

# **Analisis dan Design Berorientasi Objek (OOAD) dan Diagram Activity**

**2IA17**



**Nama Kelompok :**

ERAWAN FERNANDI (52412513)

FAUZI ALFANDRI (52412823)

GUSTI AYU PUTU PUTRI O. (53412219)

RENDY MATHIAS (56412121)

**UNIVERSITAS GUNADARMA**

**ATA 2013/2014**

## **PENDAHULUAN**

### **A. Latar Belakang**

OOAD(Object Oriented Analisis Design) adalah metode analisis yang memeriksa requirements pada suatu kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarah pada arsitektur Software yang didasarkan pada manipulasi objek-objek system atau subsitem.Dimana terdapat beberapa konsep dalam OOAD yaitu : kelas, object, metode, atribut, Atribut, Abstaksi,dll.

### **B. Rumusan Masalah**

1. Tehnik pemodelan apa saja yang ada di OOAD?
2. Jenis Jenis Diagram pada UML?
3. Macam-Macam Software UML?

### **C. Tujuan**

1. Menjelaskan tentang Pengertian,Konsep OOAD.
2. Menjelaskan tentang Diagram Activity.
3. Mengetahui tentang perbedaan dari masing masing Diagram pada UML

# **Analisis dan Design Berorientasi Objek**

## **(OOAD)**

### **1. Pengertian OOAD**

OOAD adalah metode analisis yang memeriksa requirements dari sudut pandang kelas-kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek system atau subsistem. OOAD merupakan cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara struktur data dan perilaku dalam satu entitas.

### **2. Konsep Dasar OOAD**

OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memeriksa requirement (syarat/keperluan) yang harus dipenuhi sebuah sistem) dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup perusahaan. Sedangkan OOD adalah metode untuk mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek sistem atau subsistem.

Terdapat beberapa konsep dalam OOAD, yaitu :

#### **1. Kelas (Class)**

kumpulan objek-objek dengan karakter yang sama. Sebuah kelas mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (relationship) dan arti. Suatu kelas dapat diturunkan dari kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru. Kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek.

## **2. Objek (Object)**

Abstraksi dan sesuatu yang mewakili dunia nyata. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan.

## **3. Metode (Method)**

operasi atau metode pada kelas hampir sama dengan fungsi atau prosedur pada metodologi terstruktur. Operasi merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan objek.

## **4. Atribut**

variabel global yang dimiliki kelas. Atribut dapat berupa nilai atau elemen – elemen data yang dimiliki oleh objek dalam kelas. Atribut dipunyai secara individu oleh suatu objek misalnya berat, jenis.

## **5. Abstraksi**

prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi suatu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan masalah.

## **6. Enkapsulasi**

pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

## **7. Pewarisan**

mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh dan objek lain sebagai bagian dari dirinya.

## **8. Antar muka**

biasanya digunakan agar kelas yang lain tidak mengakses langsung ke suatu kelas.

### **9. Reusability**

pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

### **10. Generalisasi & spesialisasi**

menunjukkan hubungan antar kelas dan objek yang umum dengan kelas dan objek yang khusus.

### **11. Komunikasi Antar Objek**

dilakukan lewat pesan (message) yang dikirim dari satu objek ke objek lainnya.

### **12. Polimorfisme**

kemampuan suatu objek untuk digunakan di banyak tujuan yang berbeda dengan nama yang sama sehingga menghemat program.

### **13. Package**

sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompok kelas-kelas yang bernama sama disimpan dalam package yang berbeda.

## **3. Metodologi pengembangan sistem berbasis object**

Metodologi adalah cara systematis untuk mengerjakan analisis and design. Dengan metodologi, pihak yang membangun system software dapat merencanakan dan mengulangi pekerjaan dilain waktu. Metodologi juga menghilangkan perbedaan notasi untuk suatu hal yang sama karena setiap orang akan berbicara dalam bahasa yang sama. Metodologi yang paling banyak dalam OOAD, yaitu : Object Modeling Technique (OMT) dari Rumbaugh, Object Oriented Booch, Responsibility-Driven Design/ Class Responsibility Collaboration (RDD/CRC) dari Wirf-Broock, Metodologi Coad/ Yourdan dan Jacobson Object Oriented Software Engineering (OOSE).

### **1) Object Modeling Technique (OMT)**

Dikembangkan oleh James Rumbaugh sebagai metode untuk mengembangkan sistem berorientasi objek dan untuk mendukung pemograman berorientasi objek

### **2) Object Oriented Booch**

Dikembangkan oleh Grady Booch terdiri dari diagram kelas,objek,transisi status, interaksi, modul dan proses.

### **3) Class Responsibility Collaboration (CRC)**

Merupakan bagian dari Object-Oriented Programming, System, Languages And Application(OOPSLA). Dibuat untuk menjadi kelas yang akan dianalisis.

### **4) Metodologi Coad/ Yourdan**

Menyediakan sebuah diagram kelas, pembuatannya dengan langkah-langkah berikut :

1. Mendefenisikan kelas dan objek
2. Mengidentifikasi struktur kelas dan objek.
3. Mendefenisikan subjek nama kelas.
4. Mendefenisikan atribut.
5. Mendefenisikan operasi/layanan (service).

### **5) Object Oriented Software Engineering (OOSE)**

Dikembangkan oleh Ivar Jacobson adalah metode disain berorientasi objek yang melibatkan use case.

## **4. Teknik pemodelan yang ada pada OOAD**

### **a) Model Objek :**

- Model objek Menggambarkan struktur statis dari suatu objek dalam sistem dan relasinya

- Model objek berisi diagram objek. Diagram objek adalah graph dimana nodenya adalah kelas yang mempunyai relasi antar kelas.

b) Model Dinamik

- Model dinamik menggambarkan aspek dari sistem yang berubah setiap saat.
- Model dinamik dipergunakan untuk menyatakan aspek kontrol dari sistem.
- Model dinamik berisi state diagram. State diagram adalah graph dimana nodenya adalah state dan arc adalah transisi antara state yang disebabkan oleh event.

c) Model Fungsional

- Model fungsional menggambarkan transformasi nilai data di dalam sistem.
- Model fungsional berisi data flow diagram. DFD adalah suatu *graph* dimana *nodenya* menyatakan proses dan *arcnya* adalah aliran data.

## **Unified Modelling Language (UML)**

### **A. Definisi UML**

Unified Modeling Language (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yangterkait dengan objek (Whitten L. Jeffery et al, 2004). Sementara menurut Henderi (2007: 4) Unified Modeling Language (UML) adalah sebuah bahasa pemodelan yang telah menjadistandar dalam industri software untuk visualisasi, merancang, dan mendokumentasikansistem perangkat lunak. Bahasa Pemodelan UML lebih cocok untuk pembuatan perangkatlunak dalam bahasa pemrograman berorientasi objek (C++, Java, VB.NET), namundemikian 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 denganuse case dan actor
2. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuatdengan 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 developmentdiagrams
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



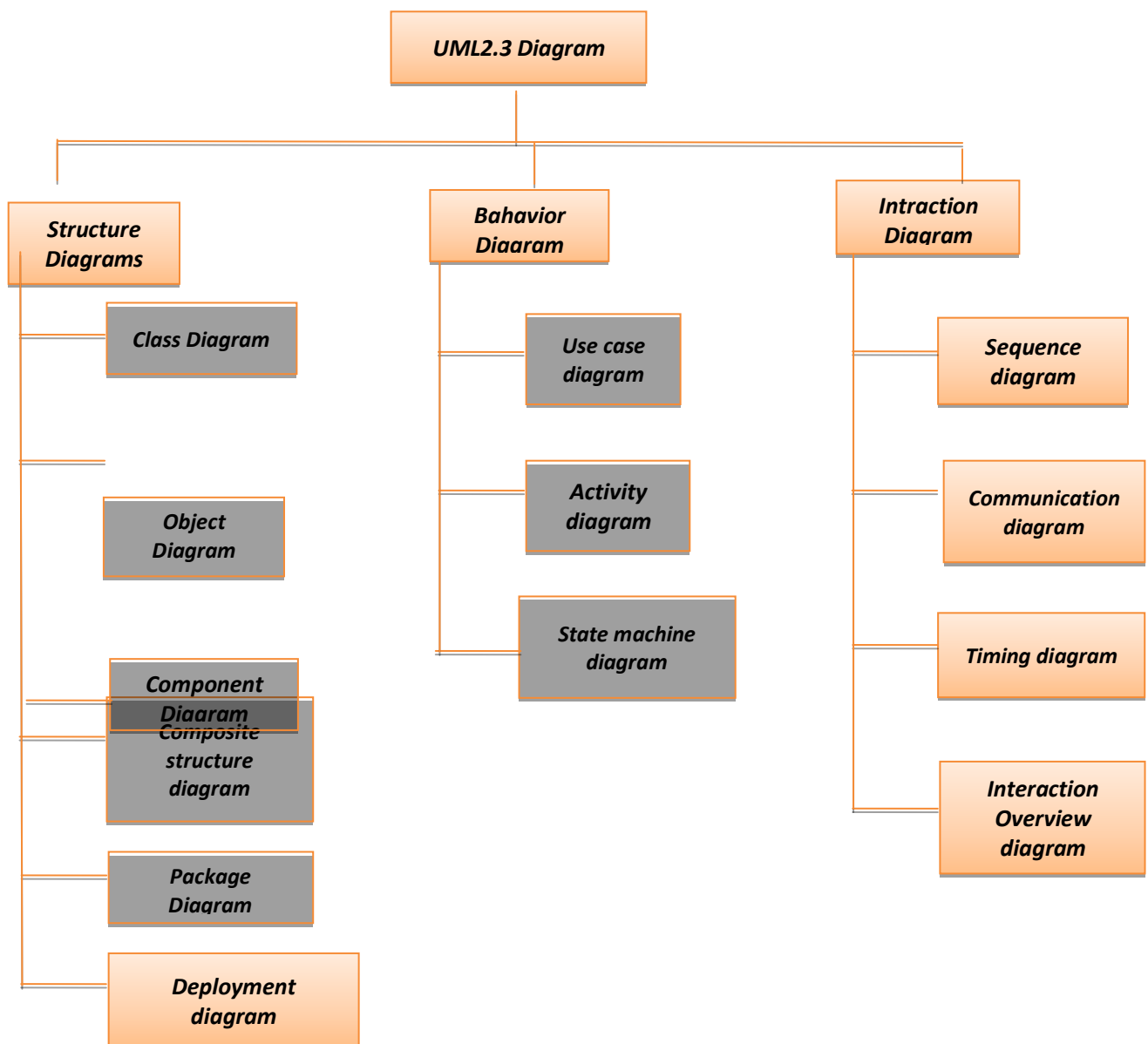
mengkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadidiagram. UML mempunyai banyak diagram yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun. Diagram-diagram tersebut digunakan untuk (Henderi et al, 2008:71):

1. Mengkomunikasikan ide2.
2. Melahirkan ide-ide baru dan peluang-peluang baru
3. Menguji ide dan membuat prediksi
4. Memahami struktur dan relasi-relasinya

## **B. Konsep Pemodelan Menggunakan Unified Modeling Language (UML)**

Pemodelan menggunakan Unified Modeling Language (UML) merupakan metode pemodelan berorientasi objek dan berbasis visual. Karenanya pemodelan menggunakan UML merupakan pemodelan objek yang fokus pada pendefinisian struktur statis dan modelsistem 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.

### C. Diagram Dasar dalam Unified Modeling Language (UML)



Penjelasan :

#### 1. Structure Diagram

➔ kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

Dalama Structure diagram terdapat beberapa jenis model diagram lagi, yaitu :

### a. Class Diagram

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

### b. Object Diagram

*Object* diagram merupakan sebuah gambaran tentang objek-objek dalam sebuah sistem pada satu titik waktu. Karena lebih menonjolkan perintah-perintah 29 daripada *class*, *object* diagram lebih sering disebut sebagai sebuah diagram perintah.

### c. Component Diagram

*Component diagram* menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan ( *dependency* ) di antaranya. Komponen piranti lunak adalah modul berisi *code* , baik berisi *source code* maupun *binary code* , baik *library* maupun *executable* , baik yang muncul pada *compile time*, *link time* , maupun *run time*.

### d. Composite Structure Diagram

Diagram struktur komposit adalah diagram yang menunjukkan struktur internal classifier, termasuk poin

interaksinya ke bagian lain dari sistem. Hal ini menunjukkan konfigurasi dan hubungan bagian, yang bersama-sama melakukan perilaku classifie. Diagram struktur komposit merupakan jenis diagram struktur statis dalam Unified Modeling Language (UML), yang menggambarkan struktur internal kelas dan kolaborasi.

#### e. Package Diagram

Diagram objek melengkapi notasi grafik untuk pemodelan objek, kelas dan relasinya dengan yang lain. Diagram objek bermanfaat untuk pemodelan abstrak dan membuat perancangan program. Untuk mengatur pengorganisasian diagram Class yang *kompleks*, dapat dilakukan pengelompokan kelas-kelas berupa *package* (paket-paket). *Package* adalah kumpulan elemen-elemen logika UML.

#### f. Deployment Diagram

*Deployment/physical diagram* menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men- *deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.

## 2. Behavior Diagram

➔ yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan pada sebuah sistem.

#### a. Use Case Diagram

Use case adalah abstraksi dari interaksi antara system dan actor. Use case bekerja dengan cara

mendeskripsikan tipe interaksi antara user sebuah system dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah system dipakai.

**b. Activity Diagram**

Activity diagram memiliki pengertian yaitu lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Memiliki struktur diagram yang mirip flowchart atau data flow diagram pada perancangan terstruktur.

**c. State Machine Diagram**

*Statechart diagram* menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima.

**3. Interaction Diagram**

➔ yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar sub sistem pada suatu sistem.

**a. Sequence Diagram**

Menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Dalam menggambarkan diagram ini harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode- metode yang dimiliki kelas.

**b. Communication Diagram**

Merupakan penyederhanaan dari Diagram Kolaborasi . Menggambarkan interaksi antar objek/bagian dalam bentuk urutan pengiriman pesan. Diagram ini

merepresentasikan informasi yang diperoleh dari diagram kelas. Dalam Diagram ini yang dituliskan adalah operasi/metode yang dijalankan antar objek.

**c. Timing Diagram**

*Timing* Diagram adalah bentuk lain dari *interaction* diagram, dimana fokus utamanya lebih ke waktu. Timing diagram sangat berdaya guna dalam menunjukkan faktor pembatas waktu diantara perubahan *state* pada objek yang berbeda.

**d. Interaction Overview Diagram**

Bentuk Diagram Aktivitas yang setiap titik merepresentasikan diagram interaksi. Notasi pada Interaction Overview Diagram sama dengan Diagram Aktivitas.

## **Diagram Activity**



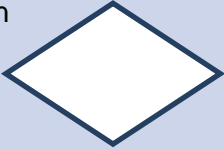
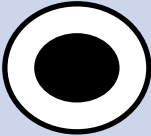
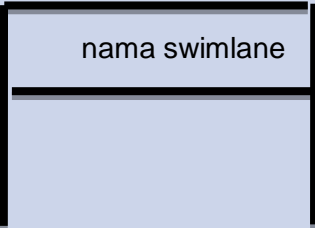

### **A. Pengertian**

Activity diagram memiliki pengertian yaitu lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis. Memiliki struktur diagram yang mirip flowchart atau data flow diagram pada perancangan terstruktur. Memiliki pula manfaat yaitu apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan. Dan activity dibuat berdasarkan sebuah atau beberapa use case pada use case diagram.

### **B. Kegunaan**

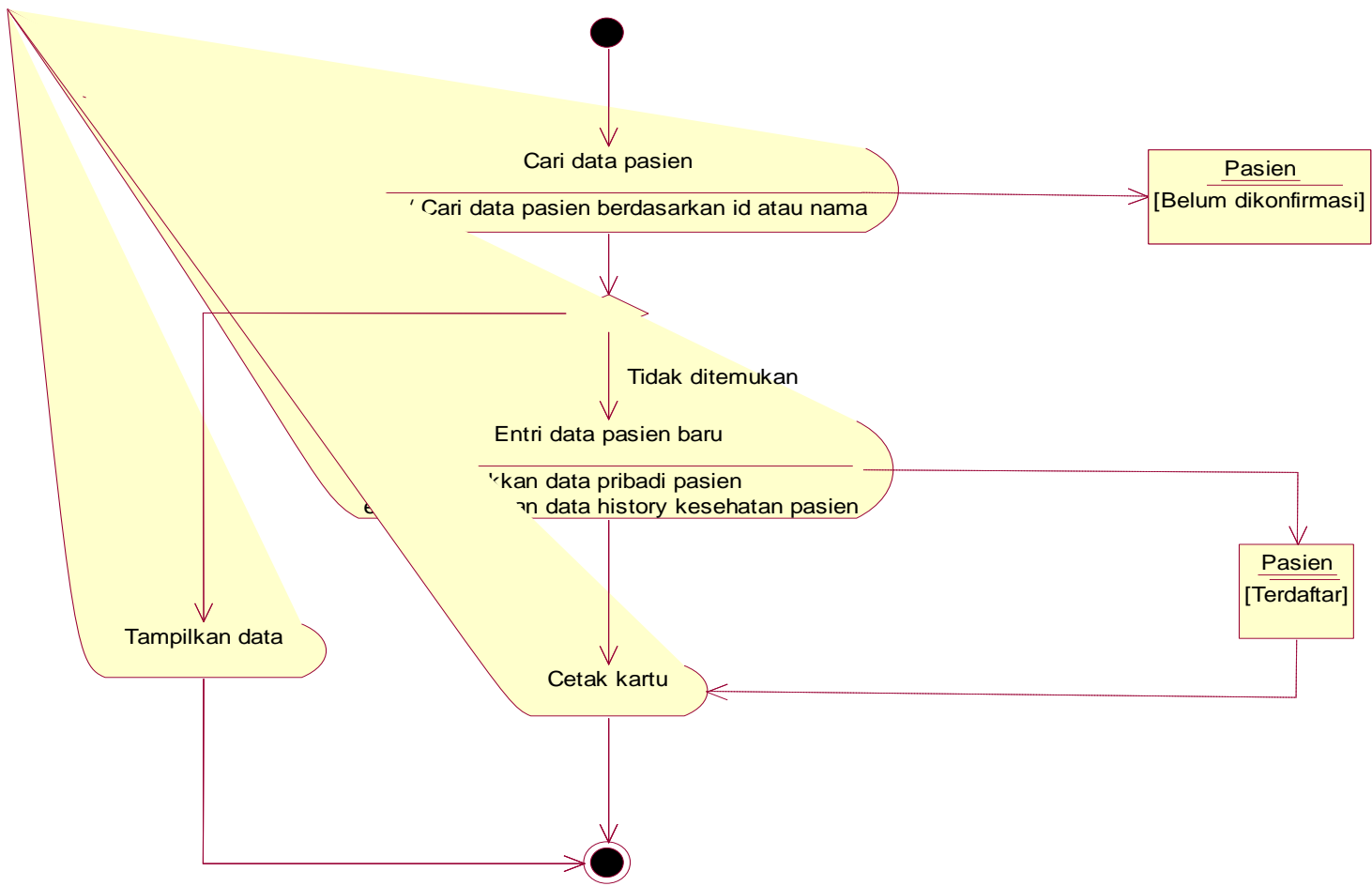
1. Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses
2. Dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis
3. Struktur diagram ini mirip flowchart atau Data Flow Diagram pada perancangan terstruktur
4. Sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan
5. Activity diagram dibuat berdasarkan sebuah atau beberapa use case pada use case diagram

### C. Simbol-simbol pada diagram activity

Simbol	Deskripsi
Initialstate 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Activity 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / decision 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / Join	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Finalstate 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi



#### D. Contoh Diagram Activity



## **Kesimpulan**

OOAD adalah metode analisis yang memeriksa requirements dari sudut pandang kelas kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek system atau subsistem.

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. Salah satu diagram UML adalah diagram activity. Diagram ini mirip dengan flowchart, diagram activity juga bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan.

## Daftar Pustaka

<http://siulukadola.blogspot.com/2012/02/simbol-diagram.html>

[tetehtifa.blogspot.com/2013/01/activity-diagram.html](http://tetehtifa.blogspot.com/2013/01/activity-diagram.html)

<http://bopungumn.blogspot.com/2012/03/activity-diagram-salahsatu-cara-untuk.html>

<http://asrielpangloli.blogspot.com/2013/06/penjelasan-tentang-activity-diagram.html>

[http://noviastutik.blogspot.com/2012/09/diagram-diagram-dalam-uml-unified\\_24.html](http://noviastutik.blogspot.com/2012/09/diagram-diagram-dalam-uml-unified_24.html)

[www.academia.edu/1831182/Object\\_Oriented\\_Analysis\\_and\\_Design](http://www.academia.edu/1831182/Object_Oriented_Analysis_and_Design)

[trinadi-putra.blogspot.com/2012/04/materi-uml.html](http://trinadi-putra.blogspot.com/2012/04/materi-uml.html)

[catatankuliah.blogspot.com/2012/06/object-oriented-analysis-and-design.html](http://catatankuliah.blogspot.com/2012/06/object-oriented-analysis-and-design.html)

[saiiamilla.wordpress.com/2010/06/04/oad-object-oriented-analysis-dan-design/](http://saiiamilla.wordpress.com/2010/06/04/oad-object-oriented-analysis-dan-design/)

<http://warbrain-hkr.blogspot.com/2013/03/a.html>

[dhienzzworld.wordpress.com/2013/03/29/uml-analisis-perancangan-berorientasi-objek/](http://dhienzzworld.wordpress.com/2013/03/29/uml-analisis-perancangan-berorientasi-objek/)

[dedepurple.blogspot.com/2012/09/analisis-dan-desain-berorientasi-objek.html](http://dedepurple.blogspot.com/2012/09/analisis-dan-desain-berorientasi-objek.html)

<http://blog.iwankiddy.net/2013/tool-untuk-membuat-uml-modeling.rna>