

MATERI PELENGKAP MODUL (BAHAN AJAR)

**DIKLAT FUNGSIONAL PRANATA KOMPUTER
TINGKAT AHLI**

Mata Diklat:
PERANCANGAN SISTEM INFORMASI



Disusun oleh :

Utama Andri A. ST., MT
Widyaiswara Ahli Muda

**PUSAT PENDIDIKAN DAN PELATIHAN BADAN PUSAT STATISTIK RI
(PUSDIKLAT BPS RI)**

Daftar Isi

Daftar Isi	2
PERANCANGAN BERORIENTASI OBYEK	3
1. Penjelasan Umum.....	3
2. Objek dan Kelas	3
3. Pengertian UML.....	3
4. Use Case Diagram.....	4
5. Class Diagram	7
6. Activity Diagram	11
6. Sequence Diagram	13

PERANCANGAN BERORIENTASI OBYEK

1. Penjelasan Umum

Perancangan berorientasi objek menitik beratkan pada penggambaran struktur dan tingkah laku sistem informasi yang meliputi data dan proses. Secara statis struktur data dan proses akan menunjukkan hubungan antar bagian dari sistem, sedangkan secara dinamis menunjukkan bagaimana bagian-bagian sistem tersebut akan berinteraksi antara satu dengan lainnya. Untuk menggambarkan tersebut diperlukan pemahaman dasar dari class, object, method, message, encapsulation, inheritance, polymorphism, dan dynamic binding.

2. Objek dan Kelas

Objek adalah gambaran suatu entitas atau unit, baik dalam dunia nyata atau konsep dengan batasan-batasan dan pengertian yang tepat. Objek dapat mewakili sesuatu yang nyata seperti komputer, mobil, dll. Objek juga dapat berupa konsep seperti proses transaksi bank, pembelian barang, penambahan pegawai, dll. Dan setiap objek memiliki tiga karakteristik yaitu state (status), behaviour (sifat/kemampuan), dan identity (identitas/atribut).

Kelas merupakan gambaran sekumpulan objek yang terbagi dalam atribut, operasi metode, hubungan, dan makna yang sama. Kelas objek merupakan wadah bagi objek, yang dapat digunakan untuk menciptakan objek

Contoh Kelas:

Orang
+ nama : String + umur : int
+ makan() : void

Nama Kelas : Orang

Atribut : Nama, Umur

Method : Makan

3. Pengertian UML

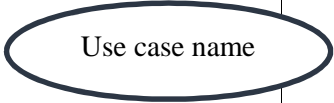
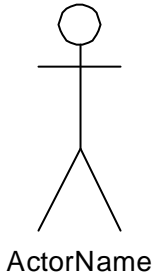
Teknik yang digunakan untuk membuat perancangan berorientasi objek adalah Unified Modelling Language (UML). UML disebut sebagai bahasa yang telah distandarisasi untuk

digunakan dalam memodelkan suatu software atau sistem. UML merupakan bahasa yang memberikan vocabulary dan tantatan penulisan kata untuk kegunaan komunikasi. UML bukan saja merupakan bahasa visual saja, namun juga dapat secara langsung dihubungkan ke berbagai bahasa pemrograman, seperti JAVA, C++, dll.

Ada beberapa diagram yang perlu dibuat dalam merancangan menggunakan UML, diantaranya yaitu Use Case Model, Class Diagram, Activity/ Sequence Diagram, dan State Chart.

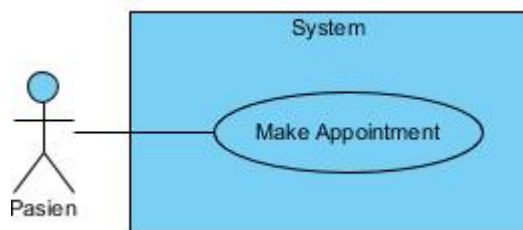
4. Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Berikut beberapa komponen dalam membuat use case diagram.

Nama Objek	Penjelasan	Simbol
Use Case	urutan tindakan, termasuk varian yang dapat dilakukan sistem (atau entitas lain), berinteraksi dengan aktor sistem	
Actor	seperangkat peran yang koheren yang dimainkan pengguna kasus penggunaan saat berinteraksi dengan kasus penggunaan ini	
System Boundary	mewakili batas antara sistem fisik dan aktor yang berinteraksi dengan sistem fisik	
Association	partisipasi aktor dalam use case, instance dari aktor dan instance dari penggunaan berkomunikasi satu sama lain	

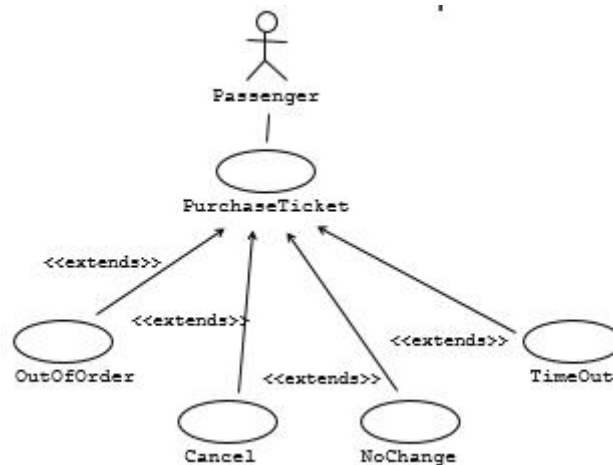
Generalization	hubungan taksonomi antara use case yang lebih umum dan use case yang lebih spesifik	
Extend	hubungan dari kasus penggunaan ekstensi ke kasus penggunaan dasar, menentukan bagaimana perilaku untuk kasus penggunaan ekstensi dapat dimasukkan ke dalam perilaku yang ditentukan untuk kasus penggunaan dasar	<pre><<extend>></pre> <pre>-----></pre>
Include	hubungan dari penggunaan ekstensi untuk kasus penggunaan inklusi, menentukan bagaimana perilaku untuk kasus penggunaan inklusi dimasukkan ke dalam behavior yang ditentukan untuk kasus penggunaan dasar	<pre>-----></pre> <pre><<include>></pre>

Contoh : Kegiatan pasien yang membuat janji



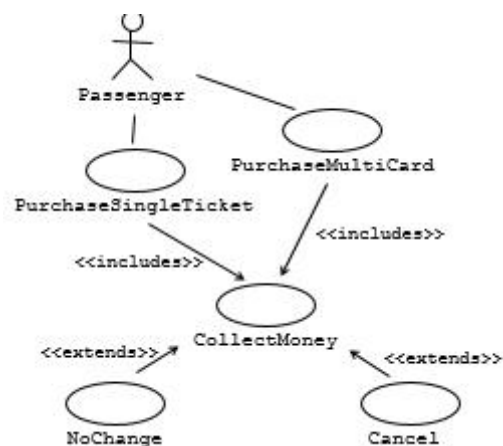
<<Extends>> Relationship

Extends relationship merepresentasikan pengecualian atau kasus yang jarang terjadi. Aliran kejadian pengecualian disebabkan karena proses utama mengalir sebagai bentuk klarifikasi. Use case dapat memiliki lebih dari 1 use case extend. Arah relasi extend diawali dari extend ke use case utama.



<<includes>> Relationship

Hubungan mewakili perilaku yang diperhitungkan dari use case. Perilaku diperhitungkan untuk digunakan kembali, bukan karena pengecualian. Arah hubungan <<includes>> adalah dengan use case (tidak seperti hubungan <<extends>>).



5 Class Diagram

- Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek.
- Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).
- Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain.

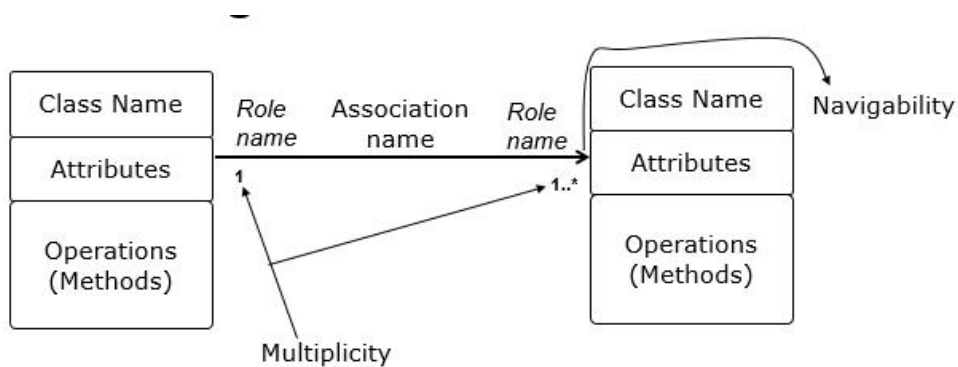
Class memiliki tiga area pokok :

- 1. Nama (dan stereotype)
- 2. Atribut
- 3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

- Private, tidak dapat dipanggil dari luar class yang bersangkutan
- Protected, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya
- Public, dapat dipanggil oleh siapa saja

Format Class Diagram dan Association

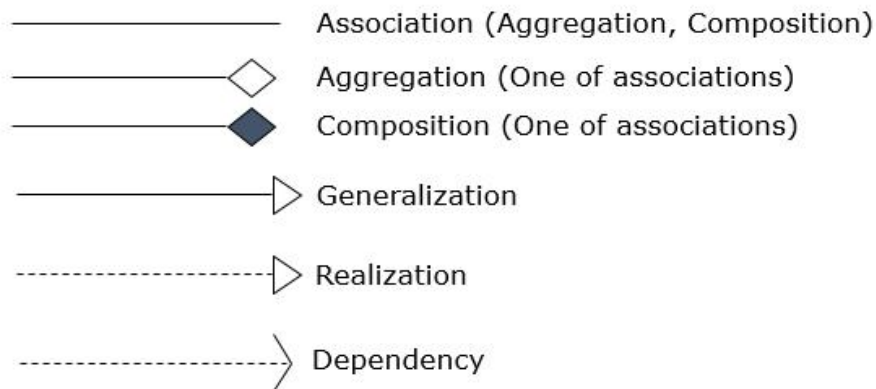


Multiplicity Notation

- 1 : One and only one
- 0..* : None or more
- 1..* : One or more
- 0..1 : None or one

Ada 3 tipe relationship:

1. Is-a (Generalization, Realization, Inheritance)
2. Has-a (Association)
3. Other (Association, Dependency)



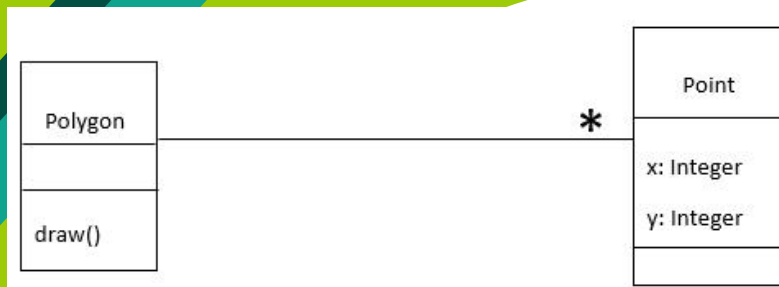
Multiplicity dari suatu titik association adalah angka kemungkinan bagian dari hubungan kelas dengan single instance (bagian) pada titik yang lain. Multiplicity berupa single number (angka tunggal) atau range number (angka batasan). Contoh, hanya bisa satu “customer” untuk setiap “order”, tetapi satu customer hanya bisa memiliki beberapa order.

Berikut tabel multiplicity

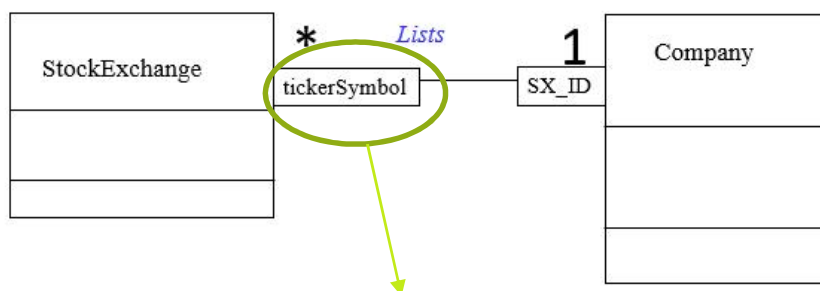
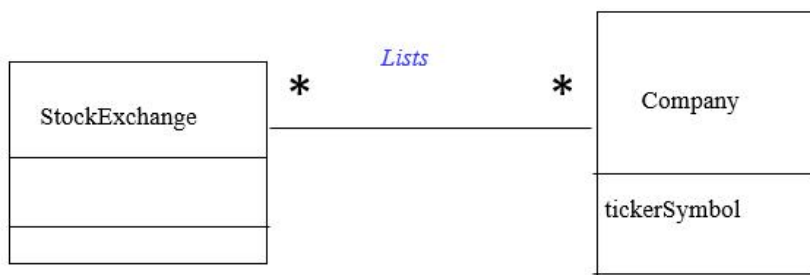
Multiplicities	Artinya
0...1	Nol atau satu bagian. N...m menerangkan n sampai m bagian
0...* or *	Tak hingga pada jangkauan bagian (Termasuk kosong)
1	Tepat satu bagian
1...*	Sedikitnya hanya satu bagian

Contoh 1-to-many associations





Many-to-many association

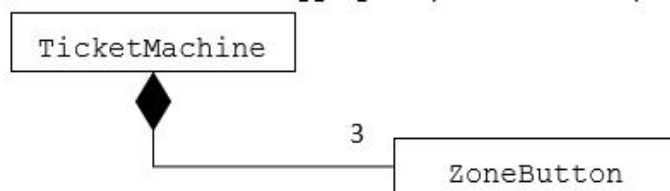


Qualifier

Kualifikasi dapat digunakan untuk mengurangi banyaknya asosiasi

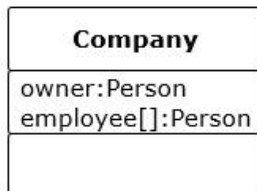
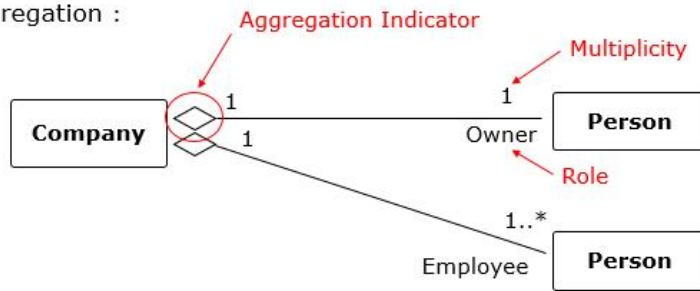
Agregation

Agregasi adalah kasus khusus asosiasi yang menunjukkan hierarki “terdiri dari”. Agregat adalah kelas induk, komponennya adalah kelas anak-anak. Solid diamond menunjukkan komposisi, bentuk agregasi yang kuat dimana komponen tidak dapat eksis tanpa agregat.



Contoh Class Diagram

Aggregation :

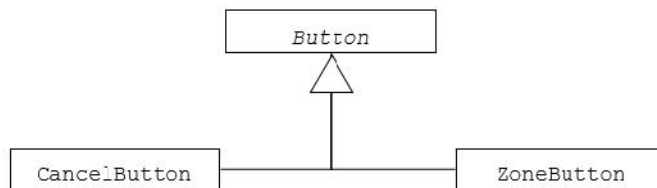


Detail of the class

Inheritance

Kelas anak mewarisi atribut dan operasi dari kelas induk. Inheritance menyederhanakan model dengan menghilangkan redundansi.

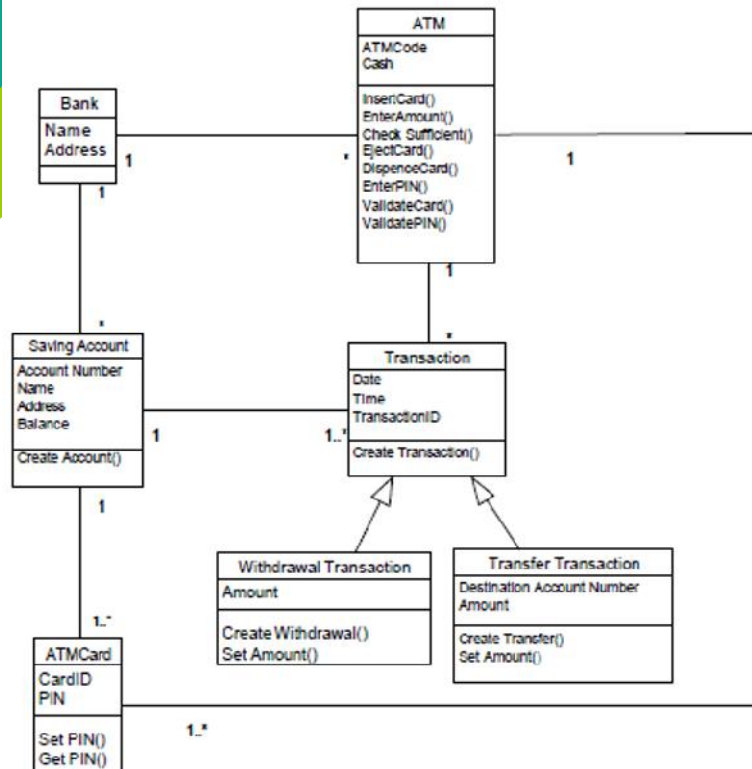
Contoh:



Langkah-langkah untuk membuat kelas

1. Identifikasi kandidat kelas dengan melihat atau mendata objek-objek dalam use case
2. Cari objek baru yang merupakan penamaan baru dari use case
3. Temukan asosiasi antar objek
4. Beri nama asosiasi
5. Tentukan multiplicity asosiasi

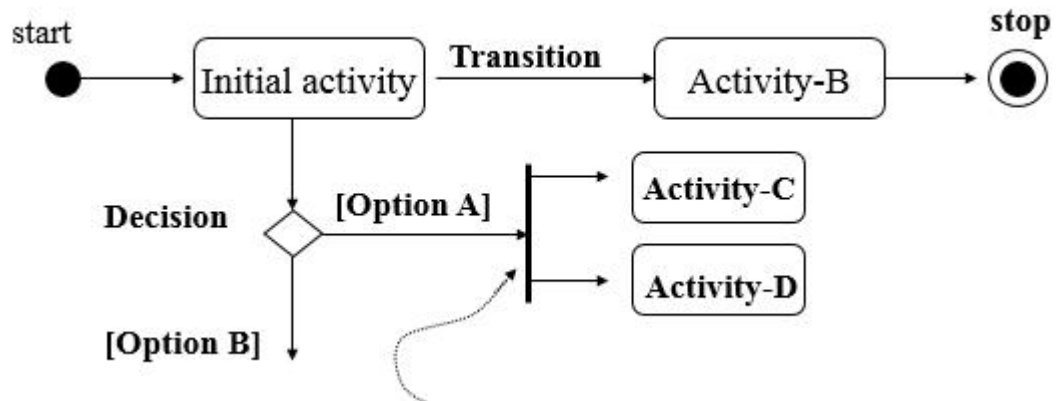
Contoh Class Diagram ATM



6. Activity Diagram

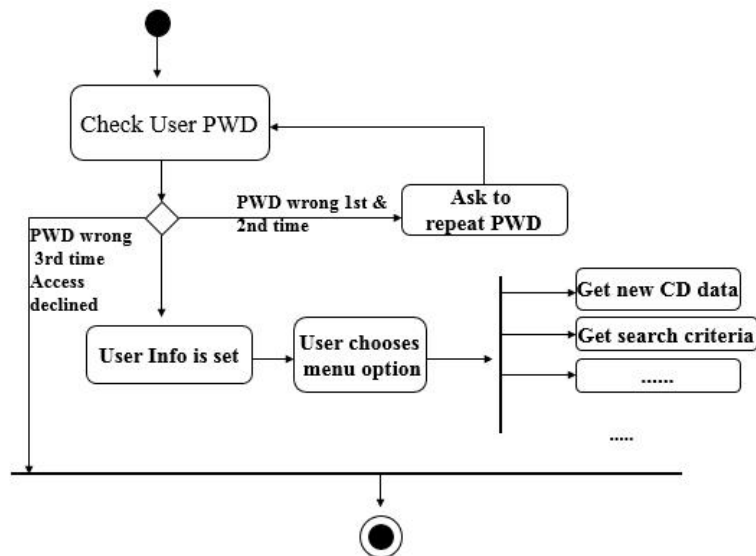
- Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir.
- Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.
- Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya state sebelumnya (internal processing).
- Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Format Activity Diagram

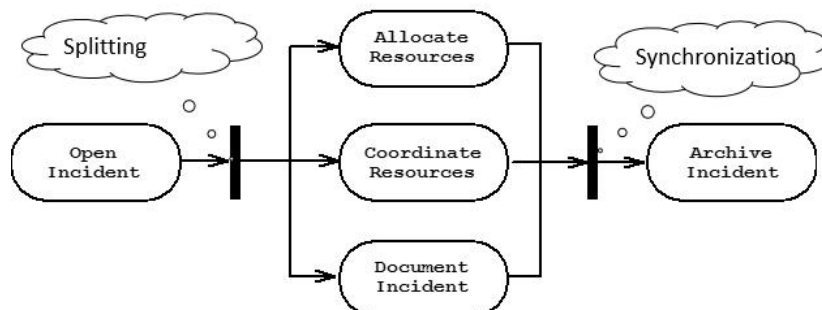


Bar menunjukkan bahwa satu aktivitas mengarah ke beberapa aktivitas yang terjadi secara paralel atau dalam urutan yang tidak dapat diprediksi.

Contoh Activity Diagram



Sinkronisasi beberapa aktivitas
Splitting aliran kontrol menjadi beberapa threads



6. Sequence Diagram

- Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. Sequence diagram terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).
- Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang men-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan.
- Sequence Diagram menggambarkan urutan interaksi antar objek ketika satu Use Case dieksekusi/dilakukan, sehingga Sequence Diagram (sebaiknya) sebanyak Use Case nya.
- Dalam membuat Sequence Diagram ini memakai metode programming MVC (Model-View-Controller) atau dalam istilah lain Model=Entity, View=Boundary, dan Controller=Control.
- View/Boundary adalah class yang berinteraksi langsung dengan Actor. Controller/Control adalah class interaksi perantara antara View/Boundary dan Model/Entity. sedang Model/Entity adalah class yang menyimpan data.

Tujuan membuat sequence Diagram adalah:

- Digunakan untuk memperlihatkan interaksi antar obyek dalam perintah yang berurut.
- Tujuan utama adalah mendefinisikan urutan kejadian yang dapat menghasilkan output yang diinginkan
- Mirip dengan activity diagram
 - Menggambarkan alur kejadian sebuah aktivitas
 - Lebih detail dalam menggambarkan aliran data, termasuk data atau behaviour yang dikirimkan atau diterima
 - Namun kurang mampu menjelaskan detail dari sebuah algoritma (loop, branching)

Komponen dalam Sequence Diagram

- Actor
- Interface (Boundary)
- Proses pembacaan (Control)
- Nama table (Entity)

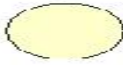
Simbol-simbol dalam Sequence Diagram

a. An Actor



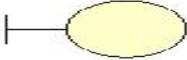
Menggambarkan orang yang sedang berinteraksi dengan sistem

b. Entity Class



Menggambarkan hubungan kegiatan yang akan dilakukan

c. Boundary Class



Menggambarakan sebuah penggambaran dari form

d. Control Class



Menggambarkan penghubung antara boundary dengan tabel

e. A focus Of Control & A life line



Menggambarkan tempat mulai dan berakhirnya sebuah message

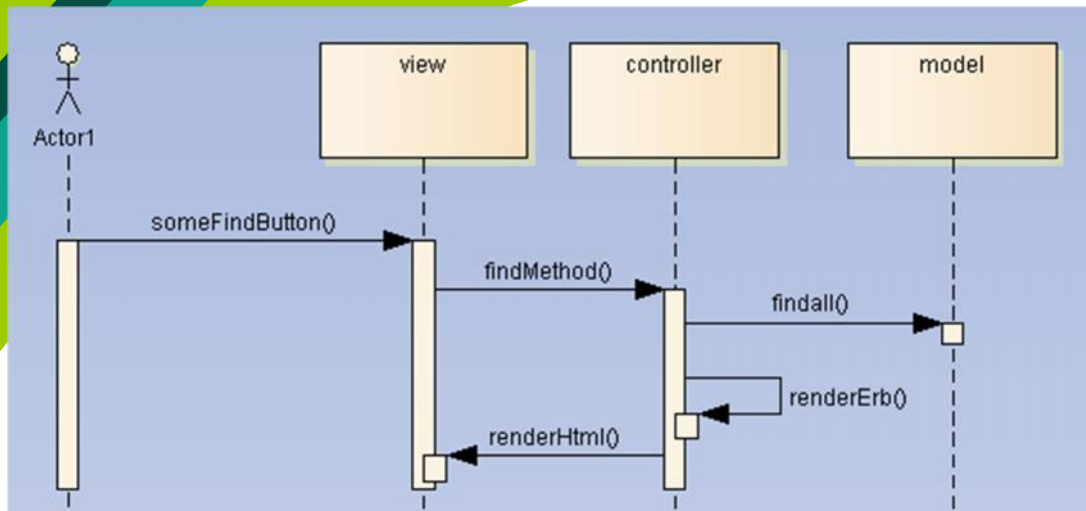
f. A message



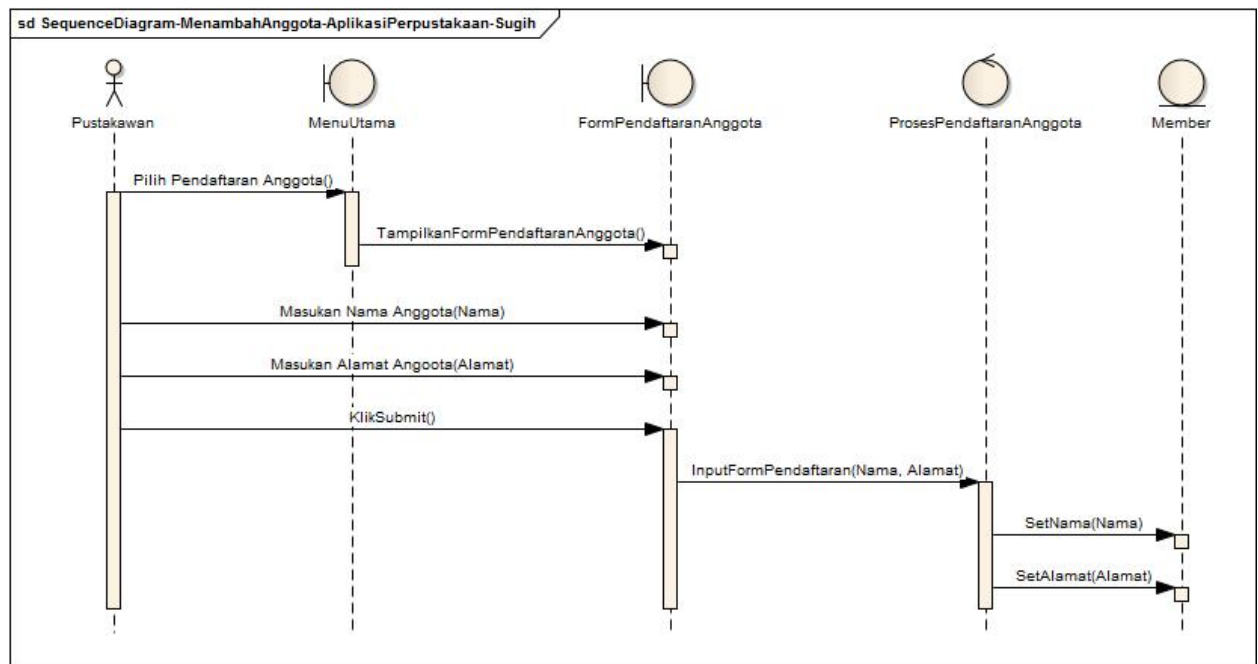
Menggambarkan Pengiriman Pesan

- Participant → Obyek yang terkait dengan sebuah urutan proses
- Lifeline → Menggambarkan daur hidup sebuah obyek
- Activation → Suatu titik waktu dimana sebuah obyek mulai berpartisipasi didalam sebuah sequence. Ditandai dengan sebuah bar
- Time → Elemen penting dalam sequence diagram. Konteksnya adalah **urutan, bukan durasi**
- Return → Suatu hasil kembalian sebuah operasi. Operasi mengembalikan hasil, tetapi boleh tidak ditulis

Sequence Diagram dengan konsep MVC



- Contoh: Sequence Diagram Menambahkan Anggota. Ada dua class Boundary yaitu MenuUtama dan FormPendaftaran Anggota, satu class Control yaitu ProsesPendaftaranAnggota, dan satu class Entity yaitu Member.



DAFTAR PUSTAKA

- Badan Pusat Statistik. 2007. Modul Diklat Fungsional Pranata Komputer Ahli. Cet. Kedua. Jakarta: Badan Pusat Statistik.
- Dennis, A., Wizom, B.H., and Tegarden, D. "Systems Analysis and Design", John Willey and Sons, Inc
- Jeffrey A. Hoffer, Joey F. George, Joseph S. Valacich, "Modern Systems Analysis & Design", second edition, Addison-Wesley, 1999.
- Jeffery L. Whitten, Lonnie D. Bentley, Kevin C. Dittman, "System Analysis and Design Methods", McGraw-Hill Companies, Inc., 2004.
- Pressman, R.S., "Software Engineering – A Practitioner's Approach", edisi kelima, McGraw-Hill Companies, Inc., 2001.
- Senn, J.A., "Analysis and Design of Information Systems", McGraw-Hill Companies, Inc., 1989.