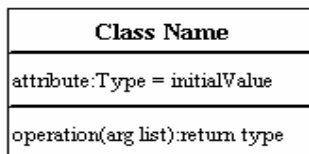
Class adalah kumpulan objek-objek dengan dan yang mempunyai struktur umum, behavior umum, relasi umum, dan semantic/kata yang umum. Class-class ditentukan/ditemukan dengan cara memeriksa objek-objek dalam sequence diagram dan collaboration diagram. Sebuah class digambarkan seperti sebuah bujur sangkar dengan tiga bagian ruangan. Class sebaiknya diberi nama menggunakan kata benda sesuai dengan domain/bagian/kelompoknya (Whitten L. Jeffery et al, 2004).

Class Diagram adalah diagram yang menunjukkan class-class yang ada dari sebuah sistem dan hubungannya secara logika. Class diagram menggambarkan struktur statis dari sebuah sistem. Karena itu class diagram merupakan tulang punggung atau kekuatan dasar dari hampir setiap metode berorientasi objek termasuk UML (Henderi, 2008). Sementara menurut (Whitten L. Jeffery et al 2004:432) class diagram adalah gambar grafis mengenai struktur objek statis dari suatu sistem, menunjukkan class-class objek yang menyusun sebuah sistem dan juga hubungan antara class objek tersebut.

Elemen-elemen class diagram dalam pemodelan UML terdiri dari: Class-class, struktur class, sifat class (class behavior), perkumpulan/gabungan (association), pengumpulan/kesatuan (agregation), ketergantungan (dependency), relasi-relasi turunannya, keberagaman dan indikator navigasi, dan role name (peranan/tugas nama).

3.2 Simbol dan Notasi Dasar Class Diagram

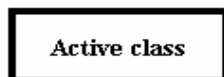
Classes merepresentasikan sebuah abstraksi dari entitas-entitas dengan sifat-sifat atau karakteristik yang bersifat umum. Asosiasi (perkumpulan/persatuan) merepresentasikan relasi (hubungan) antara class-class.



Gambar 3.1

Classes

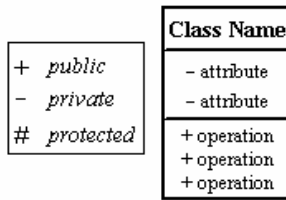
Ilustrasikan/gambarkan classes dengan bentuk empat persegi panjang yang dibagi kedalam ruang-ruang terpisah (compartments). Nama class ditempatkan pada bagian pertama (rata tengah, di-bold, dan Huruf besar), daftar atribut diletakan pada bagian kedua, dan tuliskan operasi-operasi pada class dibagian ketiga (gambar 3.1)



Gambar 3.2

Active Classes

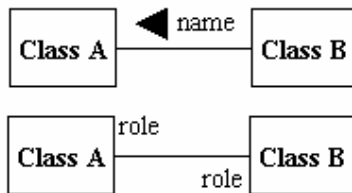
Active classes adalah class yang memulai dan mengontrol aliran/arus/arah aktifitas. Sementara passive class menyimpan data dan melayani (men-serve) class yang lain. Gambarkan active class pada sebuah bolder tebal dibagian tengah (gambar 3.2)



Gambar 3.3

Visibility

Gunakan penilai visibility (“penglihatan”) untuk menandakan siapa yang bisa mengakses informasi-informasi yang diisi kedalam sebuah class. Private visibiliy artinya informasi yang ada dalam sebuah class disembunyikan/dipartisi dari pihak luar. Public visibility mengijinkan semua class yang lainnya untuk melihat nilai informasi. Protected visibility mengijinkan class-class yang ada yang merupakan turunannya untuk mengakses informasi yang ada didalamnya karena mereka merupakan class turunan dari class induknya/inherited (gambar 3.3)



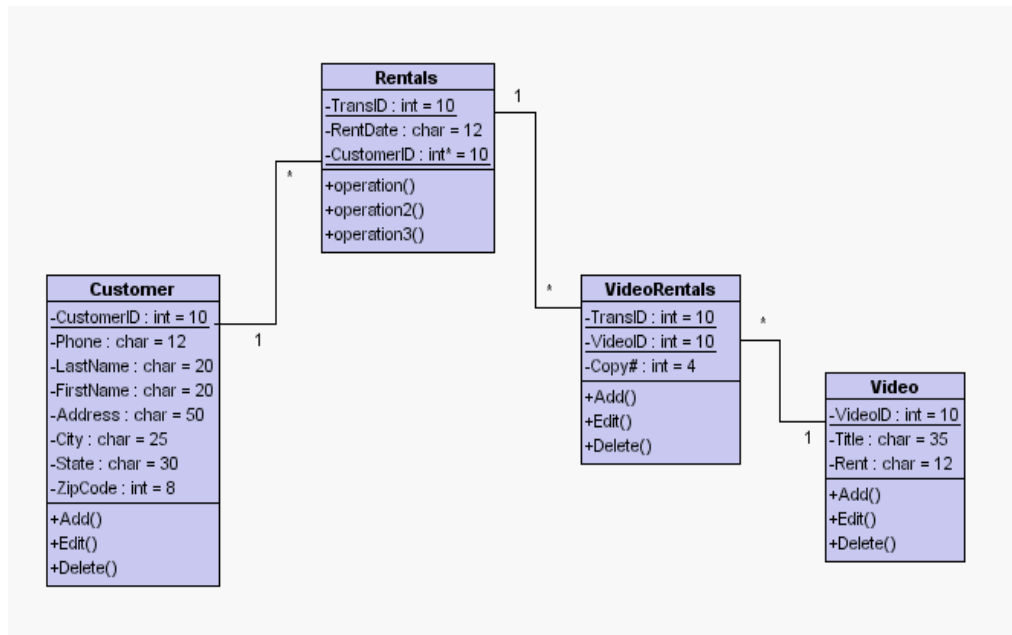
Gambar 3.4

Associations

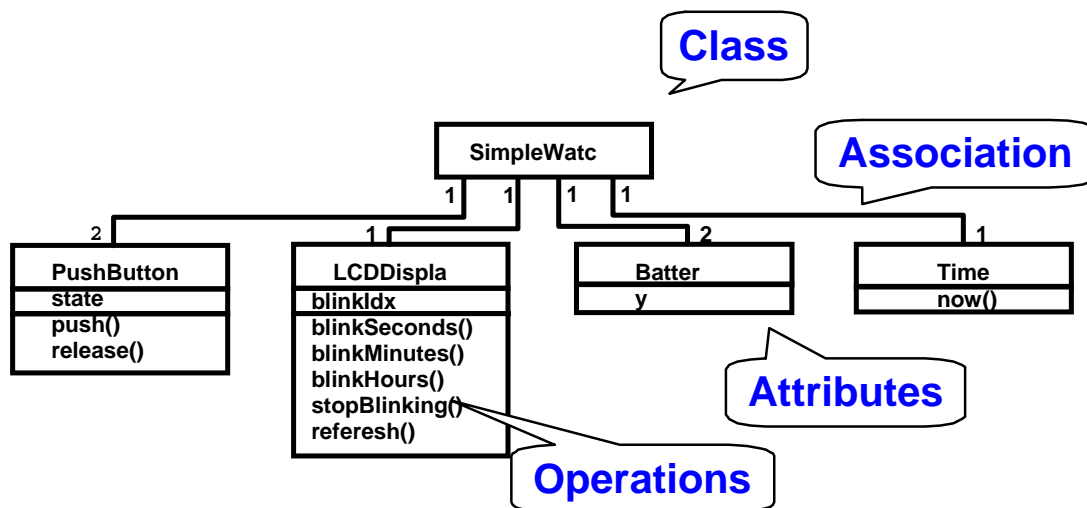
Associations adalah representasi/gambaran relasi statis diantara class-class. Tempatkan nama associations pada bagian atas, di, atau dibawah garis associations. Gunakan tanda anak panah yang berisi sebuah kata yang mengindikasikan relasi secara langsung. Letakan role (aturan/ ketentuan) pada bagian akhir associations. Aturan merepresentasikan arah bagi kedua kelas untuk saling berhubungan satu sama lain (gambar 3.4). Pada umumnya sebuah ”nama” tidak digunakan untuk menunjukan role sebuah class.

Dalam associations terdapat roles (aturan/ketentuan) yang disebut dengan multiplicity. Multiplicity adalah jumlah kejadian minimum dan maksimum dari satu object/class untuk satu kejadian tunggal dari object/class yang terkait (Whetten L. Jeffery et al, 2004:414).

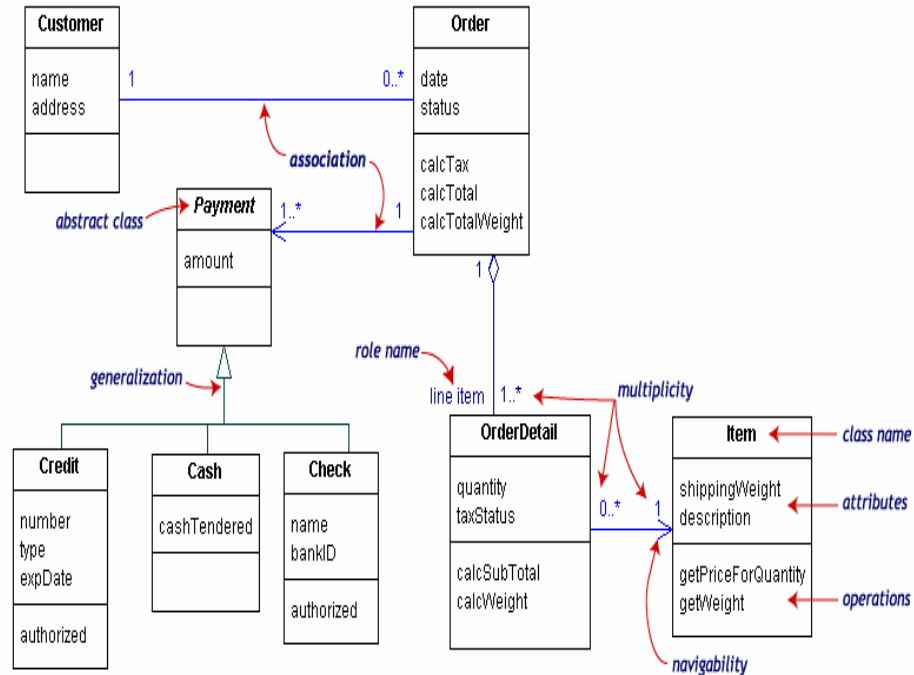
3.3 Contoh Class Diagram:



Gambar 3.5 Class Diagram VideoRental (Henderi, 2007)



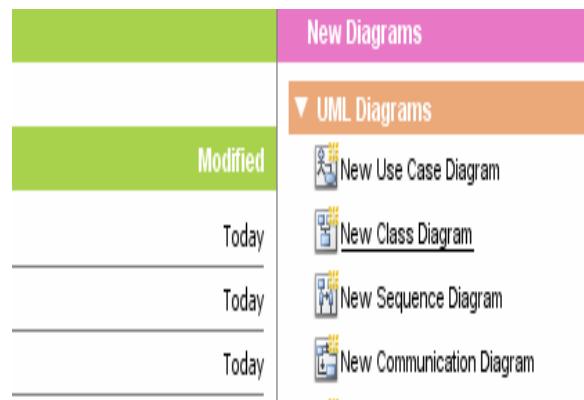
Gambar 3.6 Class Diagram SevicesWatch



Gambar 3.7 Class Diagram Customer Order From Retail (Miller Randy, 2008)

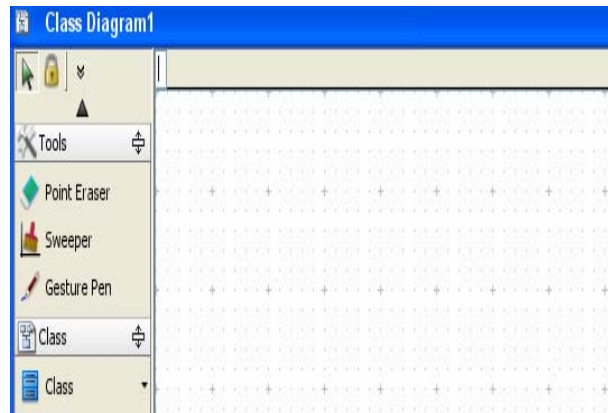
3.4 Membuat Class Diagram dengan Visual Paradigm (VP)

1. Aktifkan software VP hingga muncul dialog layar utama
2. Tempatkan kursor kalimat New Class Diagram yang ada dalam dialog utama seperti pada gambar berikut



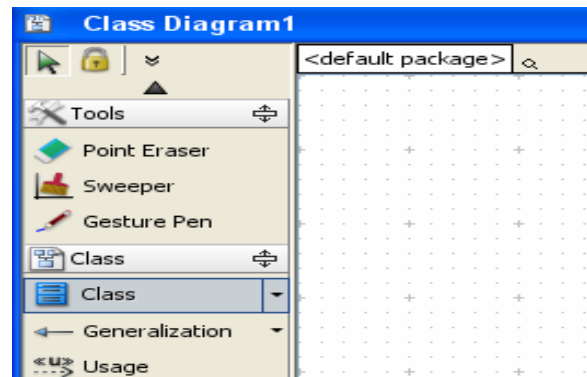
Gambar 3.8

3. Klik pilihan New Class Diagram yang ada layar dialog hingga sistem menampilkan worksheet seperti pada gambar berikut



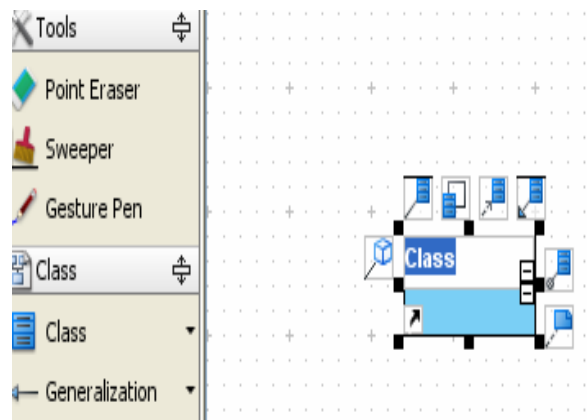
Gambar 3.9

- Tempatkan kursor pada simbol **class** yang ada pada dialog layar tool seperti pada gambar berikut



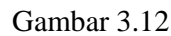
Gambar 3.10

- Klik satu kali simbol **class** dan letakan kursor pada worksheet yang ada disebelah dialog tool dan klik satu kali mouse hingga sistem menampilkan gambar berikut



Gambar 3.11

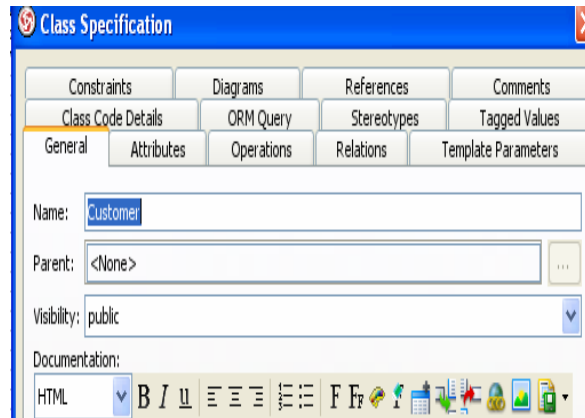
- Tuliskan nama **class** pada dialog yang muncul (rubah nama class pada simbol class) sesuai dengan kebutuhan (misalnya **customer**) dan tekan enter sehingga sistem menampilkan gambar berikut



- Gambar 3.13

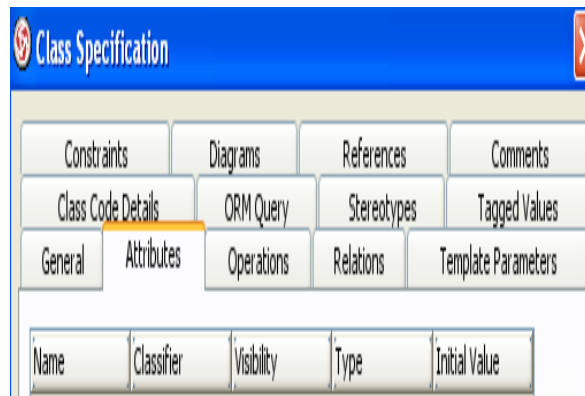
- Gambar 3.14

- Object Oriented Design With Unified Modeling Language (UML)
Oleh: Henderi, M. Kom. (Dosen Matakuliah Perancangan Sistem Informasi STMIK Raharja)
e-mail: henderi@pribadiraharja.com, henderi@yahoo.com, <http://www.freewebs.com/henderi/>



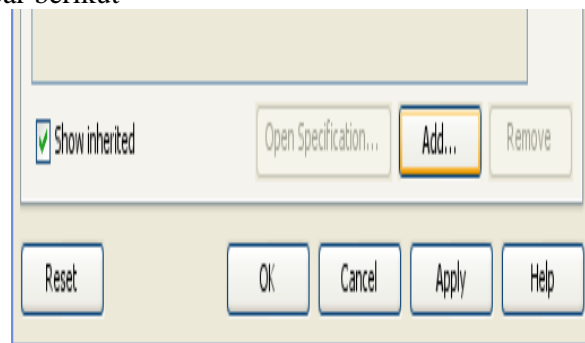
Gambar 3.15

10. Klik kata **Attributes** yang ada pada dialog layar pada sistem untuk melengkapi **class** dengan atribut-atributnya seperti pada gambar berikut



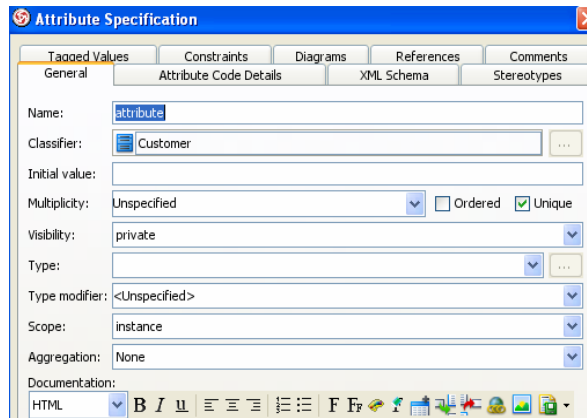
Gambar 3.16

11. Klik fungsi Add yang ada pada bagian bawah layar dialog Class Specification seperti tampak pada gambar berikut



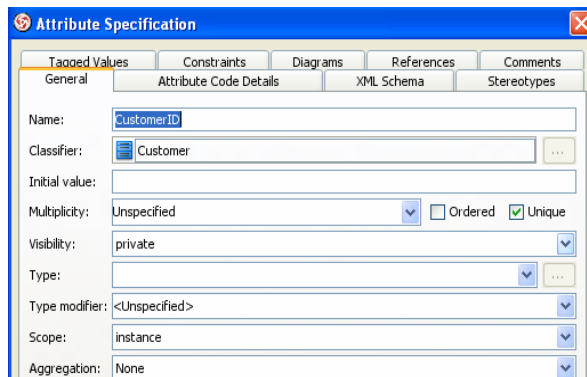
Gambar 3.17

12. Lengkapi Attribute Specification sesuai dengan kebutuhan dengan mengisi/memilih opsional yang ada pada dialog layar berikut



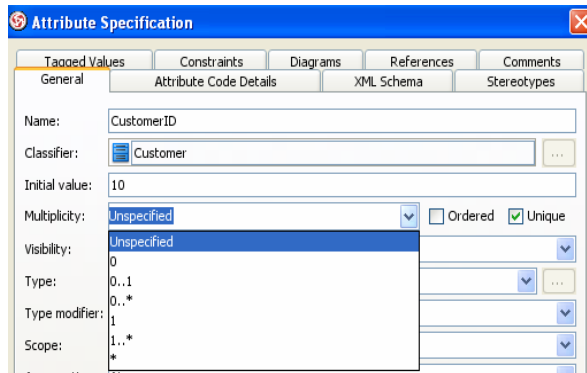
Gambar 3.18

13. Tuliskan CustomerID sebagai atribut pertama class pada bagian Name seperti pada gambar berikut



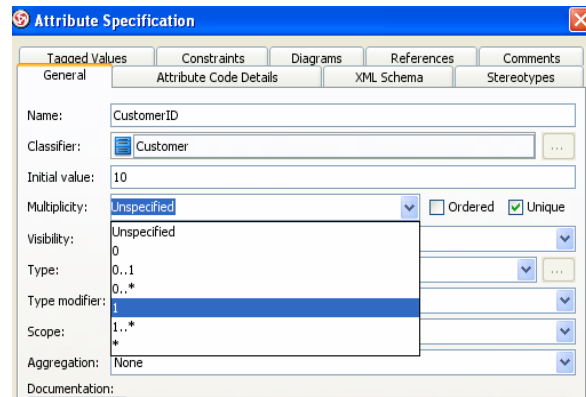
Gambar 3.19

14. Lengkapi panjang atribut CustomerID sesuai kebutuhan pada bagian Initial value seperti pada gambar berikut



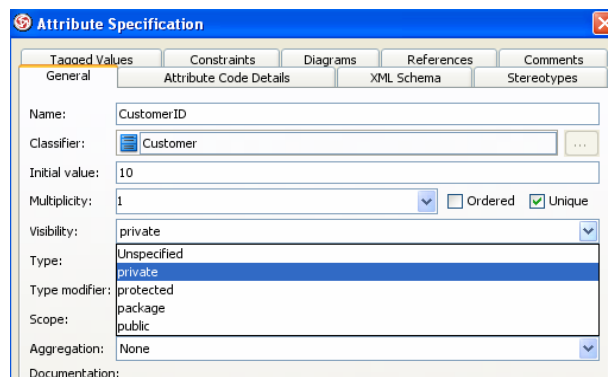
Gambar 3.20

15. Klik list box pada bagian Multiplicity dan pilih jenis multiplicity sesuai kebutuhan (misalnya 1) seperti pada gambar berikut



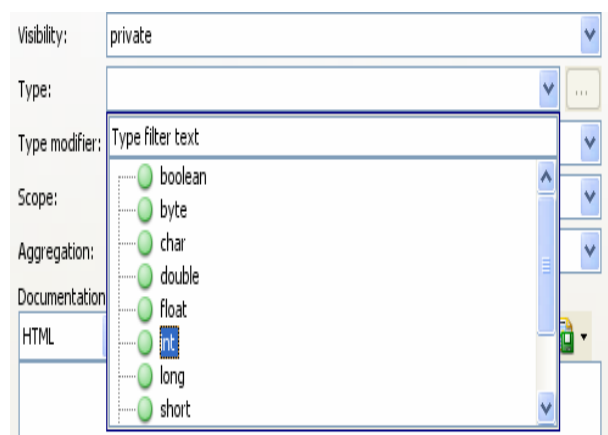
Gambar 3.21

16. Klik list box pada bagian visibility dan pilih kategori visibility sesuai kebutuhan seperti pada gambar berikut



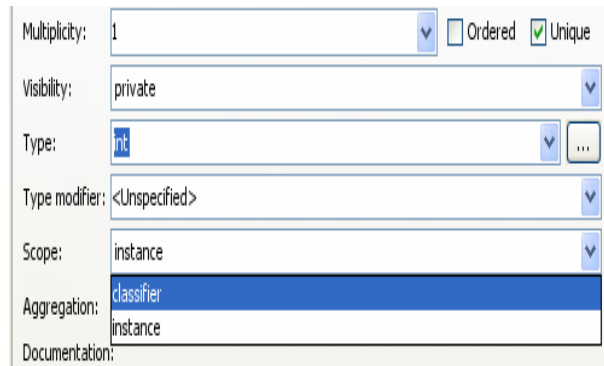
Gambar 3.22

17. Klik list box Type pada layar dialog Attribute Specification hingga tampil dialog layar berikut dan pilih Type atribut sesuai dengan kebutuhan (misalnya integer)



Gambar 3.23

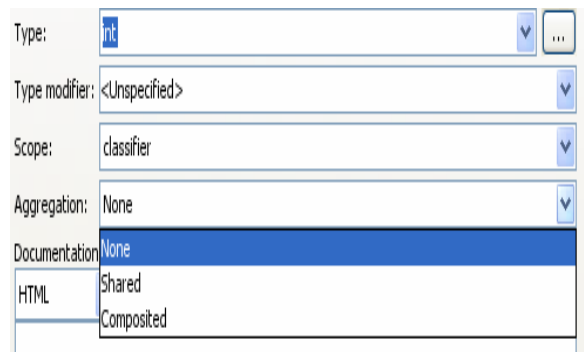
18. Klik list box pada bagian Type modifier dan pilih jenis modifier sesuai dengan kebutuhan, atau biarkan dalam status default yaitu ,Unspecified
19. Klik list box pada bagian Scope untuk memilih jenis atribut sesuai dengan kebutuhan (misalnya classifier karena atribut CustomerID sebagai kunci utama/penentu bagi class Customer) seperti pada gambar berikut



Multiplicity:	1	<input type="checkbox"/> Ordered	<input checked="" type="checkbox"/> Unique
Visibility:	private		
Type:	int		
Type modifier:	<Unspecified>		
Scope:	instance		
Aggregation:	classifier		
Documentation:	instance		

Gambar 3.24

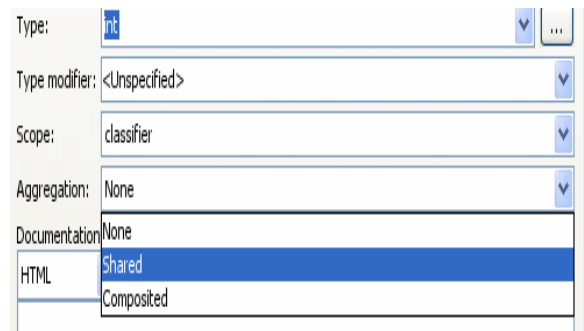
20. Klik list box pada bagian Agregat untuk memilih jenis agregat sesuai kebutuhan (misalnya tetap pada status default yaitu none) seperti pada gambar berikut



Type:	int	
Type modifier:	<Unspecified>	
Scope:	classifier	
Aggregation:	None	
Documentation:	None	
HTML	Shared	
	Composited	

Gambar 3.25

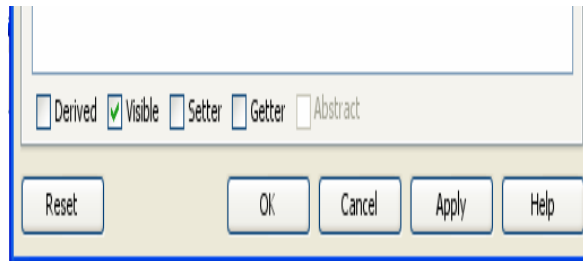
21. Atau pilih **share** pada list box Aggregation karena atribut CustomerID akan di-share kepada class-class yang lain seperti pada gambar berikut



Type:	int	
Type modifier:	<Unspecified>	
Scope:	classifier	
Aggregation:	None	
Documentation:	None	
HTML	Shared	
	Composited	

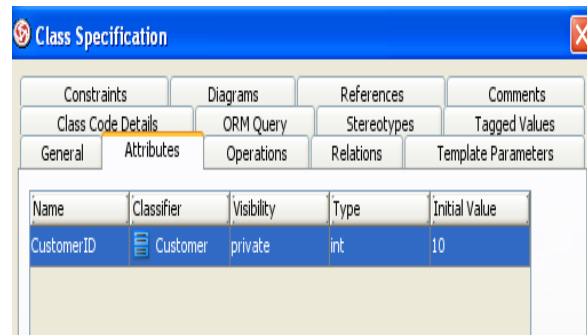
Gambar 3.26

22. Klik fungsi **OK** yang ada pada bagian bawah layar dialog pada sistem seperti pada gambar berikut



Gambar 3.27

23. Sistem akan menampilkan layar dialog attribute specification yang sudah berisi atribut CustomerID seperti pada gambar berikut

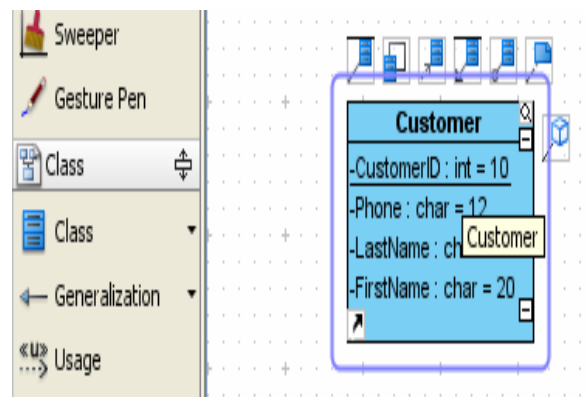


Gambar 3.28

24. Lengkapi class dengan attribut sesuai dengan kebutuhan seperti dan dimulai dengan langkah kesebelas dan seterusnya

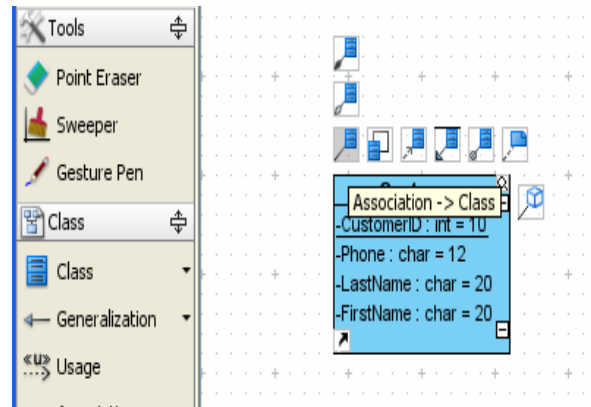
3.5 Membuat Associations (Relasionsip) antar Class

1. Aktifkan simbol class dengan cara menempatkan kursor pada simbol class hingga sistem menampilkan gambar berikut



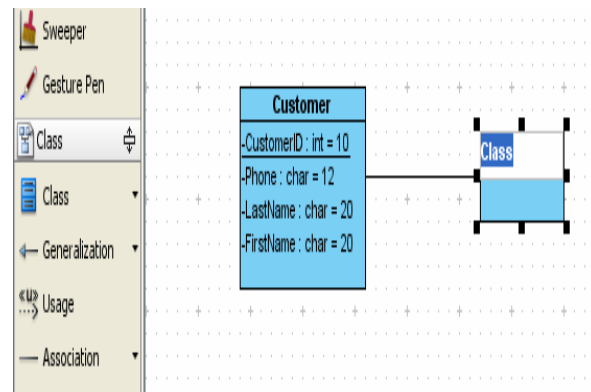
Gambar 3.29

2. Letakan kursor pada simbol Association (relasi) class yang ada pada bagian kiri atas simbol class yang sedang aktif sehingga sistem menampilkan dialog seperti gambar berikut



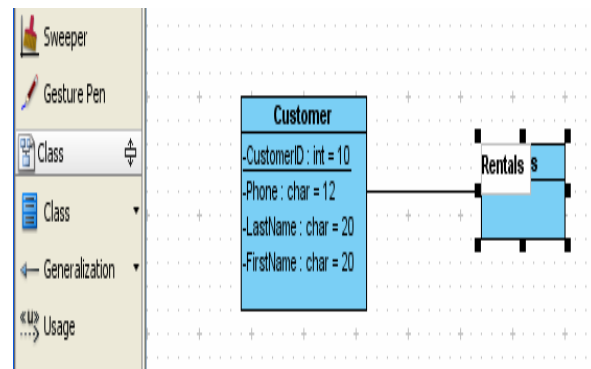
Gambar 3.30

3. Tarik simbol Association class tersebut dan letakan pada tempat sembarang didalam worksheet aktif hingga sistem menampilkan gambar berikut



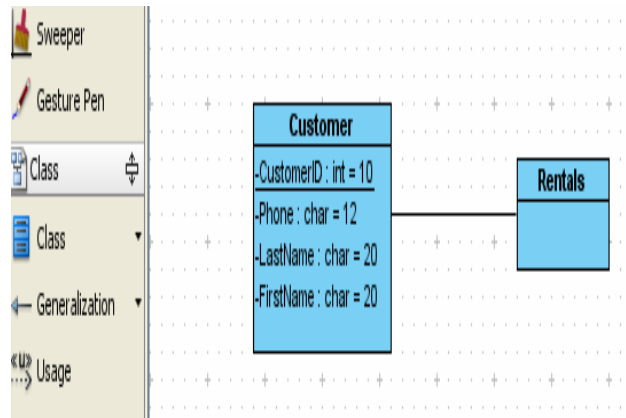
Gambar 3.31

4. Tulisan nama class pada layar dialog class baru yang ditampilkan oleh sistem sesuai kebutuhan (misalnya **Rentals**) seperti pada gambar berikut



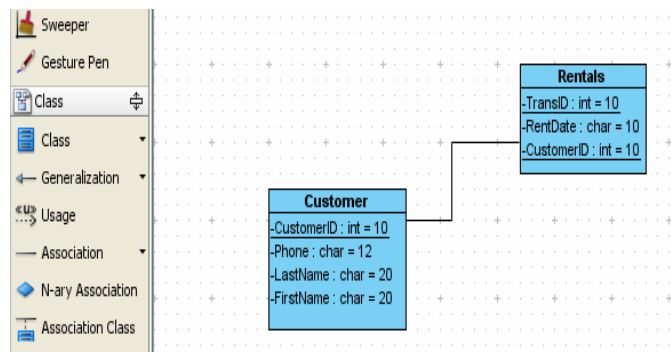
Gambar 3.32

5. Klik disembarang tempat hingga relasi antar class Customer dan Class Rentals ditampilkan sistem sebagai berikut



Gambar 3.33

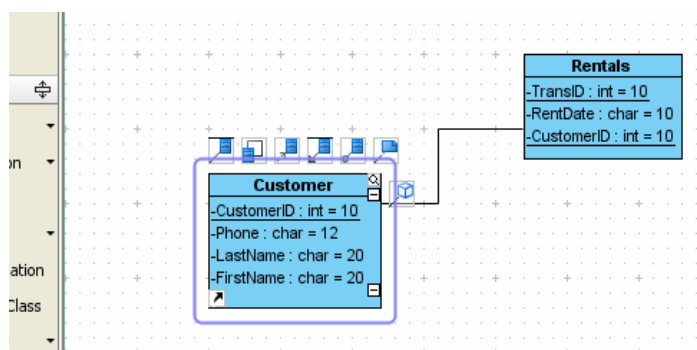
6. Lengkapi nama atribut pada Class Rentals sesuai kebutuhan hingga relasi antar class Customer dan Rentals dan tampilannya tampak seperti pada gambar berikut



Gambar 3.34

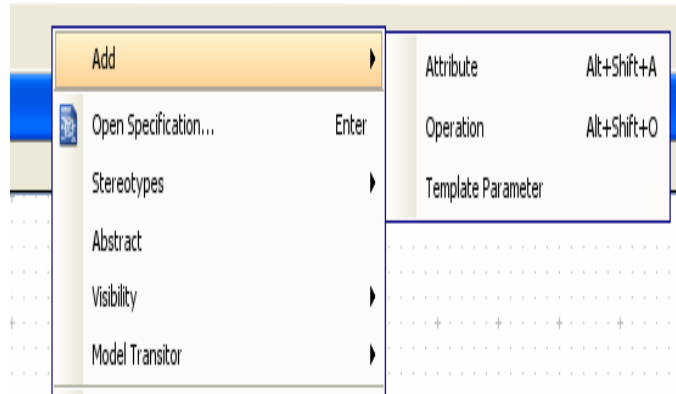
3.6 Melengkapi Class dengan Operations

1. Aktifkan simbol class (misalnya class Customer) dengan cara menempatkan kursor pada simbol class yang ingin dilengkapi hingga sistem menampilkan gambar berikut



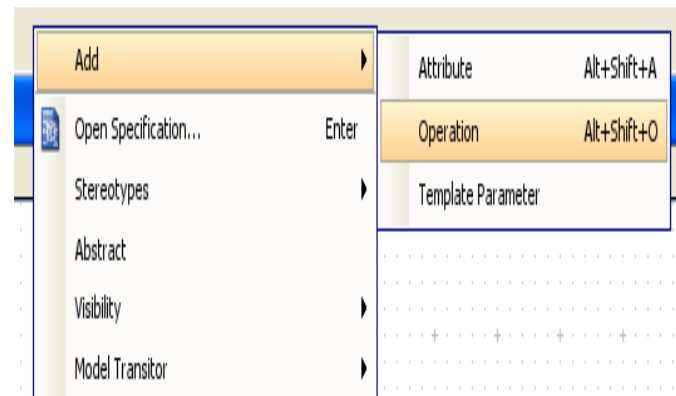
Gambar 3.35

2. Ketika simbol class aktif, klik kanan mouse hingga sistem menampilkan layar dialog sebagai berikut



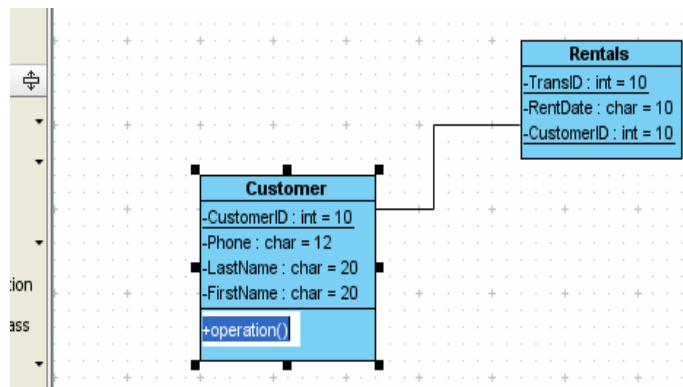
Gambar 3.36

3. Tempatkan kursor pada fungsi **Operation** yang ditampilkan oleh layar dialog sistem seperti pada gambar berikut



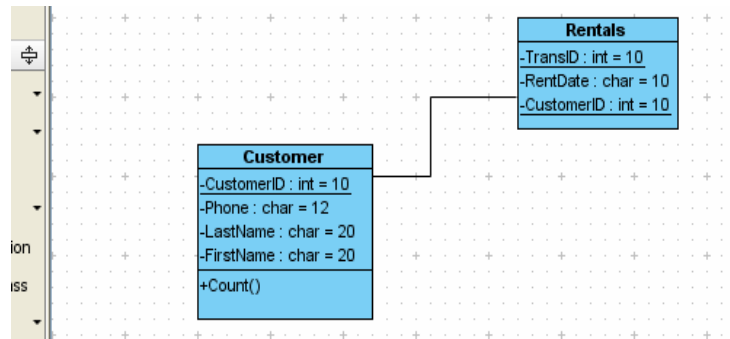
Gambar 3.37

4. Sistem akan menampilkan dialog layar **Operation** seperti pada gambar berikut



Gambar 3.38

5. Ketik nama **Operation** pada layar dialog sesuai kebutuhan (misalnya **count**), dan klik di sembarang tempat pada lembar worksheet sehingga sistem menampilkan Count sebagai operation pada class customer seperti pada gambar berikut

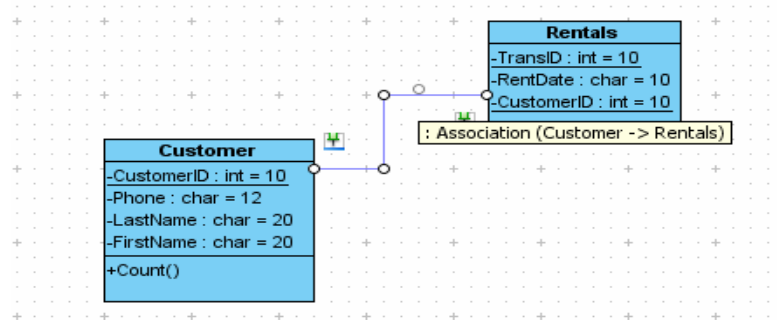


Gambar 3.39

6. Lengkapi class tersebut dengan operations yang lainnya sesuai dengan kebutuhan dengan mengikuti langkah satu sampai 6

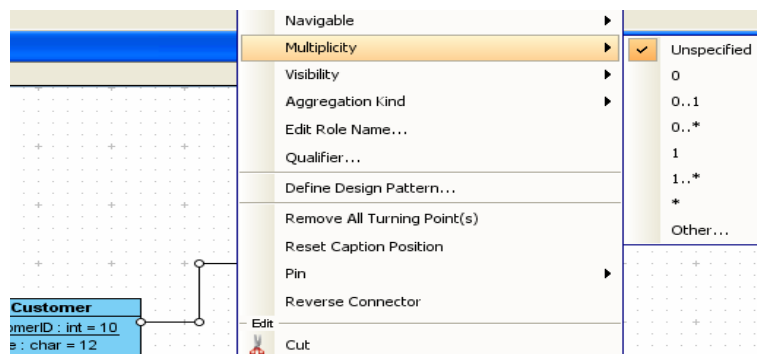
3.7 Melengkapi Class Diagram dengan Multiplicity

1. Tempatkan kursor pada simbol Association hingga sistem menampilkan gambar berikut



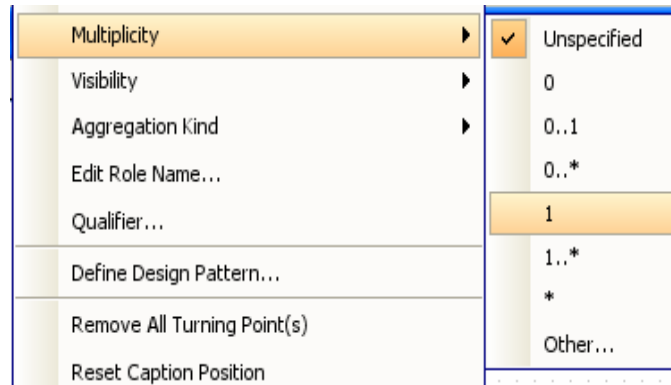
Gambar 3.40

2. Tempatkan kursor dan klik kanan simbol Association tersebut hingga sistem menampilkan dialog layar sebagai berikut



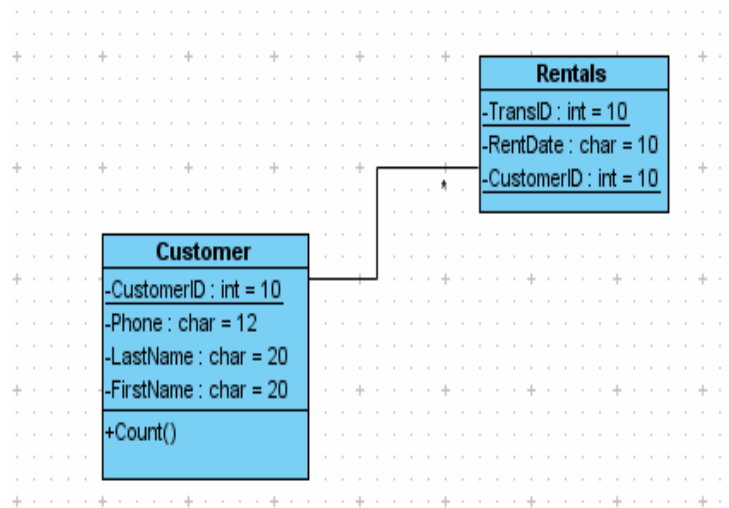
Gambar 3.41

3. Tempatkan kursor pada salah satu jenis Multiplicity yang ada pada dialog layar (misalnya 1) seperti pada gambar berikut



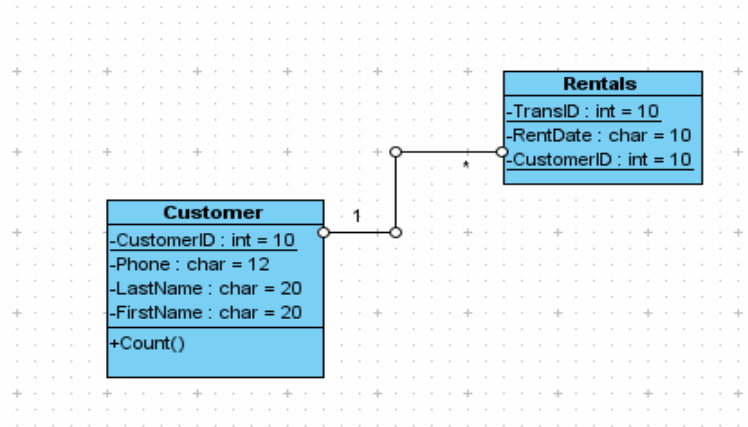
Gambar 3.42

4. Klik jenis multiplicity yang dipilih sesuai kebutuhan hingga sistem menampilkan Association suatu class (dalam hal ini class Rentals) yang sudah dilengkapi dengan Multiplicity seperti pada gambar berikut



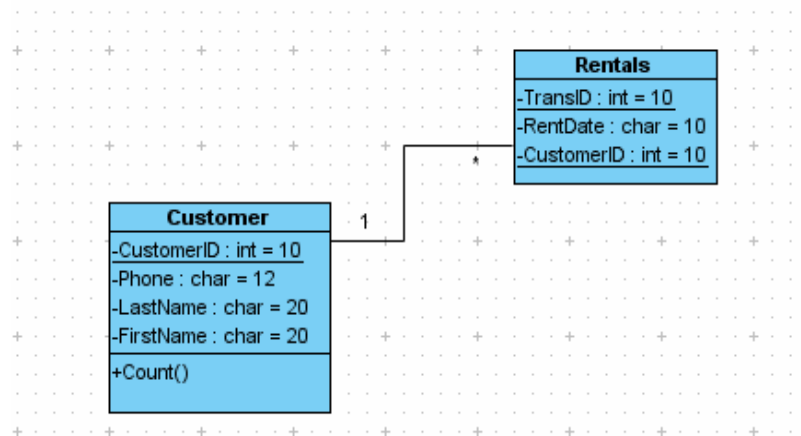
Gambar 3.43

5. Untuk melengkapi Multiplicity pada Association class Customer, letakan kursor pada simbol Association yang dekat dengan simbol class Customer hingga sistem menampilkan gambar langkah kedua.
6. Ikuti langkah ketiga dan keempat untuk melengkapi Multiplicity pada Association class Customer hingga sistem menampilkan gambar berikut



Gambar 3.44

7. Klik sembarang tempat pada worksheet yang aktif sehingga sistem akan menampilkan Class Diagram yang sudah lengkap dengan atribut, association dan multiplicity seperti pada gambar berikut



Gambar 3.45

Soal Latihan

1. Gambarkan class diagram secara lengkap untuk merepresentasikan sistem Student Information Services (SIS) pada Perguruan Tinggi Raharja. Sistem SIS tersebut dapat memenuhi kebutuhan informasi yang dibutuhkan oleh mahasiswa, misalnya: informasi tentang Kartu Studi Mahasiswa, Kartu Hasil Studi Mahasiswa, Transkrip Nilai Akademik Mahasiswa, Jadwal Kuliah, Dosen Pengajar, dan Berbagai Formulir yang dibutuhkan oleh mahasiswa dalam mengikuti dan melaksanakan kegiatan belajar mengajar di Perguruan Tinggi Raharja.

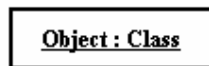
BAB IV

SEQUENCE DIAGRAM

4.1 Definisi Sequence Diagram

Sequence Diagram adalah suatu diagram yang memperlihatkan/menampilkan interaksi-interaksi antar objek di dalam sistem yang disusun pada sebuah urutan atau rangkaian waktu. Interaksi antar objek tersebut termasuk pengguna, display, dan sebagainya berupa "pesan/message" (Henderi, 2008). Sequence diagram dapat juga didefinisikan sebagai suatu diagram yang menggambarkan interaksi-interaksi yang ada antar class dalam suatu hubungan perubahan dengan sebuah pesan pada akhir waktu (Miller Randy, 2008). Sequence Diagram digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai sebuah respon dari suatu kejadian/event untuk menghasilkan output tertentu. Sequence Diagram diawali dari apa yang me-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan (Henderi, 2007).

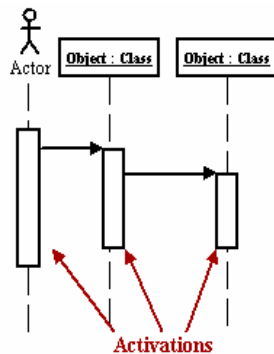
4.2 Simbol dan Notasi Dasar Sequence Diagram



Gambar 4.1

Class roles

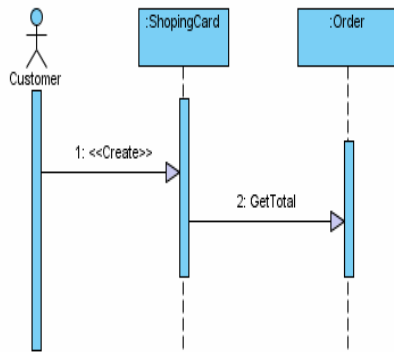
Class roles menggambarkan way (jalan) sebuah objek akan menunjukkan reaksi (berkelakuan) dalam sebuah keadaan (konteks). Gunakan simbol object UML untuk menggambarkan aturan-aturan class, tetapi bukan daftar atribut-atribut objek (gambar 4.1)



Gambar 4.2

Activation

Activation boxes merepresentasikan waktu yang dibutuhkan oleh sebuah objek untuk melaksanakan sebuah tugas/perintah secara lengkap (gambar 4.2)



Gambar 4.3

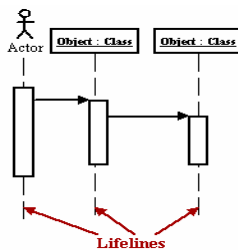
Messages

Message adalah anak panah yang merepresentasikan komunikasi antara objek. Message berguna untuk mengirimkan perintah kepada LifeLine. Message bisa berasal dari actor kepada LifeLine atau dari LifeLine kepada LifeLine yang lain). Gunakan setengah garis anak panah untuk merepresentasikan pesan-pesan asynchronous. Pesan asynchronous dikirim dari sebuah objek yang tidak akan menunggu respon dari penerima sebelum melanjutkan perintahnya (gambar 4.3)

Arrow	Message type
	Simple
	Synchronous
	Asynchronous
	Balking
	Time out

Gambar 4.5

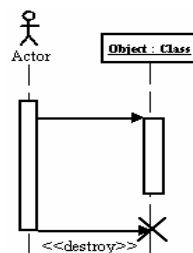
Berbagai tipe message untuk sequence diagram dan collaboration diagram (gambar 4.5)



Gambar 4.6

Lifelines

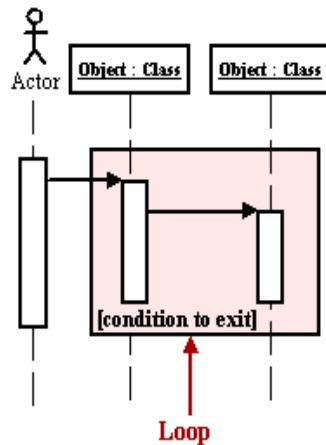
Lifeline adalah tanda garis pisah yang mengindikasikan kehadiran objek pada saat terakhir/akhir waktu (gambar 4.6)



Gambar 4.7

Destroying Objects

Objek dapat di-akhiri secara lebih cepat menggunakan sebuah garis anak panah yang diberi label “<<destroy>>” dan diujungnya diberi sebuah label X (gambar 4.7)



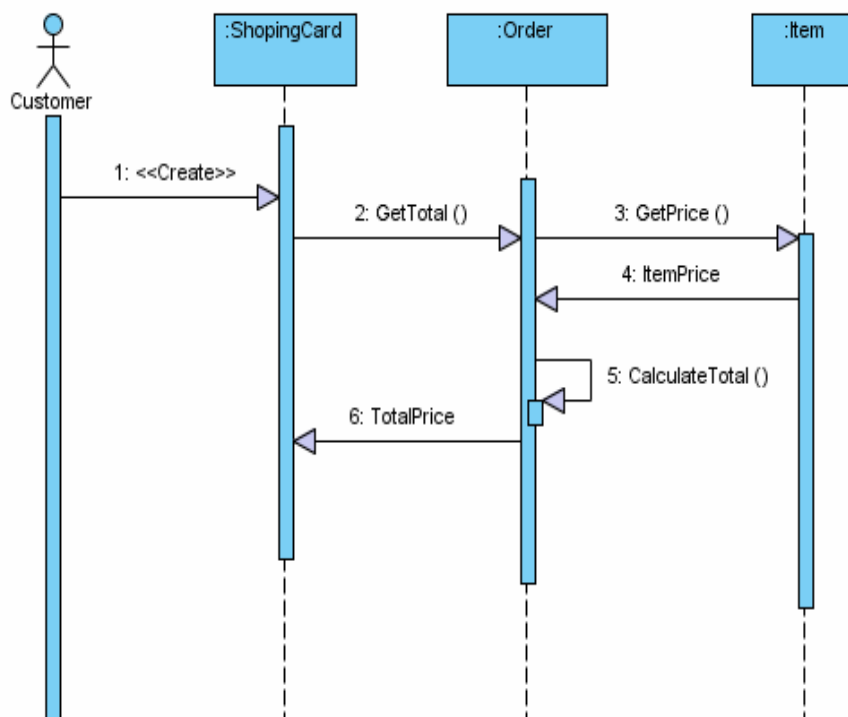
Gambar 4.8

Loops

Suatu pengulangan atau loop dalam sebuah sequence diagram digambarkan seperti sebuah empat persegi panjang. Letakan kondisi untuk keluar pada pengulangan di bagian kiri bawah di dalam dalam kurang empat persegi panjang (gambar 4.8)

4.3 Contoh Sequence Diagram:

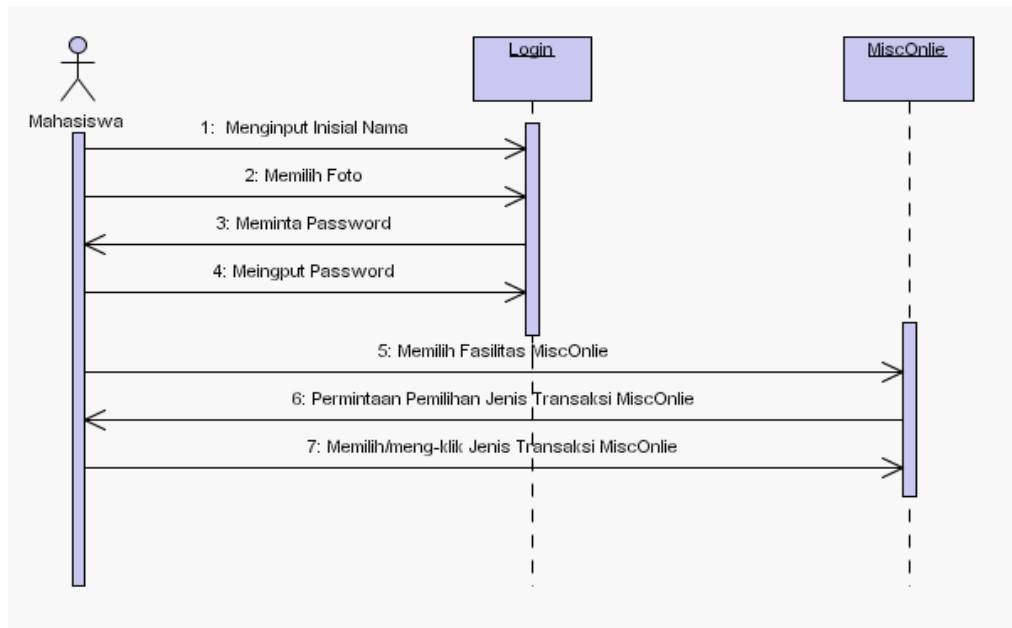
Berikut ini adalah contoh sequence diagram shopping card (berbelanja menggunakan kartu belanja).



Gambar 4.9 Sequence Diagram Shopping Card

Berdasarkan gambar delapan diatas diketahui bahwa sequence diagram shopping card melibatkan actor customer dan tiga object class yaitu shopping card, order dan item. Sequence diagram ini diawali dengan kegiatan Actor mengirimkan perintah (message) kepada LifeLine Shopping Card, dan selanjutnya LifeLine mengirimkan message kepada LifeLine yang lainnya, termasuk diantaranya ada satu LifeLine yang ditujukan kepada

LifeLine itu sendiri (message to selft) yang terjadi pada LifeLine Order. Sequence diagram shopping card menggambarkan enam interaksi yang terjadi antar objek.

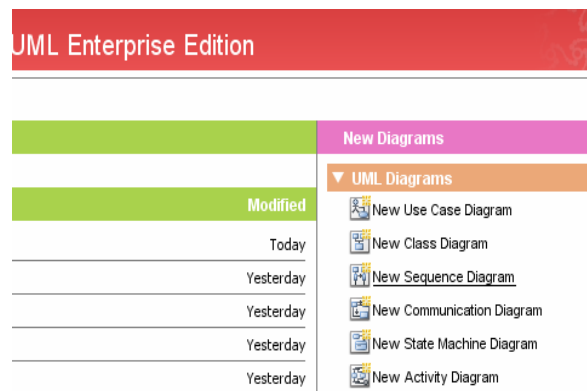


Gambar 4.10 Sequence Diagram MisOnline Pada SIS Raharja (Henderi et al, 2008)

4.4. Membuat Sequence Diagram dengan Visual Paradigm (VP)

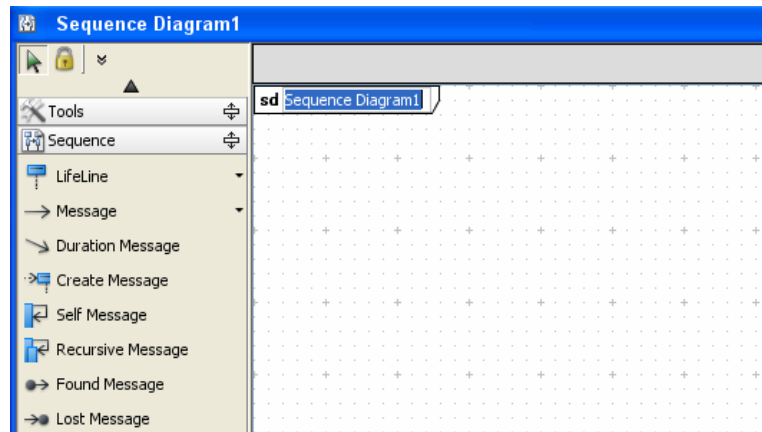
4.4.1 Membuat Actor dan Object pada Sequence Diagram

1. Aktifkan software Visual Paradigm hingga menampilkan layar dialog utama berikut



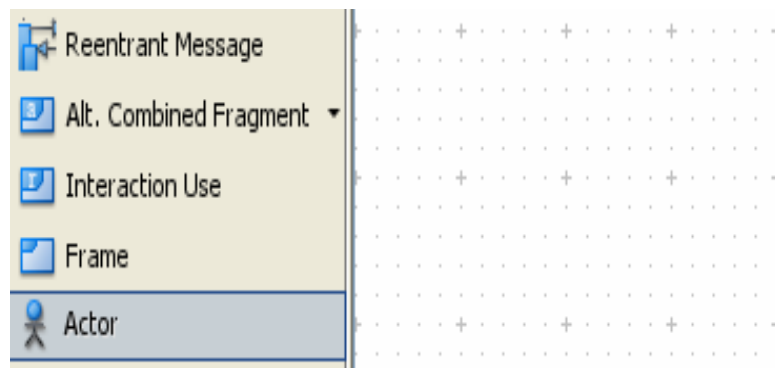
Gambar 4.11

2. Klik pilihan **New Sequence Diagram** pada dialog utama UML diagram hingga sistem menampilkan gambar worksheet berikut



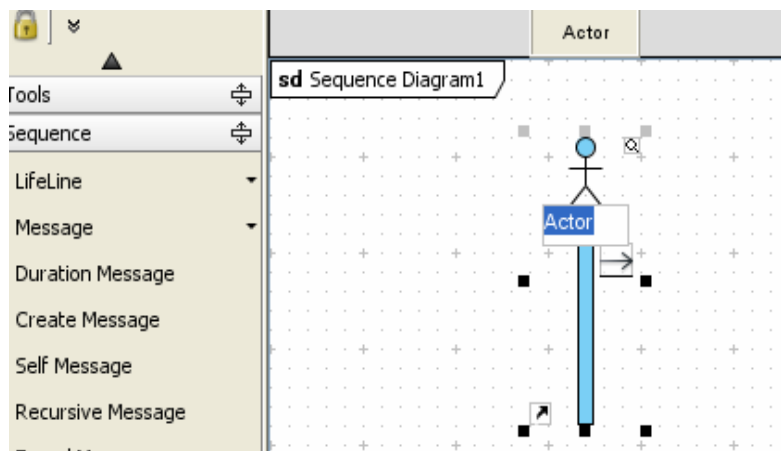
Gambar 4.12

3. Tempatkan kursor pada simbol actor yang pada pada daftar simbol layar dialog Sequence seperti pada gambar berikut



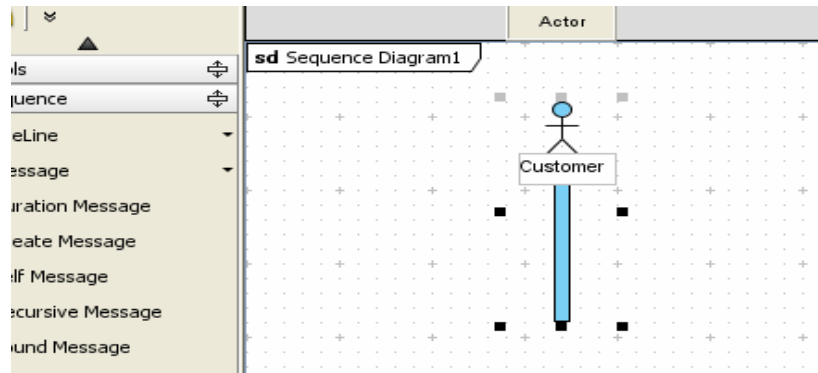
Gambar 4.13

4. Tempatkan kursor pada lembar worksheet dan klik satu kali hingga sistem menampilkan simbol actor pada worksheet seperti gambar berikut



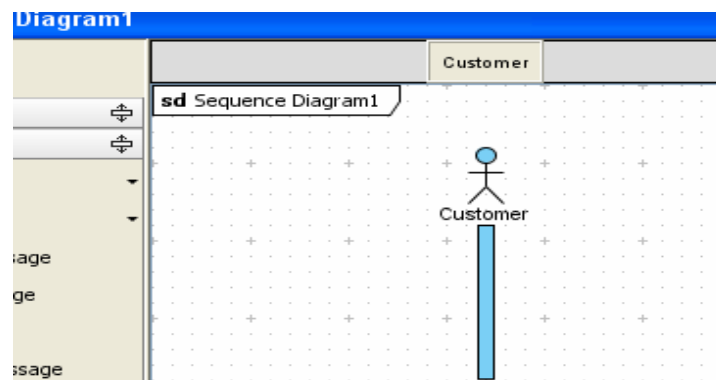
Gambar 4.14

5. Ketikkan nama actor sesuai kebutuhan (misalnya Customer) pada dialog layar actor hingga nama actor menjadi Customer seperti tampak pada gambar berikut



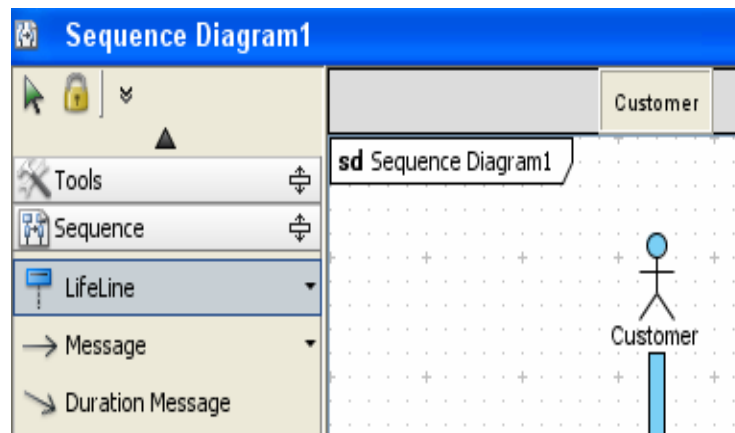
Gambar 4.15

6. Tempatkan kursor dan klik mouse disembarang tempat pada worksheet hingga simbol actor pada sequence diagram tampak seperti pada gambar berikut



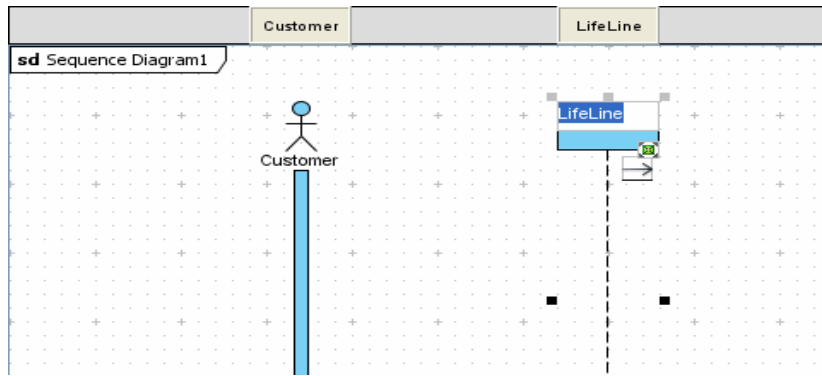
Gambar 4.16

7. Klik simbol LifeLine yang ada pada list simbol Sequence seperti pada gambar



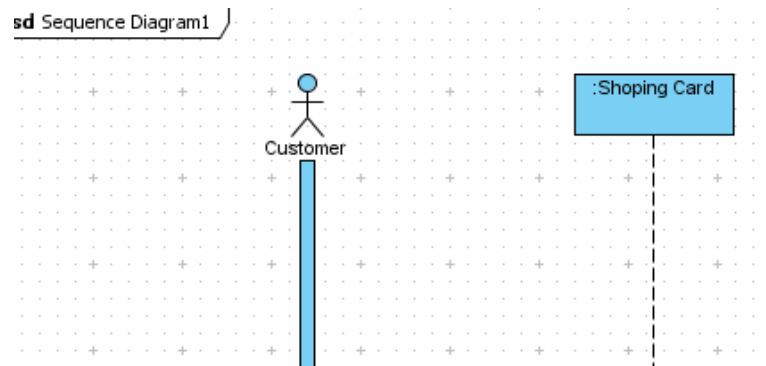
Gambar 4.17

8. Tempatkan kursor di sebelah kanan simbol actor dan klik mouse satu kali hingga sistem menampilkan gambar sebagai berikut



Gambar 4.18

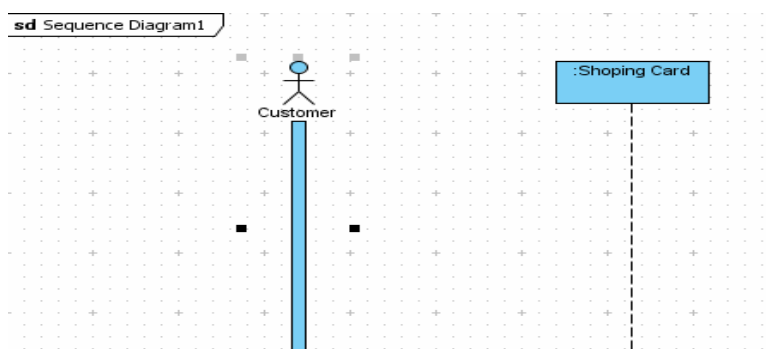
9. Ketikkan nama Object atau nama LifeLine sesuai kebutuhan pada bagian atas simbol LifeLine (misalnya Shopping Card) dan klik mouse satu kali pada worksheet hingga sistem menampilkan gambar berikut



Gambar 4.19

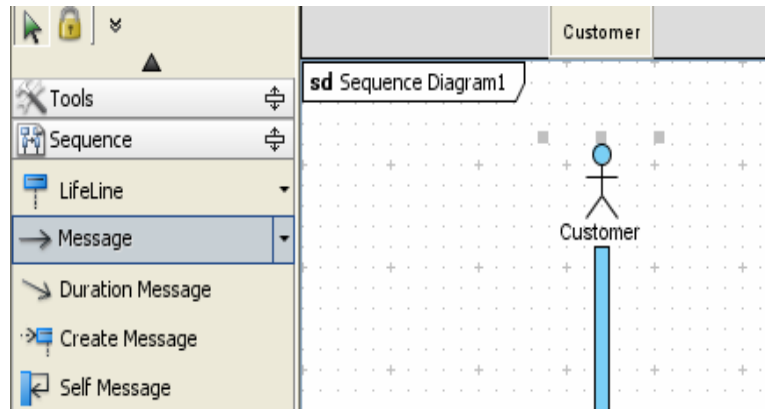
4.4.2 Membuat Message

1. Klik simbol actor pada sequence diagram satu kali hingga sistem menampilkan gambar berikut



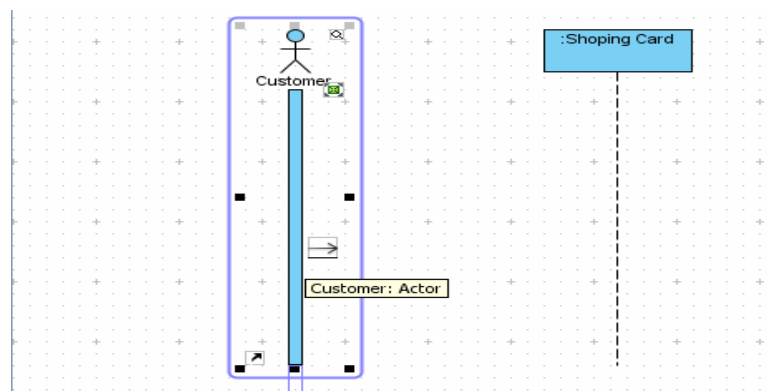
Gambar 4.20

2. Klik simbol message yang pada list simbol sequence seperti pada gambar berikut



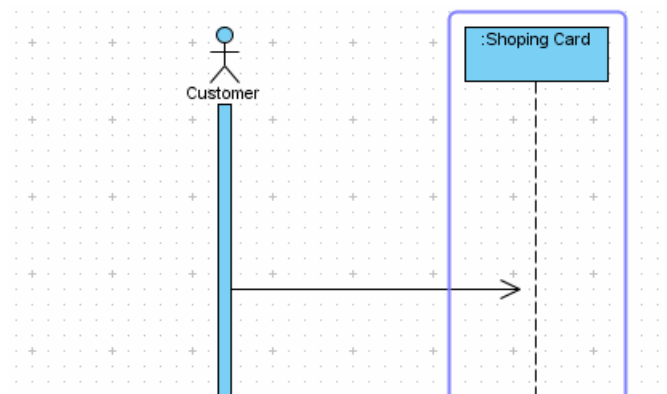
Gambar 4.21

3. Letakan kursor pada LifeLine simbol actor hingga sistem menampilkan gambar berikut



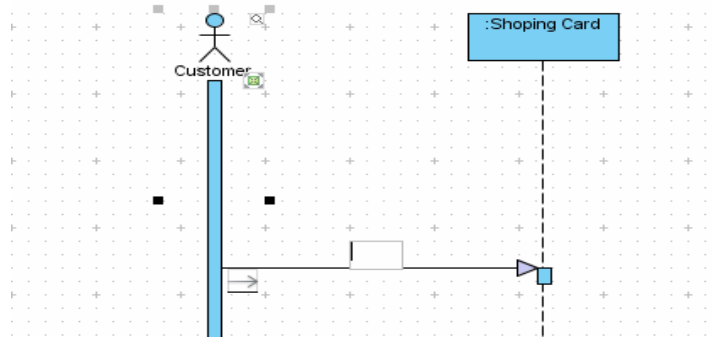
Gambar 4.22

4. Drag/tarik kursor sampai pada daerah LifeLine/Object Shopping Card hingga sistem menampilkan gambar berikut



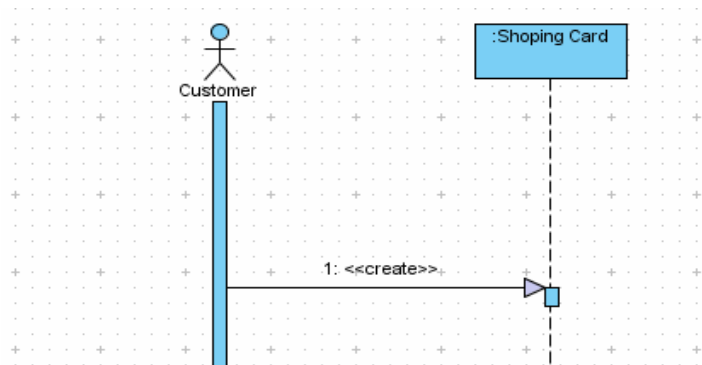
Gambar 4.23

5. Bebaskan mouse hingga sistem menampilkan gambar berikut



Gambar 4.24

6. Ketikkan message untuk sequence diagram pada dialog arrow message sesuai dengan kebutuhan (misalnya <<create>>), dan klik sembarang tempat pada worksheet hingga sistem menampilkan gambar berikut



Gambar 4.25

7. Lengkapi/sempurnakan Sequence Diagram sesuai dengan kebutuhan dengan mengulangi langkah nomor tujuh sampai lima belas

Soal Latihan:

Buatlah sequence diagram untuk menggambarkan/menampilkan interaksi-interaksi antara object *an order entry windows*, *an order*, dan *a delivery item*. Skenario diawali dengan adanya interaksi atau perintah dari *an order entry window* sebagai persiapan kepada object *an order*. Selanjutnya diteruskan dengan adanya message berupa display stock dalam keadaan kosong (out stock) kepada *an order entry window*, dan dilanjutkan dengan message *in stock* kepada object atau LifeLine *a delivery item* dalam sequence diagram ini interaksi hanya terjadi antara LifeLine.

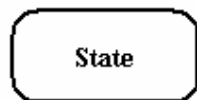
BAB V

STATECHART DIAGRAM

5.1 Definisi Statechart Diagrams

Statechart diagram adalah diagram yang menggambarkan perilaku (behavior) dinamis sistem dalam merespon stimulasi atau aksi yang berada dari luar (Henderi, 2007). State diagram juga menggambarkan perilaku class dalam merespon pemicu/ stimulasi yang berasal dari luar (Miller Randy, 2008). Model diagram ini merupakan diagram alir kontrol dinamis dari kejadian satu kepada kejadian yang lainnya dalam sebuah sistem (Henderi et al, 2008). Karena itu, statechart diagram terutama berguna dalam pemodelan berorientasi reaksi objek yang keadaannya dipicu oleh kejadian-kejadian/peristiwa khusus.

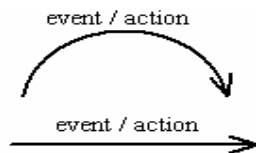
5.2 Simbol dan Notasi Dasar Statechart Diagram



Gambar 5.1

States

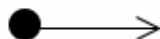
States merepresentasikan keadaan “life” sebuah objek selama objek tersebut ada. State digambarkan dalam bentuk empat persegi panjang yang keempat sudutnya berbentuk bundar (gambar 5.1).



Gambar 5.2

Transition

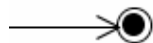
Suatu anak panah yang tebal merepresentasikan jalan antara state yang berlainan dari sebuah objek. Berikan nama transisi (peralihan) dengan kejadian yang menjadi pemicunya dan aksi yang dihasilkan olehnya (gambar 5.2)



Gambar 5.3

Initial Pseudo State

Initial Pseudo State digunakan untuk memulai Statechart Diagram. Inisital Pseudo State digambar menggunakan *arrow* yang pada bagian awalnya terdapat lingkaran berisi penuh bold (gambar 5.3)

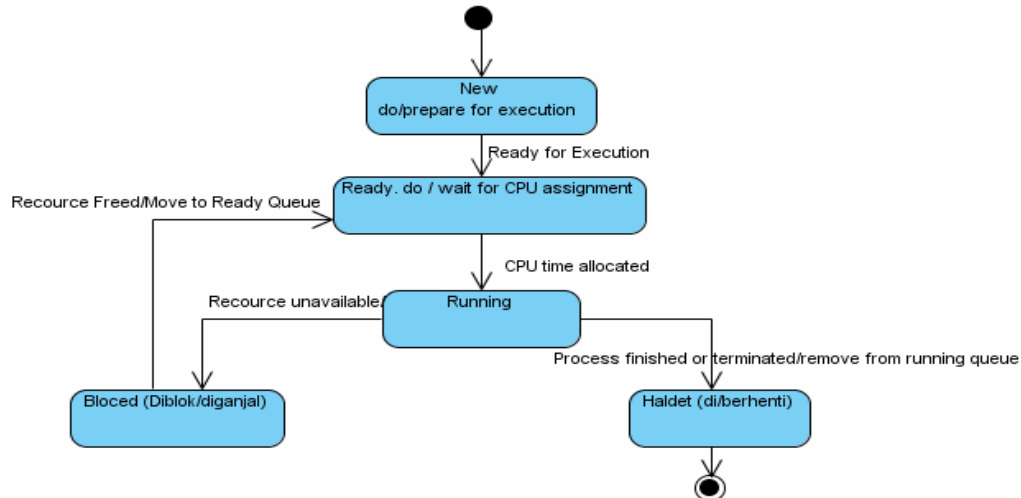


Gambar 5.4

Final State

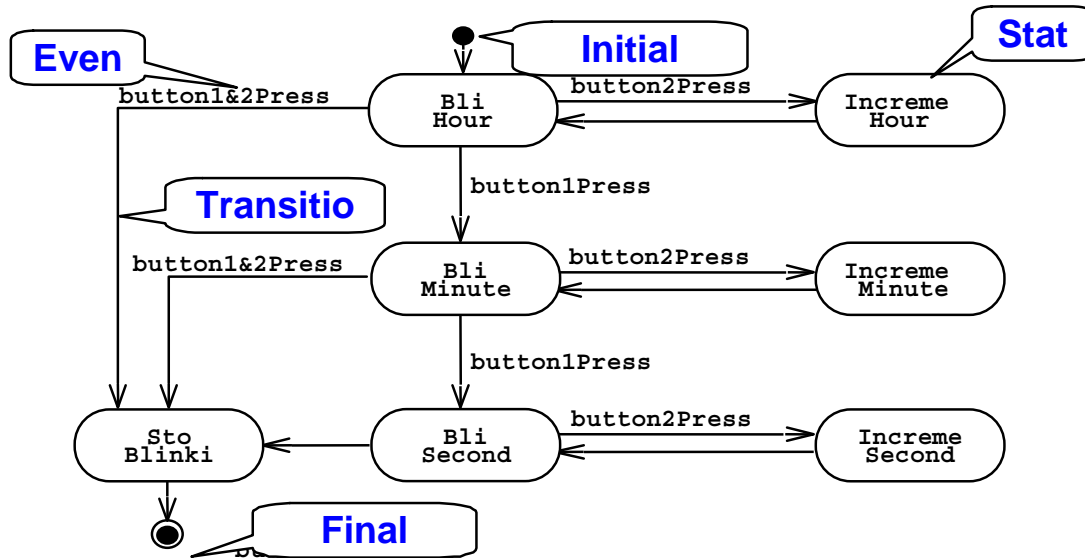
Final State digunakan untuk mengakhiri diagram statechart. Final State digambar menggunakan *arrow* yang diujungnya terdapat lingkaran yang berisi lingkaran berisi penuh bold (gambar 5.4)

5.3 Contoh Statechart Diagram



Gambar 5.5 Statechart Diagram CPU Execution

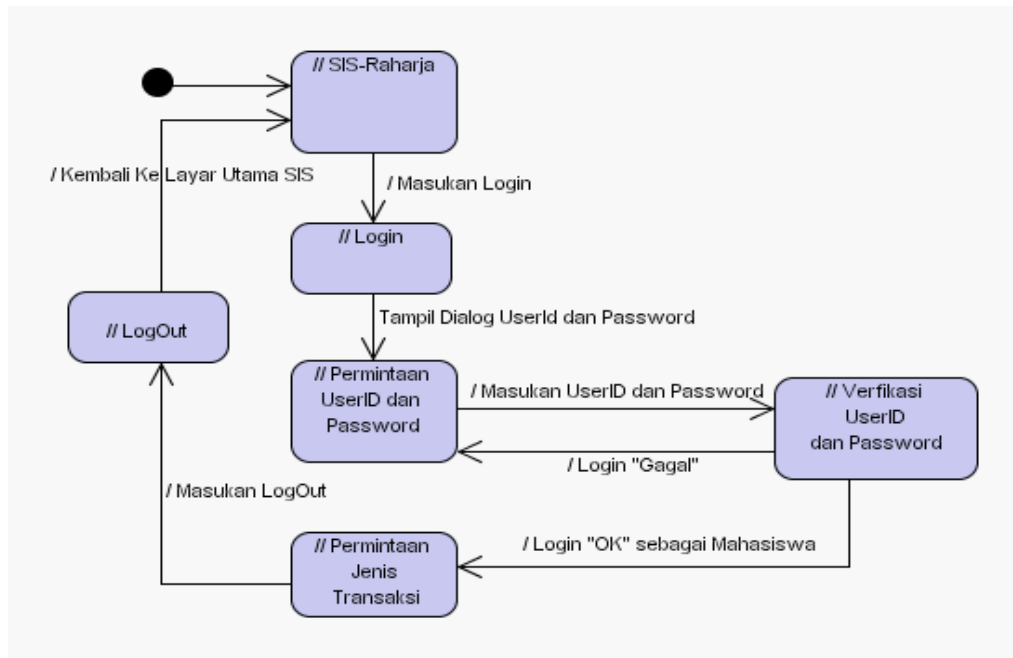
Statechart diagram order pada gambar lima di atas merupakan diagram alir kontrol dinamis dari kejadian satu kepada kejadian yang lainnya dalam suatu sistem CPU Execution. Action/event antara state dibuat untuk menggambarkan jalan antara state yang berlainan dari sebuah objek. Untuk memperjelas dan memperdalam pemahaman, berikut contoh statechart diagram yang lain.



Gambar 5.6 Statechart Diagram Time Setting Within a Watch

Statechart diagram pada gambar 5.6 di atas merupakan diagram alir kontrol dinamis dari kejadian satu kepada kejadian yang lainnya dalam sebuah sistem. Statechart diagram di atas menggambarkan perilaku (behavior) dinamis sistem dalam merespon stimulasi atau aksi yang berada dari luar. Model diagram pada gambar ini juga merupakan diagram alir

kontrol dinamis dari kejadian satu kepada kejadian yang lainnya dalam sebuah sistem pengaturan waktu pada suatu jam.

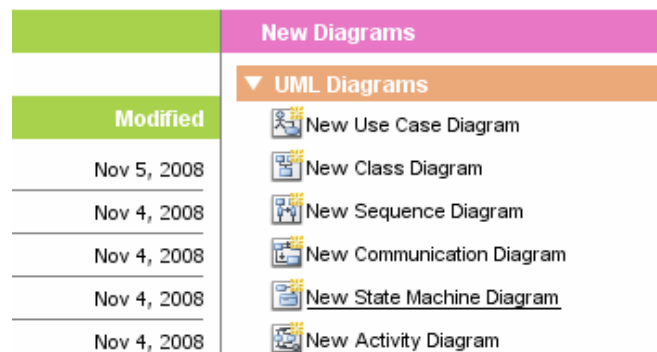


Gambar 5.7 Statechart Diagram LogOut SIS PT Raharja (Henderi et al, 2008)

5.4 Membuat Statechart Diagram dengan Visual Paradigm (VP)

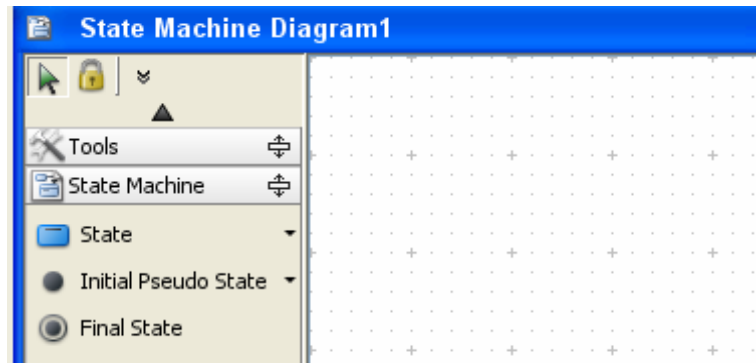
5.4.1 Membuat Initial Pseudo State (Awal Sate) dan State Pertama

1. Arahkan kursor kepada fungsi New State Machine Diagram yang pada pada layar dialog UML Diagram yang ada pada dialog utama software VP seperti pada gambar berikut.



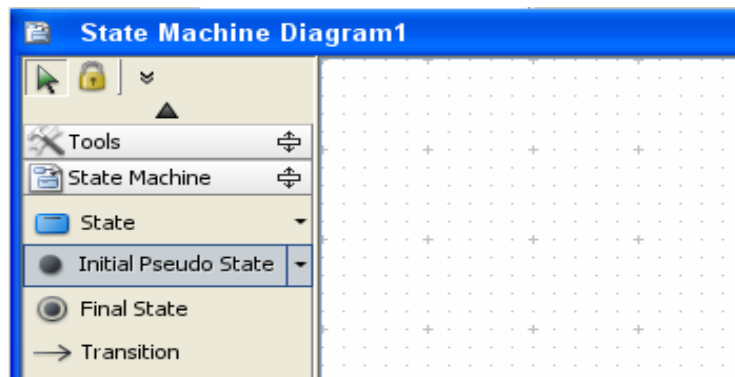
Gambar 5.8

2. Klik pilihan New State Machine Diagram yang ada pada dialog menu UML Diagram tersebut hingga sistem menampilkan worksheet Statechart diagram seperti gambar berikut (dalam VP disebut dengan State Machine Diagram)



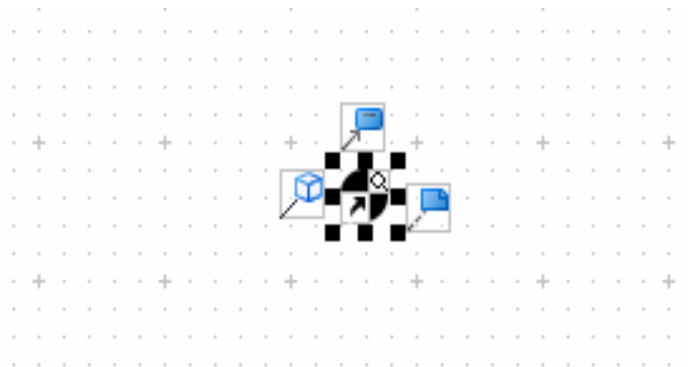
Gambar 5.9

3. Arahkan kursor dan klik simbol fungsi Initial Pseudo State yang ada pada dialog layar Tools State Machine seperti tampak pada gambar



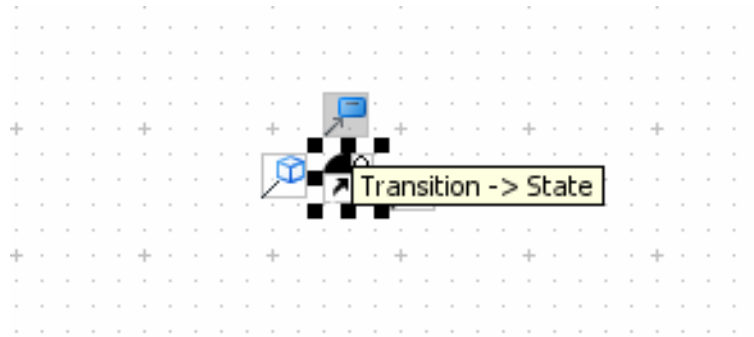
Gamabr 5.10

4. Tempatkan kursor pada worksheet yang aktif dan klik mouse satu kali hingga sistem akan menampilkan gambar Initial Pseudo State berikut (sebagai simbol dimulainya state)



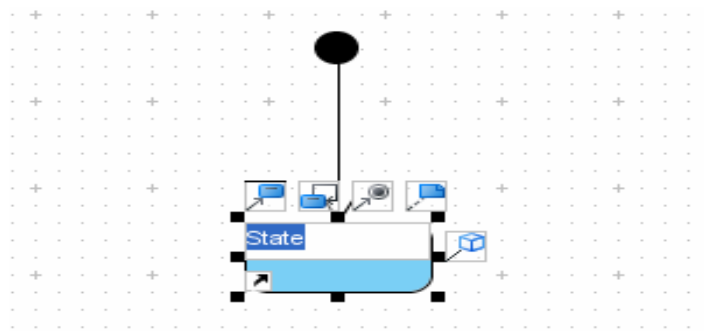
Gambar 6.11

5. Arahkan kursor pada simbol state yang ada di atas simbol Initial Pseudo State sehingga sistem menampilkan gambar berikut



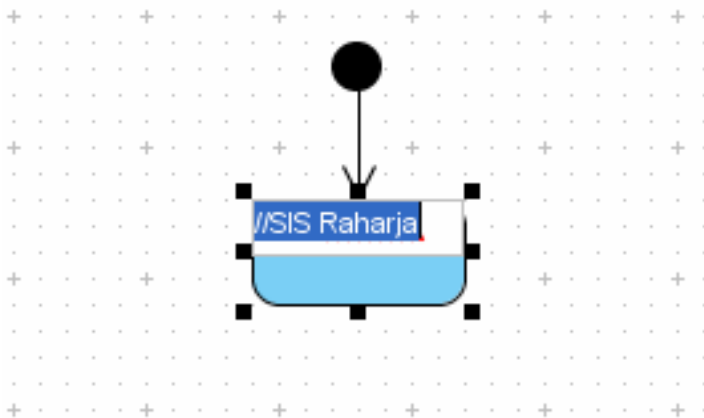
Gamabr 6.12

6. Tekan dan drag/tarik simbol state tersebut ke bawah (atau sesuai keinginan) dan bebaskan mouse sehingga sistem menampilkan gambar berikut



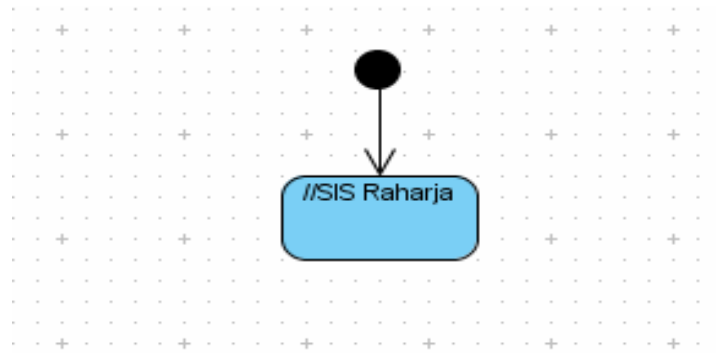
Gambar 6.13

7. Ketikkan nama state pada dialog layar yang ditampilkan sistem sesuai kebutuhan untuk menggantikan kata state pada simbol state (misalnya SIS Raharja) hingga nama state tampak seperti pada gambar berikut



Gambar 6.14

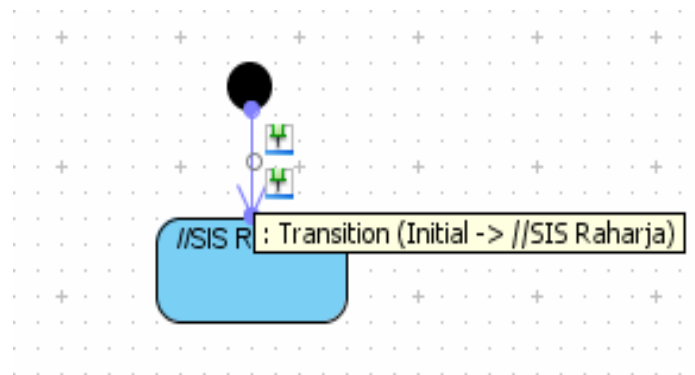
8. Tempatkan dan klik mouse satu kali di luar simbol state hingga sistem menampilkan gambar berikut



Gambar 5.15

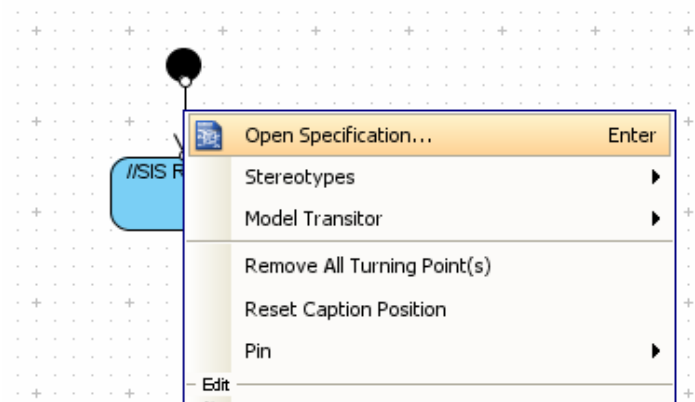
5.4.2 Membuat/Melengkapi State dengan Event atau Action pada Statechart Diagram

1. Tempatkan cursor pada simbol event/action hingga sistem menampilkan gambar berikut



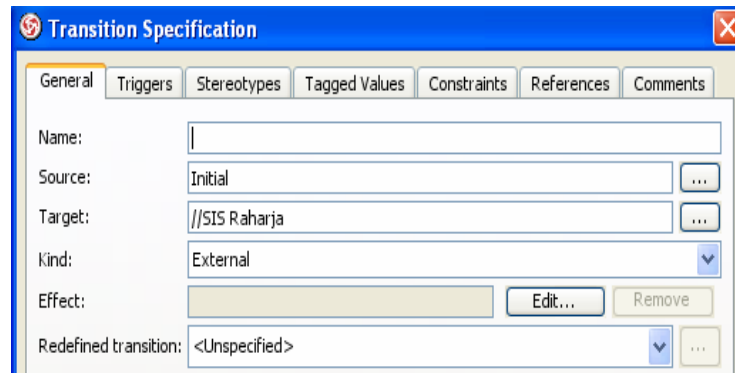
Gambar 5.16

2. Klik kanan mouse sehingga sistem menampilkan dialog layar berikut



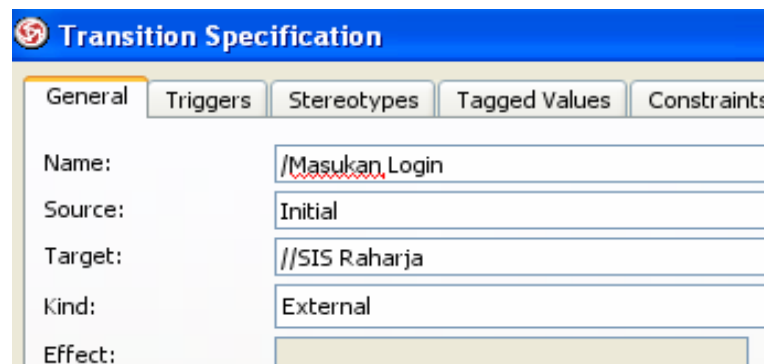
Gambar 5.17

3. Klik fungsi Open Specification yang ada pada dialog layar pada langkah kedua sehingga sistem menampilkan dialog layar Transition Specification berikut



Gambar 5.18

4. Ketikkan /”Masukan Login” pada bagian fungsi Name seperti pada gambar berikut



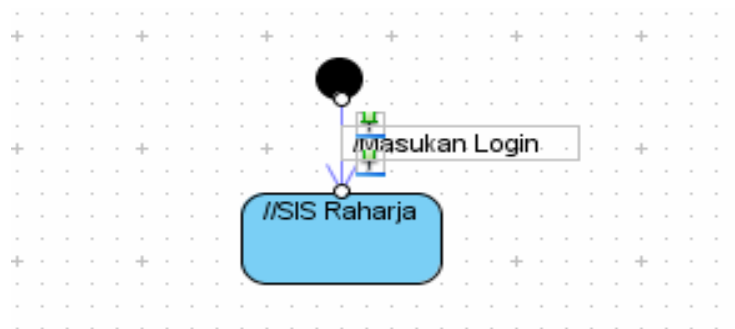
Gambar 5.19

5. Klik fungsi OK yang ada pada bagian bawah dialog layar Transition Specification seperti tampak pada gambar berikut



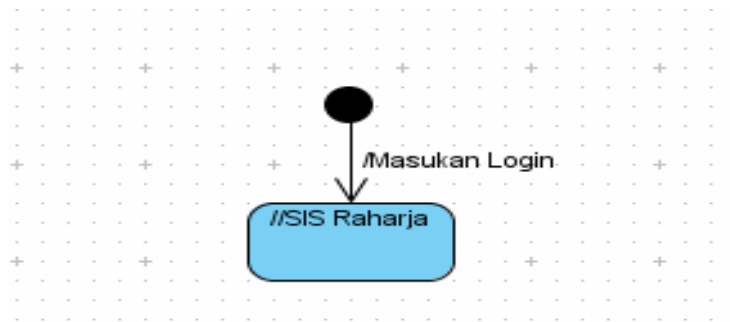
Gambar 5.20

6. Setelah diklik fungsi OK pada langkah kelima sistem akan menampilkan gambar berikut



Gambar 5.21

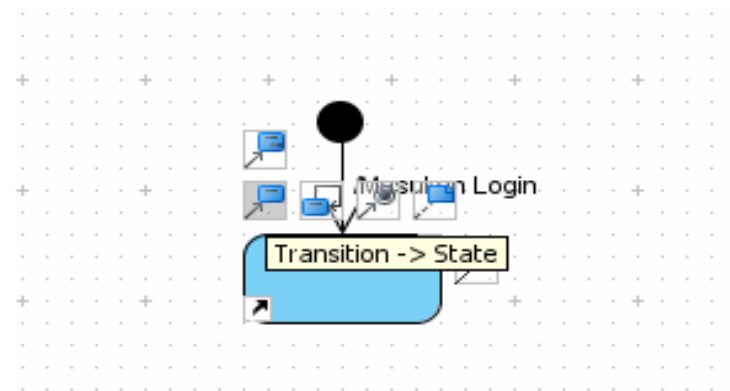
7. Tempatkan dan klik satu kali mouse pada bagian luar simbol state dan event/action hingga sistem akan menampilkan gambar sebagai berikut



Gambar 5.22

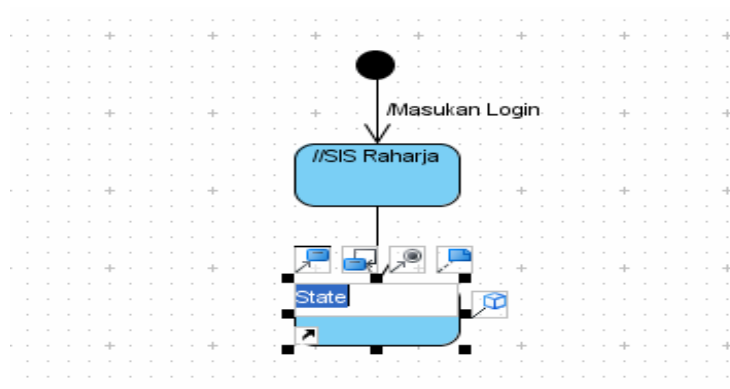
5.4.3 Menghubungkan State dengan State

1. Aktifkan simbol state dengan cara menempatkan kursor pada simbol state hingga sistem menampilkan gambar berikut



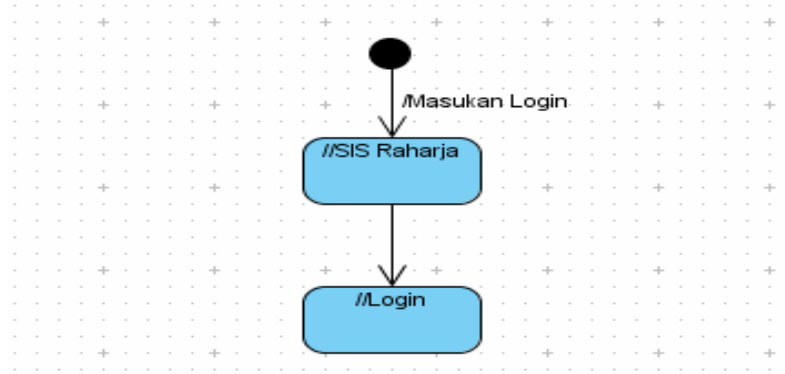
Gambar 5.23

2. Tempatkan kursor pada simbol state seperti pada gambar di atas, drag dan tempatkan state tersebut sesuai dengan kebutuhan hingga sistem menampilkan gambar berikut



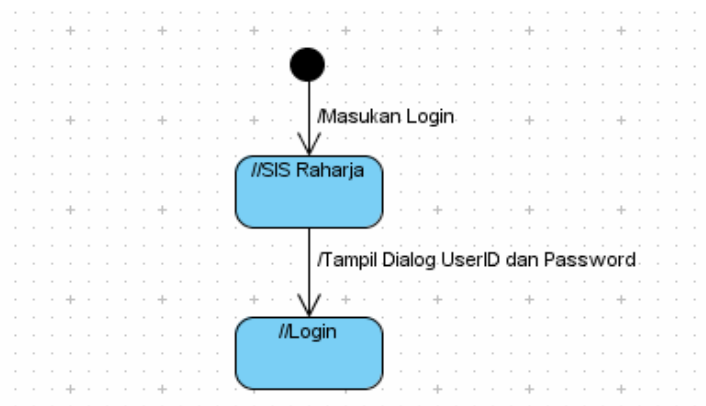
Gambar 5.24

3. Ketikkan nama state ”//Login” dan klik mouse satu kali dibagian luar simbol state hingga sistem menampilkan gambar berikut



Gambar 5.25

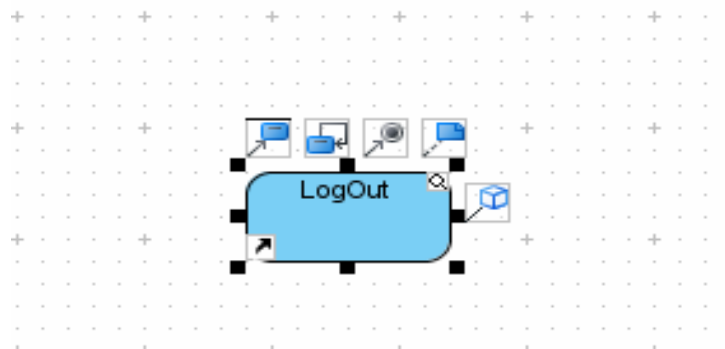
4. Lengkapi nama event/action dari state pertama dan kedua (ulangi langkah-langkah dalam membuat event/action yang telah dijelaskan sebelumnya) hingga sistem menampilkan gambar berikut



Gambar 5.26

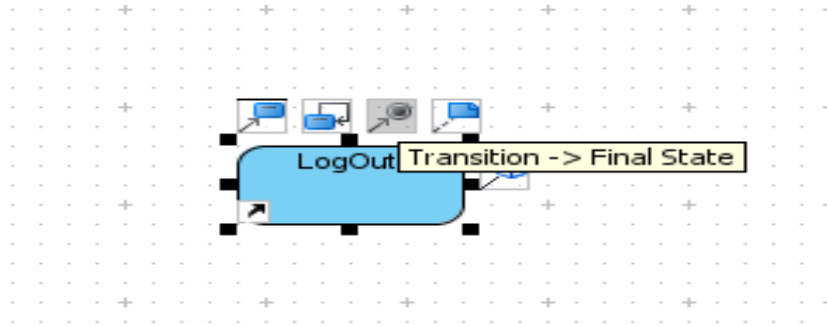
5.4.4 Membuat Akhir State (Final State)

1. Aktifkan simbol state yang merupakan state terakhir statechart diagram yang dibuat dengan cara menempatkan mouse pada simbol state hingga sistem menampilkan gambar seperti berikut



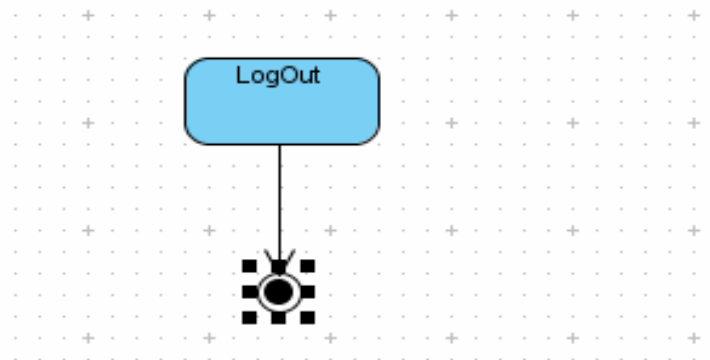
Gambar 5.27

2. Tempatkan kursor pada simbol Final State yang ada di bagian atas simbol state yang diaktifkan hingga tampil dialog berikut



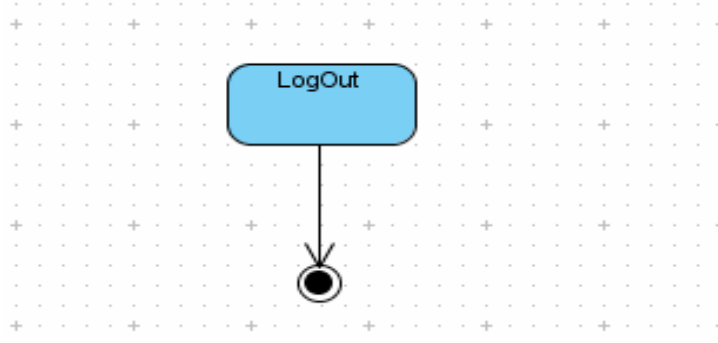
Gambar 5.28

3. Drag simbol Final State tersebut keluar simbol state yang aktif (tempatkan sesuai keinginan) dan lepaskan mouse hingga sistem menampilkan gambar berikut



Gambar 5.29

4. Klik mouse satu kali pada sembarang tempat diluar simbol Final State hingga sistem menampilkan gambar berikut



Gambar 5.30

Soal Latihan

1. Buatlah Statechart diagram untuk menggambarkan behavior (perilaku) sistem ATM dalam merespon stimulasi yang dilakukan oleh nasabah yang ingin menarik dananya melalui ATM. Stimulasi yang dilakukan oleh nasabah adalah memasukkan kartu,



memilih jenis bahasa, memasukan PIN, memilih jenis transaksi penarikan, dan memilih jenis penarikan dengan nominal yang sudah disiapkan oleh sistem ATM dan selesai.

2. Buatlah Statechart diagram untuk menggambarkan perilaku pengiriman sebuah e-mail.

BAB VI

ACTIVITY DIAGRAM

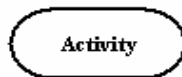
6.1 Definisi Activity Diagram

Activity diagram adalah diagram yang menggambarkan sifat dinamis secara alamiah sebuah sistem dalam bentuk model aliran dan kontrol dari aktivitas ke aktivitas lainnya (Henderi, 2007). Sebuah aktivitas merepresentasikan suatu operasi pada beberapa class dalam sistem yang menghasilkan suatu perubahan keadaan (state) dari sistem tersebut. Secara khusus, activity diagram biasa digunakan untuk memodelkan diagram alir sebuah sistem kerja (*workflow*) atau proses bisnis (prosedur bisnis) dan operasi-operasi secara internal (Miller Randy, 2008). Sementara menurut Whitten L. Jeffery el al (2004: 428), activity diagram adalah sebuah diagram yang dapat digunakan untuk menggambarkan secara grafis aliran proses bisnis, langkah-langkah sebuah use case atau logika behavior (metode) object. Karena sebuah activity diagram adalah bentuk khusus dari statechart diagram, maka activity diagram kadangkala digunakan untuk memodelkan suatu kebiasaan sesuai dengan ketentuan/kaedah bisnis.

Untuk dapat membangun activity diagram yang baik, berikut proses yang unggul untuk membangun activity diagram:

1. Tambahkan poin awal dan akhir pada sebuah use case
2. Tambahkan sebuah kegiatan untuk tiap langkah utama pada use case (atau tiap langkah utama setiap pelaku yang menginisialisasi)
3. Tambahkan transisi dari setiap kegiatan ke kegiatan lain, poin keputusan, atau poin akhir
4. Tambahkan bar sinkronisasi di mana kegiatan dilakukan secara paralel.

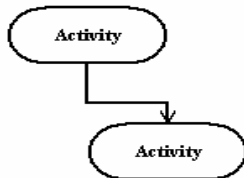
6.2 Simbol dan Notasi Dasar Activity Diagram



Gambar 6.1

Action states

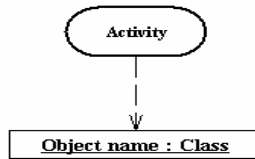
Action state adalah representasi/gambaran dari aksi yang tidak bisa diganggu oleh aksi yang berasal dari objek-objek. Action state digambarkan dalam bentuk empat persegi panjang yang pada sudut-sudutnya melingkar (gambar 6.1)



Gambar 6.2

Action Flow

Action digambarkan dalam bentuk anak panah yang mengilustrasikan relasi antara action pada state (gambar 6.2)

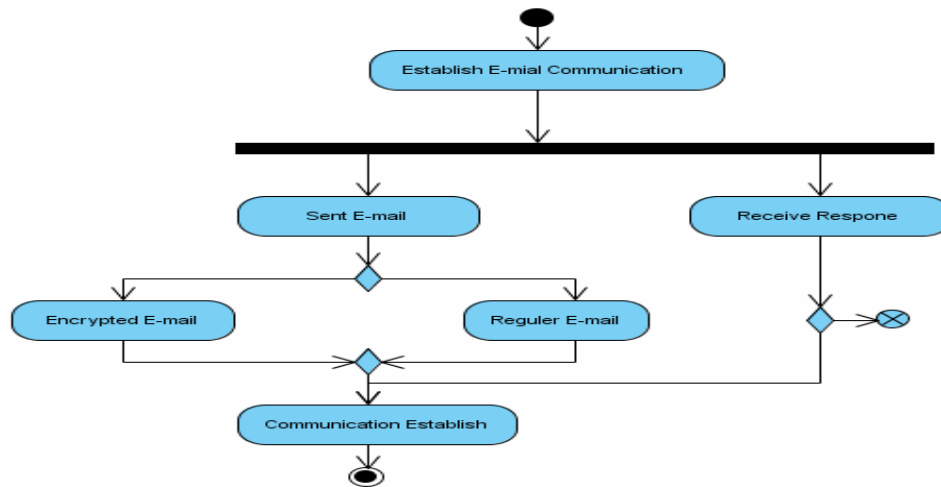


Gambar 6.3

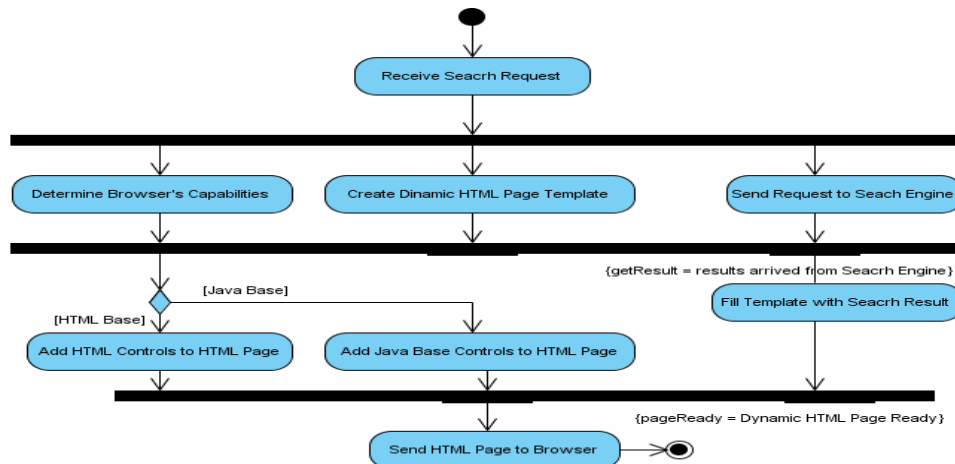
Object Flow

Object flow menunjuk kepada kegiatan penciptaan (meng-create) dan memodifikasi oleh objek melalui activities. Sebuah tanda panah objek flow dari suatu aksi kepada suatu objek berarti bahwa aksi tersebut meng-create atau mempengaruhi objek tersebut. Sementara suatu tanda panah dari objek kepada suatu aksi mengindikasikan bahwa aksi state tersebut menggunakan objek (gambar 6.3).

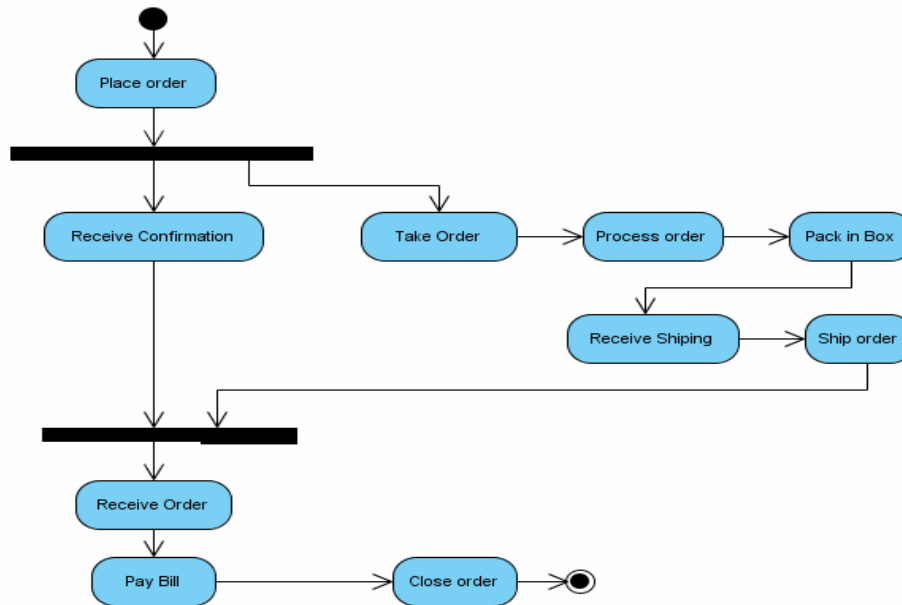
6.3 Contoh Activity Diagram



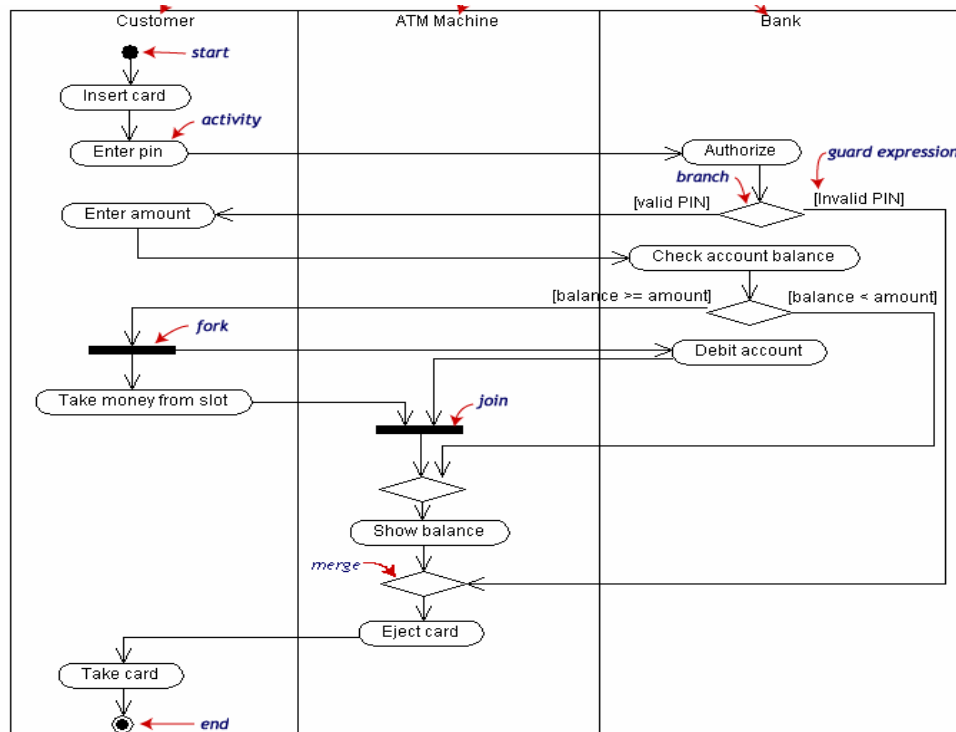
Gambar 6.4 Activity Diagram E-mail Connection



Gambar 6.5 Activity Diagram Web Site



Gambar 6.6 Activity Diagram Order

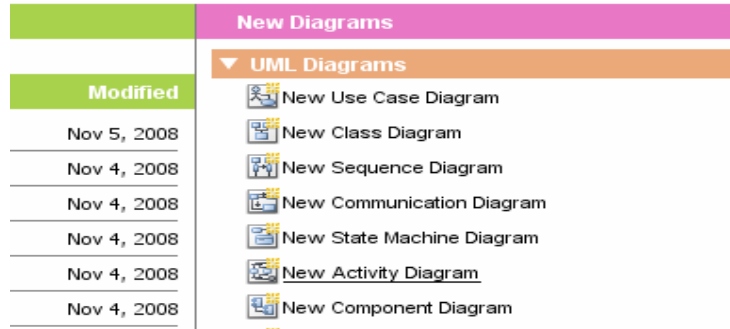


Gambar 6.7 Activity Diagram Menarik Uang
Pada Sebuah Bank Melalui ATM (Miller Randy, 2008)

6.4 Membuat Activity Diagram dengan Visual Paradigm (VP)

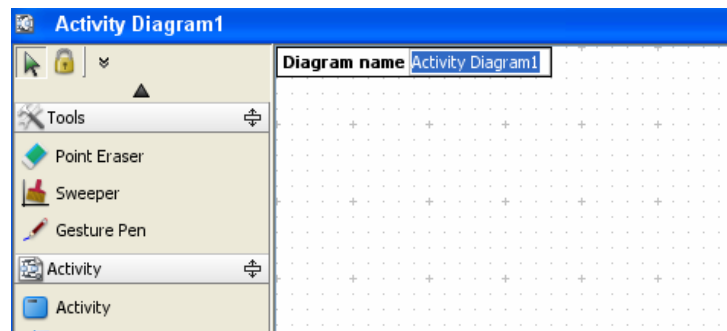
6.4.1 Mengaktifkan Worksheet dan Membuat Awal Activity

1. Arahkan kursor pada bagian **New Activity Diagram** pada dialog utama Visual Paradigm seperti pada gambar berikut



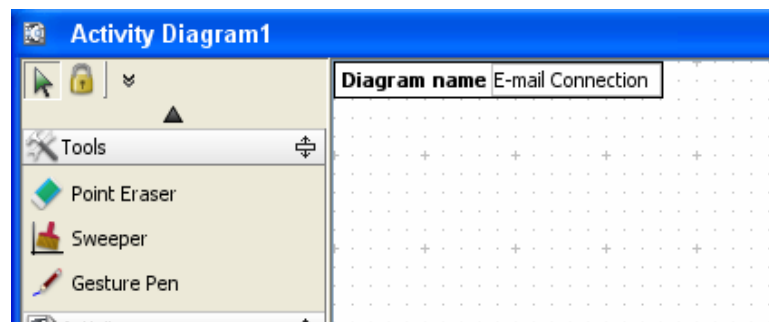
Gambar 6.7

2. Klik satu kali fungsi **New Activity Diagram** tersebut hingga sistem menampilkan worksheet activity diagram sebagai berikut



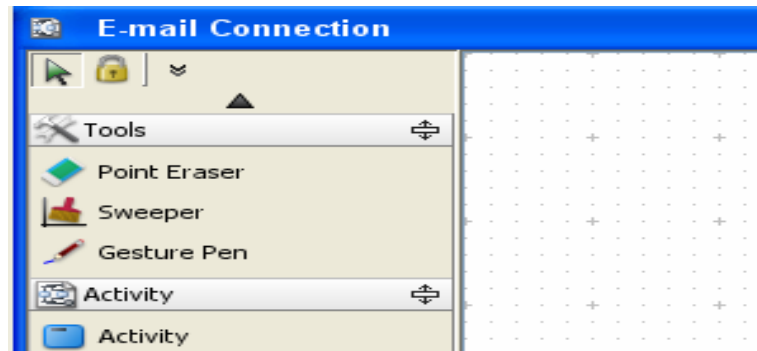
Gambar 6.8

3. Ketikkan nama Activity Diagram yang ingin dibuat pada dialog layar worksheet **Diagram Name** pada gambar langkah kedua (misalnya: E-mail Connection) seperti pada gambar berikut



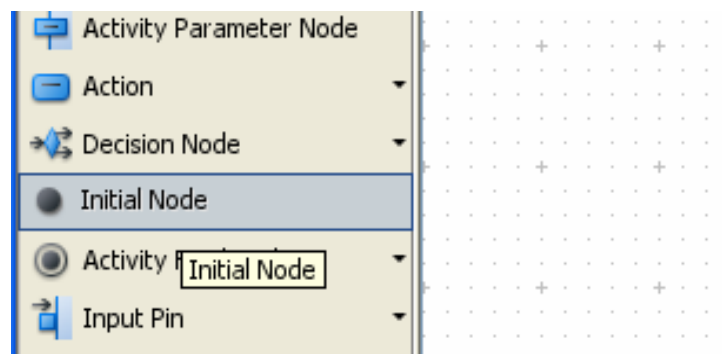
Gambar 6.9

4. Klik mouse satu kali pada sembarang tempat worksheet activity diagram hingga sistem menampilkan worksheet seperti pada gambar berikut



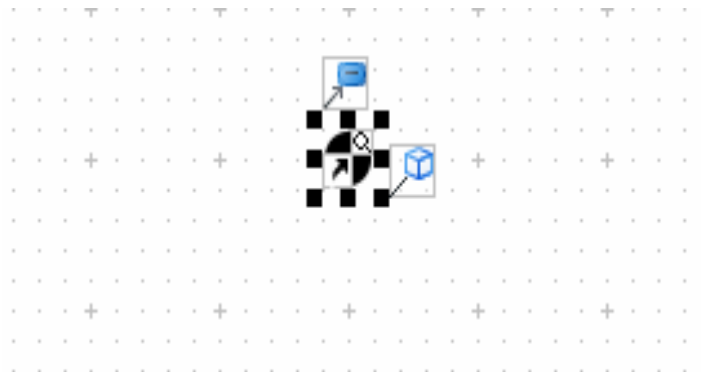
Gambar 6.10

5. Arahkan kursor pada simbol Initial Node yang ada pada dialog layar **Tools** yang ditampilkan oleh sistem seperti pada gambar berikut



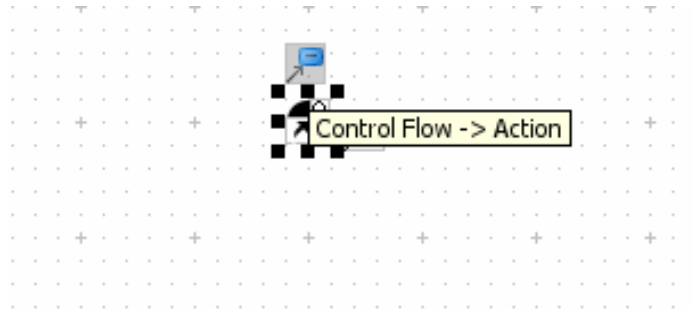
Gambar 6.11

6. Klik satu kali simbol Initial Node dan letakan kursor pada layar worksheet dan klik satu kali mouse hingga sistem menampilkan gambar berikut



Gambar 6.12

7. Arahkan kursor pada simbol Activity yang ada pada bagian simbol Initial Node hingga sistem menampilkan gambar berikut



Gambar 6.13

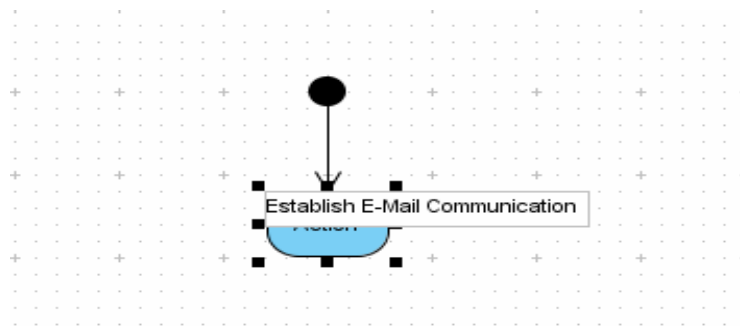
8. Buatlah activity pertama dengan cara men-drag (klik dan tarik) simbol Activity/Control Flow, dan menempatkannya pada salah satu bagian worksheet, dan lepaskan mouse dari tekanan hingga sistem menampilkan gambar berikut



Gambar 6.14

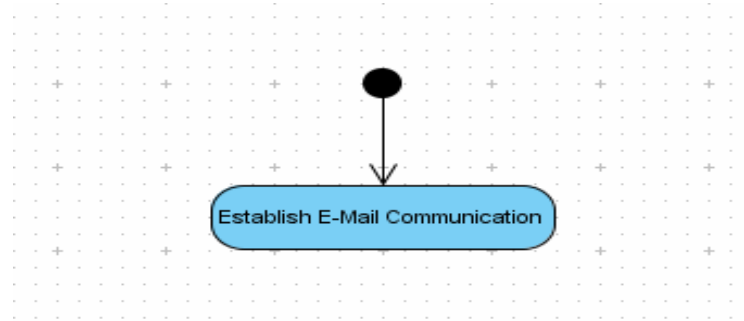
6.4.2 Memberi Nama Action State/Activity dan Action Flow

1. Ketikkan nama Action State (activity) pada dialog layar action pada gambar langkah kedelapan misalnya dengan kalimat **Establish E-mail Connection** seperti pada gambar berikut



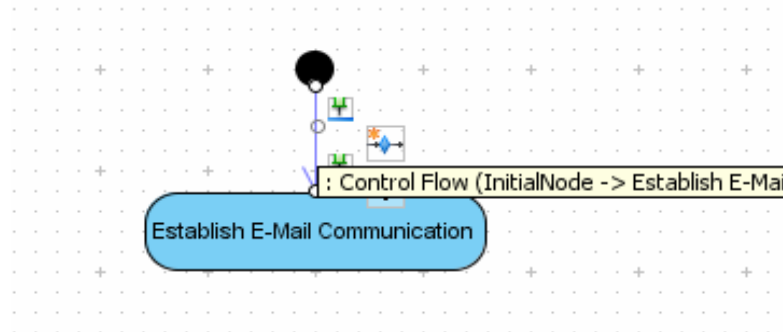
Gambar 6.15

2. Klik satu kali pada sembarang tempat hingga sistem menampilkan nama action state seperti pada gambar berikut



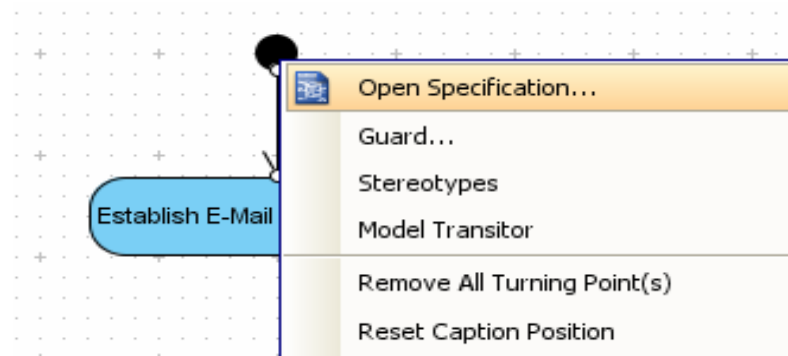
Gambar 6.16

3. Memberikan nama pada action flow; tempatkan kursor pada simbol action flow dan klik satu kali simbol tersebut hingga sistem menampilkan gambar berikut



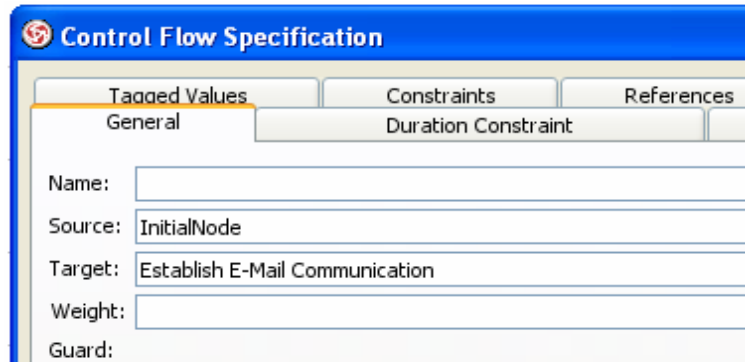
Gambar 6.17

4. Klik kanan mouse hingga sistem menampilkan dialog layar berikut



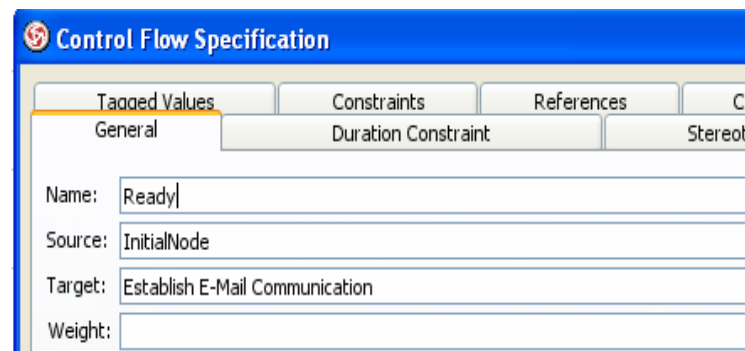
Gambar 6.18

5. Klik fungsi **Open Spesification** hingga sistem menampilkan dialog layar berikut



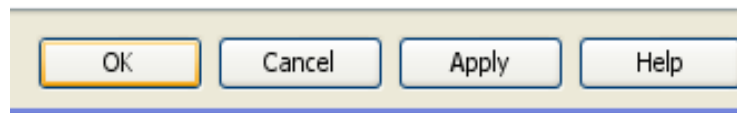
Gambar 6.19

6. Tempatkan kursor pada dialog layar Name dan ketikkan nama Control Flow Spesification sesuai dengan kebutuhan (misalnya Ready) seperti pada gambar berikut



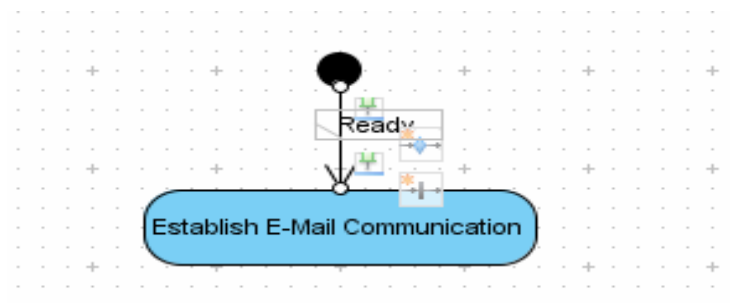
Gambar 6.20

7. Klik fungsi **OK** yang ada pada bagian bawah dialog layar seperti pada gambar berikut



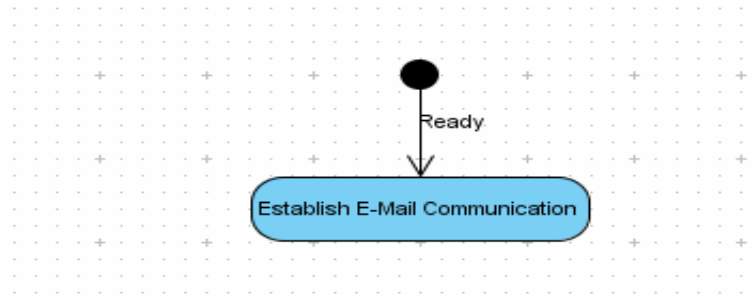
Gambar 6.21

8. Sistem akan menampilkan gambar berikut (sesuaikan tata letak nama control flow) hingga sistem menampilkan gambar berikut



Gambar 6.22

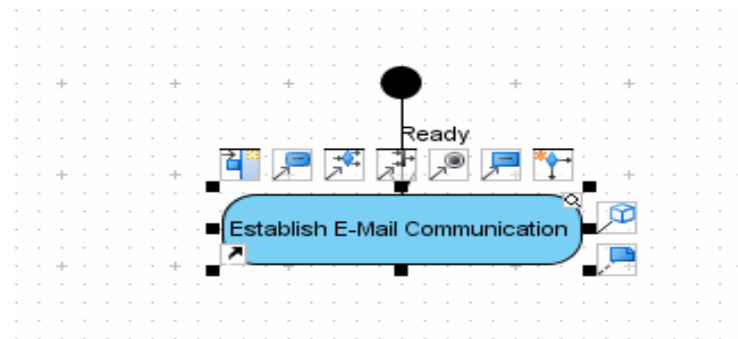
9. Klik satu kali mouse di sembarang tempat worksheet hingga sistem menampilkan gambar activity diagram berikut



Gambar 6.23

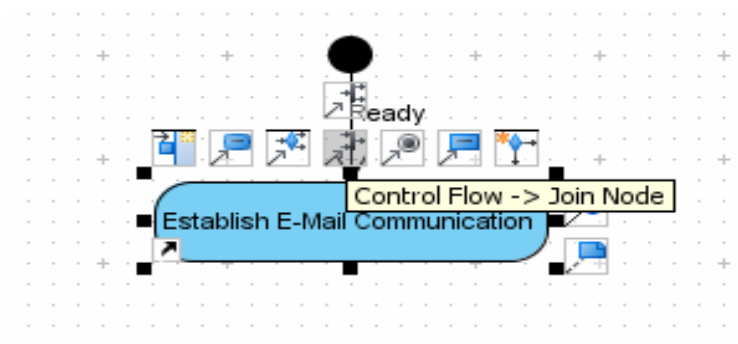
6.4.3 Membuat Join Node

1. Aktifkan simbol state hingga sistem menampilkan gambar berikut



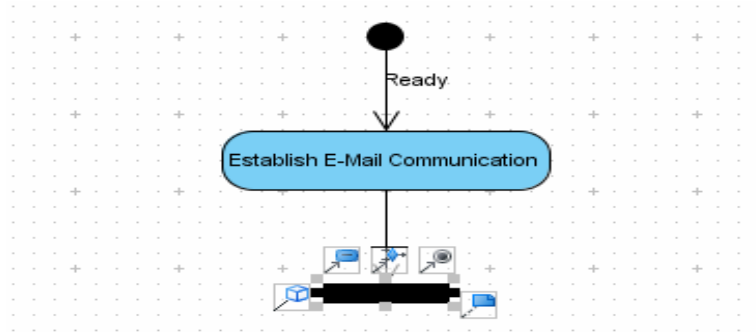
Gambar 6.24

2. Tempatkan kursor pada simbol Control Flow → Join Node hingga sistem menampilkan gambar berikut



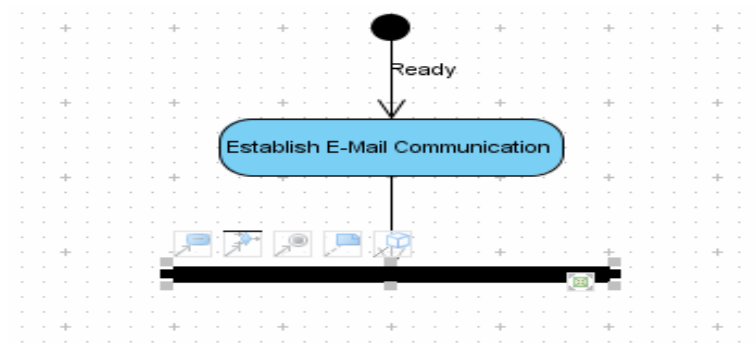
Gambar 6.25

3. Drag (tekan, tarik dan lepas) simbol Control Flow → Join Node kepada bagian bawah state yang aktif hingga sistem menampilkan gambar berikut



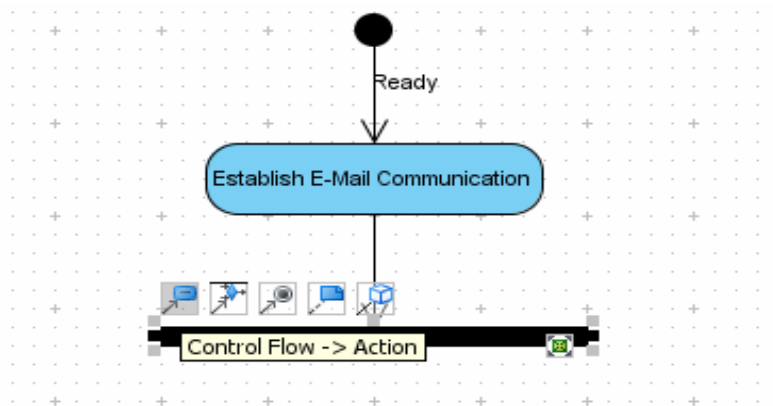
Gambar 6.26

4. Tarik simbol Join Node ke kanan dan ke kiri sesuai dengan kebutuhan seperti gambar berikut



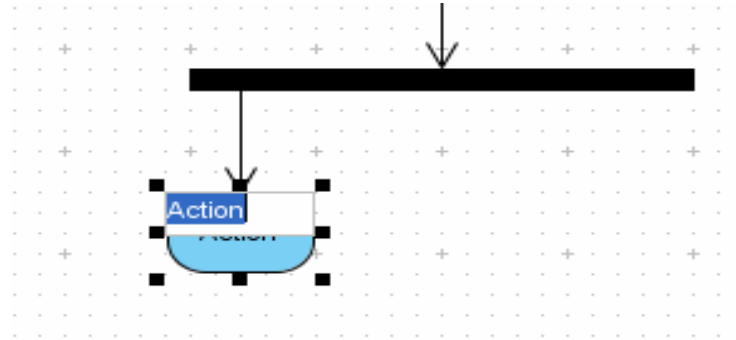
Gambar 6.27

5. Tempatkan kursor pada simbol control flow → action seperti pada



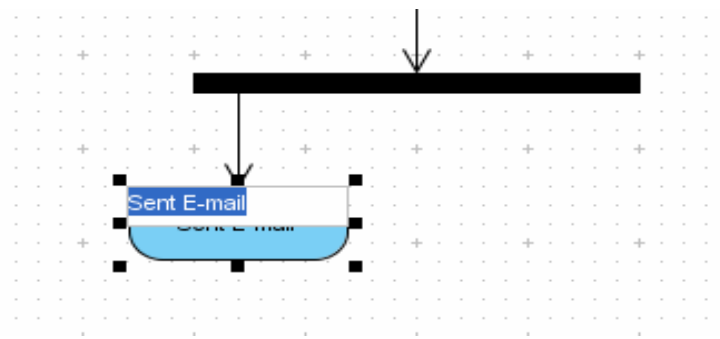
Gambar 6.28

6. Drag (tekan-tarik dan lepas) simbol Control Flow → Action ke bawah hingga sistem menampilkan gambar berikut



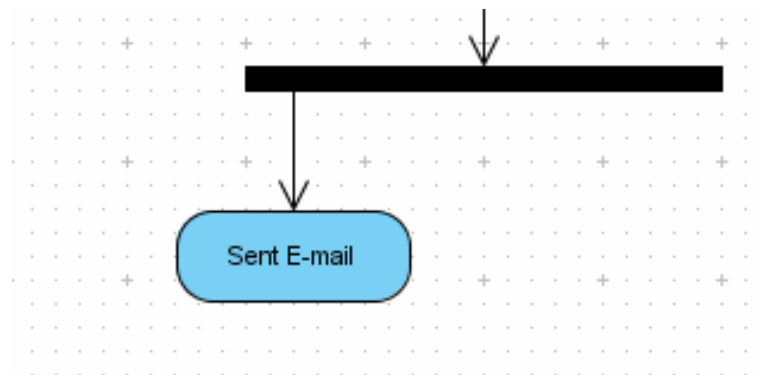
Gambar 6.29

7. Ketikkan nama action state pada bagian dialog layar state langkah keenam hingga sistem menampilkan gambar berikut



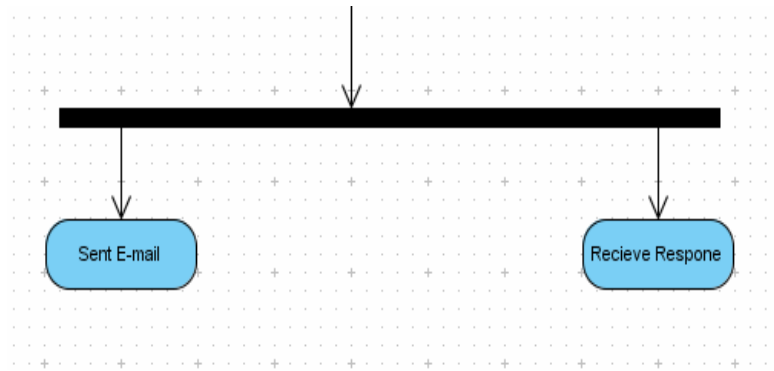
Gambar 6.30

8. Klik mouse satu kali disembarang tempat pada worksheet hingga sistem menampilkan gambar berikut



Gambar 6.40

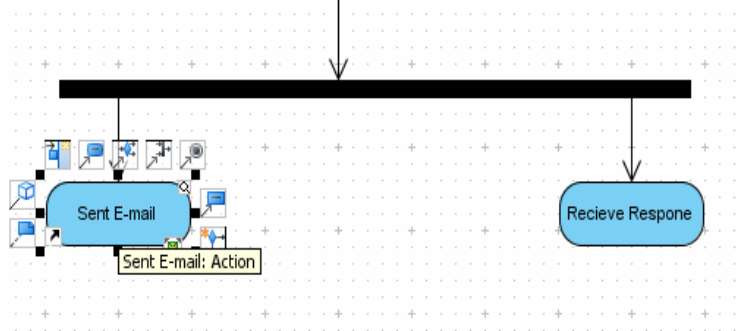
9. Buat action state pada bagian kanan simbol Join Node dengan mengulangi langkah enam sampai delapan dengan nama action state **Recieve Response** hingga sistem menampilkan gambar berikut



Gambar 6.41

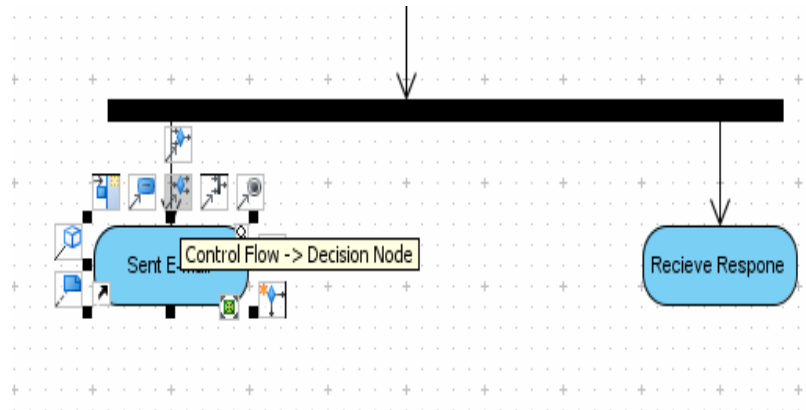
6.4.4 Membuat Decision Node

1. Aktifkan simbol action states Sent E-mail pada langkah kesembilan langkah membuat simbol **Join Node** hingga sistem menampilkan gambar berikut



Gambar 6.42

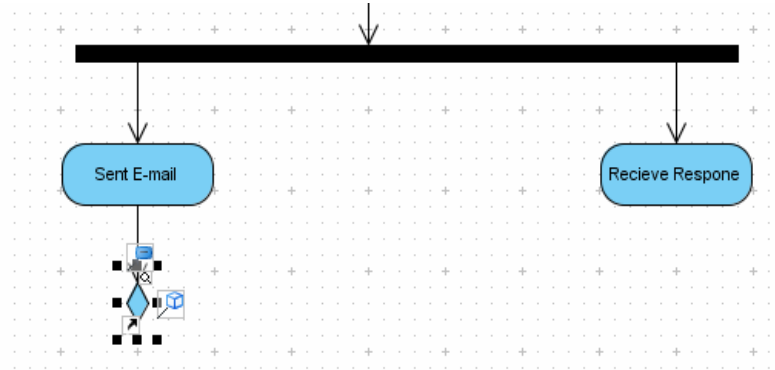
2. Tempatkan kursor pada simbol Decision Node yang ada pada bagian atas simbol action state hingga sistem menampilkan gambar berikut



Gambar 6.43

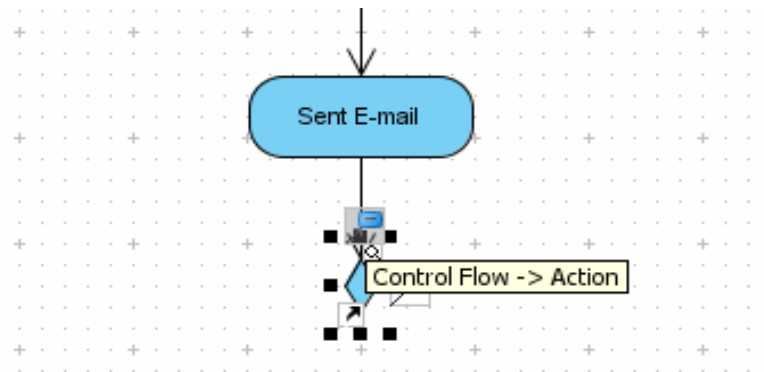
3. Drag (tekan, tarik dan lepaskan) simbol Control Flow → Decision Node hingga sistem menampilkan gambar berikut





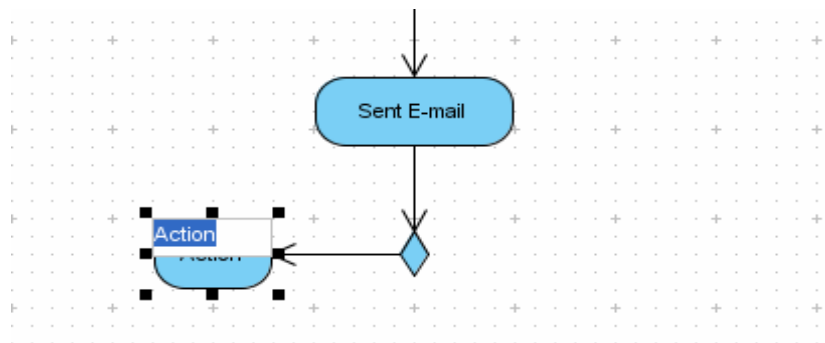
Gambar 6.44

4. Buatlah action state yang mengarah ke kiri dan ke kanan simbol Decision Node dengan mengaktifkan simbolnya dan mengarahkan kursor pada simbol Control Flow → action seperti gambar berikut



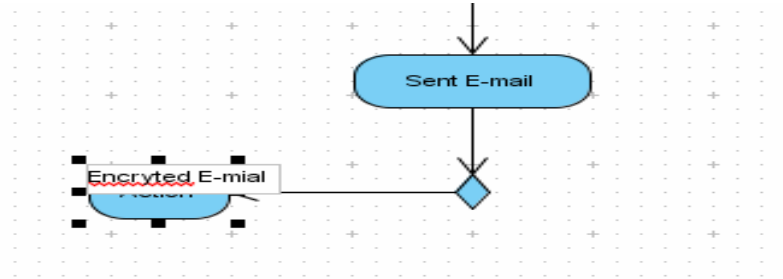
Gambar 6.45

5. Drag (tekan, tarik dan lepas) simbol action state sebelah kiri Decision Node hingga sistem menampilkan gambar berikut



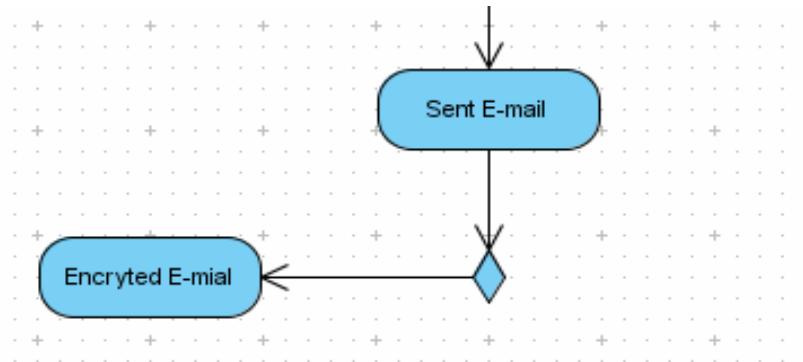
Gambar 6.46

6. Ketikkan nama action state pada dialog layar state (misalnya Encrypted E-mail) seperti pada gambar berikut



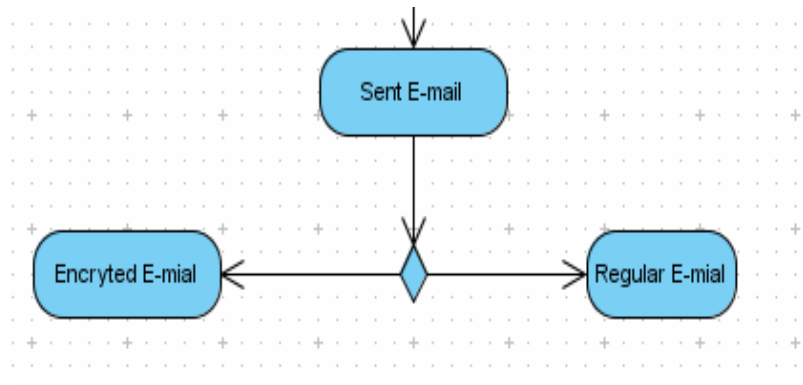
Gambar 6.47

- Klik mouse satu kali dibagian luar simbol action state hingga sistem menampilkan gambar berikut



Gambar 6.48

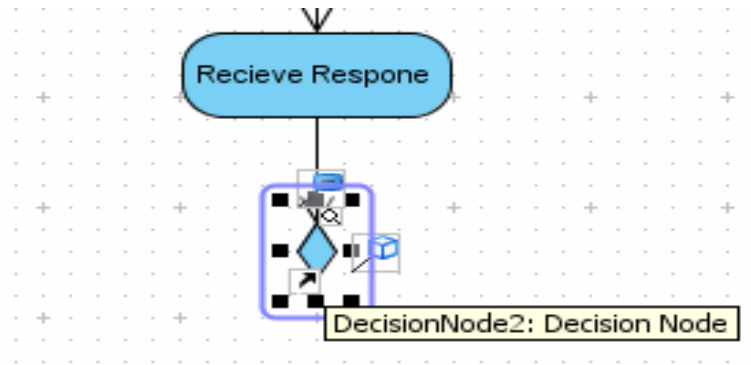
- Lengkapi Decision Node dengan action state untuk bagian sebelah kanan dengan mengulangi langkah nomor empat sampai tujuh hingga sistem menampilkan gambar berikut



Gambar 6.49

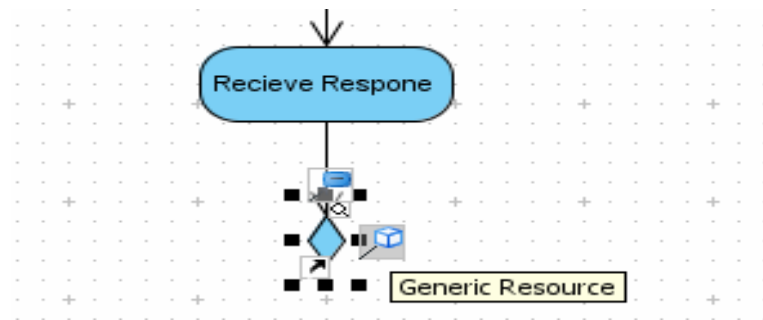
6.4.5 Membuat Flow Final Node

- Klik satu kali simbol Decision Node untuk mengaktifkannya hingga sistem menampilkan gambar berikut



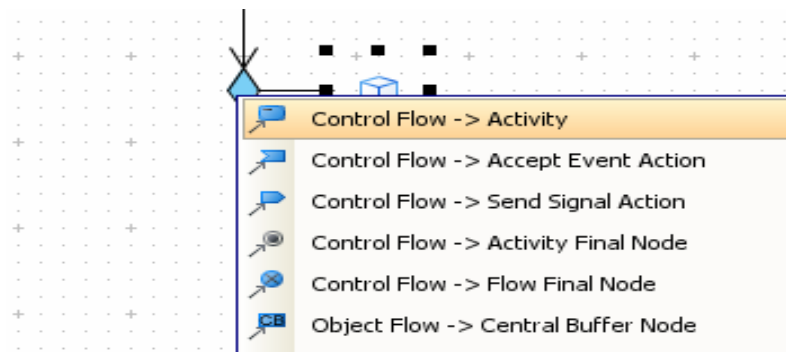
Gambar 6.50

- Tempatkan kursor pada simbol Generric Recource yang ada pada simbol Decision Node seperti pada gambar berikut



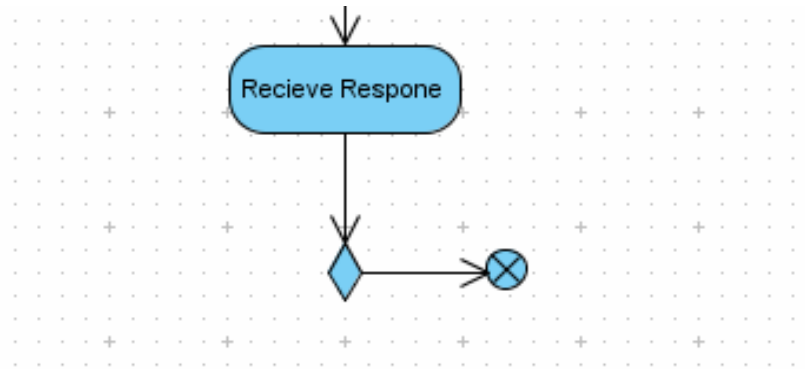
Gambar 6.51

- Drag (tekan, tarik dan lepas) simbol Generic Recource kearah sebelah kanan Decision Node hingga sistem menampilkan layar dialog berikut



Gambar 6.52

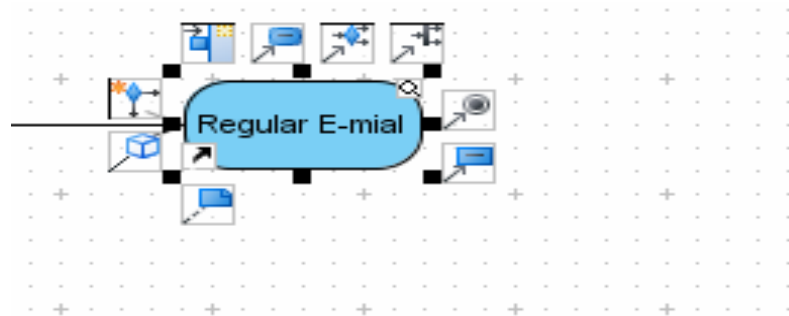
- Klik fungsi **Control Flow→Flow Final Node** yang ada pada layar dialog pada langkah ketiga hingga sistem menampilkan gambar Flow Final Node berikut



Gambar 6.53

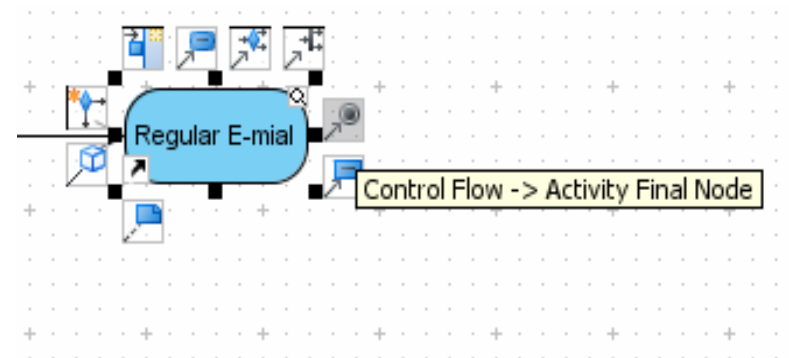
6.4.6 Membuat Activity Final Node

1. Aktifkan simbol action state yang merupakan action state/activity dari activity diagram (misalnya action state/activity close order) hingga sistem menampilkan gambar berikut



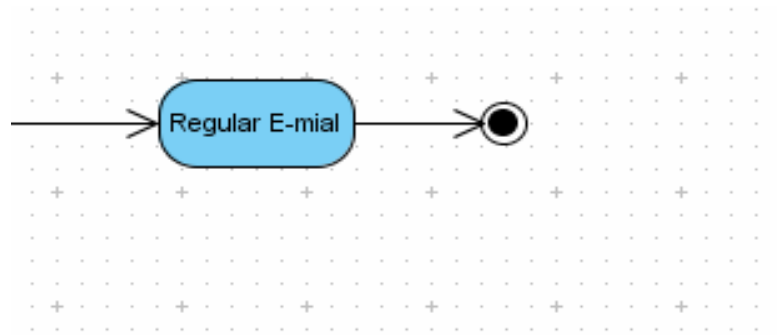
Gambar 6.54

2. Tempatkan kursor pada simbol Activity Final Node yang ada pada sebelah kanan simbol action state seperti gambar berikut



Gambar 6.55

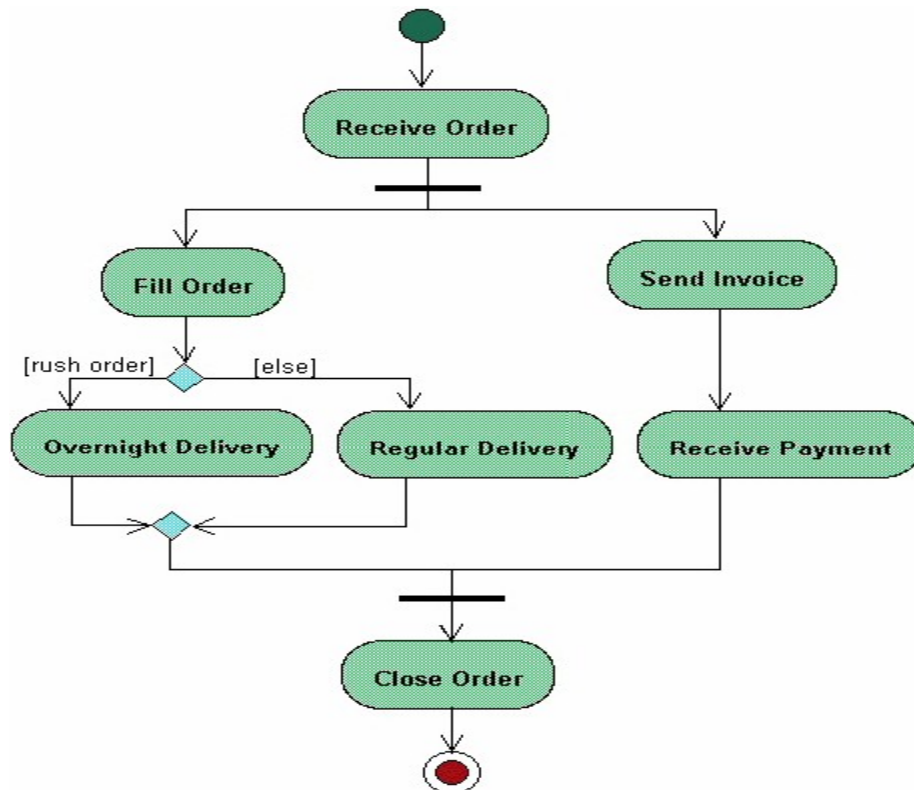
3. Drag (tekan, tarik dan lepas) simbol Control→Activity Final Node ke arah sesuai kebutuhan hingga sistem menampilkan gambar berikut (Activity Final Node selesai dibuat)



Gambar 6.56

Soal Latihan

Buatlah activity diagram berdasarkan gambar berikut menggunakan software yang mendukung pemodelan berbasis visual menggunakan Unified Modeling Language (UML).



Gambar 5.57 Activity Diagram Order



CASE STUDY 1

Suatu universitas yang mempunyai lima fakultas dengan masing-masing fakultas paling tidak mempunyai 3 jurusan dengan program studi lebih dari dua. Universitas ini sebelumnya telah menerapkan sistem komputerisasi pada sistem informasi akademik, tetapi belum terintegrasi antara satu fakultas dengan fakultas lainnya termasuk antara jurusan dengan jurusan lainnya. Saat ini Universitas ini telah mempunyai mahasiswa 10 ribuan yang berasal dari berbagai daerah dengan alumni lebih dari 3500-an yang tersebar di seluruh wilayah Indonesia bahkan beberapa negara di dunia. Baru-baru ini Manajemen Universitas telah memutuskan bahwa sistem informasi akademik tersebut perlu dikembangkan menjadi sistem informasi akademik yang terintegrasi dengan catatan pemrosesan data dapat dilakukan secara terdistribusi dan semaksimal mungkin dapat meningkatkan efisiensi dari segi operasional, tetapi sistem informasi ini tetap dituntut memperhatikan mutu perguruan tinggi. Prioritas utama bisa memenuhi kebutuhan tingkatan pengguna database dan semua tingkatan manajemen.

Pertanyaan:

Team anda memenangkan tender proyek pengembangan sistem tersebut dengan nilai proyek 250 juta. Dalam team ini, anda adalah anggota Pokja Analist dan Perancang Sistem dan diminta untuk merancang sistem yang akan digunakan pada sistem tersebut. Buatlah rancangan sistem tersebut menggunakan:

1. Use Case Diagram
2. Class Diagram
3. Sequence Diagram
4. Statechart Diagram
5. Activity Diagram

CASE STUDY 2

Sanggar Tari Nusantara mengajarkan berbagai tari-tarian daerah serta tari-tarian modern-kreatif. Tari-tarian yang diajarkan secara kelompok (minimal 5 peserta, maksimal 10 peserta), serta diajarkan secara privat (seorang peserta diajar oleh seorang pegajar).

Biaya tari untuk peserta yang berlatih secara kelompok adalah Rp. 10.000,- per jam per orang, sedangkan peserta yang berlatih secara privat harus membayar sebesar Rp. 50.000,- per orang per jam. Latihan diadakan dua kali seminggu selama satu jam. Bila ketika latihan peserta tidak hadir, maka ketidakhadiran peserta adalah resiko peserta itu sendiri.

Pendaftaran peserta dilakukan setiap hari, dan latihan dimulai pada bulan berikutnya. Bila batas minimum peserta tidak dipenuhi, maka kegiatan latihan tidak dilakukan. Sementara penghasilan yang diperoleh sanggar tari selain berasal kegiatan latihan tari adalah dari order-order menari yang sering diperoleh dari hotel-hotel maupun dari televisi.

Honor pengajar = gaji pokok + jumlah jam mengajar (indeks honor) + honor order tarian. Indeks honor mengajar besarnya bervariasi untuk setiap pengajar mulai dari Rp. 20.000,- sampai Rp. 40.000,- per jam. Pengajar *part time* tidak mendapat gaji pokok. Untuk pengajar *full time*, gaji pokok diberikan variatif.

Sanggar tari tersebut sering mengisi acara di hotel-hotel dan televisi baik untuk menarik tarian tradisional maupun tarian modern-kreatif. Order-order ini digunakan untuk memberikan kesempatan bagi siswa untuk meningkatkan kemampuan mereka, serta untuk memberikan tambahan penghasilan bagi para pengajar. Setiap ada order tarian, pengurus membuat rancangan biaya dan diturunkan sampai detail untuk setiap tarian yang diminta.

Pertanyaan (berdasarkan Case Study di atas dan untuk menjawab kebutuhan sistem tersebut):

1. Buatlah Use Case Diagram
2. Buatlah Class Diagram
3. Buatlah Sequence Diagram
4. Buatlah Statechart Diagram
5. Buatlah Activity Diagram



CASE STUDY 3

Perusahaan rekaman Notown memutuskan untuk menyimpan semua informasi mengenai musisi yang mengerjakan albumnya [seperti halnya data perusahaan lain] dalam sebuah database. Pihak perusahaan menyewa Anda sebagai desainer database [dengan biaya konsultasi sebesar \$2.500/hari]. Beberapa informasi awal yang diberikan oleh Perusahaan Notown kepada Anda adalah sebagai berikut:

- a. Tiap musisi yang melakukan rekaman di Notown mempunyai SSN, nama, alamat, dan nomor telepon. Para musisi yang dibayar lebih rendah akan mendapatkan alamat yang sama dengan musisi lain, dan satu alamat mempunyai satu nomor telepon
- b. Tiap instrumen yang digunakan untuk merekam berbagai macam lagu di Notown mempunyai nama [contoh: gitar, synthesizer, flute] dan kunci musik (contoh: C, B-flat, E-flat)
- c. Tiap album yang dicatat di Notown mempunyai judul rekaman, tanggal copyright, format (contoh: CD atau MC), dan sebuah identifikasi album
- d. Tiap lagu yang dicatat di Notown mempunyai judul dan pengarang lagu
- e. Tiap musisi mungkin memainkan beberapa instrumen, dan tiap instrumen dapat dimainkan oleh beberapa musisi
- f. Tiap album mempunyai beberapa lagu di dalamnya, tetapi tidak ada lagu yang muncul bersamaan dalam satu album
- g. Tiap lagu dibawahkan oleh satu atau lebih musisi, dan seorang musisi bisa membawakan beberapa lagu
- h. Tiap album dibawahkan seorang musisi yang berperan sebagai produser. Seorang musisi bisa menghasilkan beberapa album











Pertanyaan:

Team anda memenangkan tender proyek pengembangan sistem tersebut dengan nilai proyek \$150 juta. Dalam team ini, anda adalah anggota Pokja Desain Sistem, dan ditugaskan untuk merancang database dan sequence diagram sesuai dengan requirement yang telah ditetapkan sebelumnya (pada soal).

DAFTAR SIMBOL YANG DIGUNAKAN DALAM UNIFIED MODELING LANGUAGE (UML)

(Disarikan oleh Henderi et al, 2007 (dari berbagai sumber))

Daftar Simbol dalam Use Case Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>independent</i>).
3		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
4		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

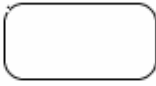




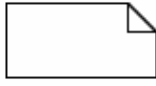
Daftar Simbol dalam Class Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
2		<i>N-Ary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		<i>Class</i>	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		<i>Collaboration</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.
6		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan memengaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya






Daftar Simbol dalam Sequence Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		<i>LifeLine</i>	Objek <i>entity</i> , antarmuka yang saling berinteraksi.
2		<i>Message</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
3		<i>Message</i>	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

Daftar Simbol dalam Statechart/Statemachine Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		<i>State</i>	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		<i>Initial Pseudo State</i>	Bagaimana objek dibentuk atau diawali
3		<i>Final State</i>	Bagaimana objek dibentuk dan dihancurkan
4		<i>Transition</i>	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		<i>Node</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Daftar Simbol dalam Activity Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		<i>Initial Node</i>	Bagaimana objek dibentuk atau diawali.
4		<i>Activity Final Node</i>	Bagaimana objek dibentuk dan dihancurkan
5		<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran



P U S T A K A

Henderi, 2007, **Analysis and Design System with Unified Modeling Language (UML)**, STMIK Raharja, Tangerang

Henderi, 2007, **Advanced Modeling with Unified Modeling Language (UML)**, STMIK Raharja, Tangerang

Henderi, 2007, **Perancangan Sistem Informasi**. Diakses pada tanggal 15 Juli 2007 dari <http://www.rme.com/rce/materi.asp?km=SI1111&fol=henderi&kd=02019>

Henderi, Efana Rahwanto, Arie Waskito, 2007, **Tutorial Unified Unified Modeling Language (UML)**, Perguruan Tinggi Raharja Tangerang

Henderi, Euis Siti Nuraisyh, dan Efana Rahwanto, 2008, **Handout Workshop Unified Unified Modeling Language (UML) dan Tutorial Visual Paradigm For UML Enterprise Edition**. Diakses pada tanggal 17 Juni 2008 dari <http://www.stmik-raharja.com/Henderi2/rec/UML.htm>

Henderi, Padeli, dan Suyaton, 2008, **Membangun E-Procurement dengan Prinsip Good Governance dengan Visual UML**. Creative Communication and Innovative Technology (CCIT). 2 (3), 69-79

Miller Randy, **Practical UML: A Hands-On Introduction for Developers**. Diakses pada tanggal 14 Mei 2008 dari: www.Sequence-Diagram-Hands-On Introduction for Developers.mht

Whitten L Jeffery, Bentley D. Lonnie, Dittman C. Kevin, 2004, **Metode Desain dan Analisis Sistem**, Edisi 6., Edisi Internasional Mc.Graw Hill Education dan Penerbit Andi Jogjakarta

Ziga Turck, Assoc. Prof., **What is UML**, Istambul Techincal University, MBA in Contruction Informatics, Rational Corporation Istambul. Diakses pada 4 Desember 2007 dari: www.rational.com dan http://www.gdpro.com/what_is_uml.html

www.rational.com

<http://www.uml.org/>

http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/index.htm

<http://www.uml.com.pl/>

http://en.wikipedia.org/wiki/Unified_Modeling_Language

<http://www.methodsandtools.com/uml.html>



Henderi, M. Kom.

Dilahirkan di Bengkulu Selatan, 4 Desember 1974. Menyelesaikan Pendidikan Dasar sampai Menengah Atas di Bengkulu Selatan, Sarjana Komputer (S1) Jurusan Sistem Informasi di Universitas Bina Darma Palembang Tahun 2000, dan Magister Komputer dari Sekolah Tinggi Teknologi Informasi Eresha (d/h STTI Benarif Indonesia) Tahun 2006. Saat ini dalam proses preparation for Doctor in Computer Science .

