Diagram Class, Diagram Objek Diagram Component dan Deployment

Citra Noviyasari, S.Si, MT
SI - UNIKOM

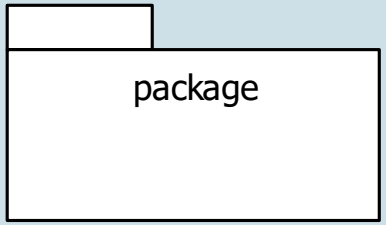
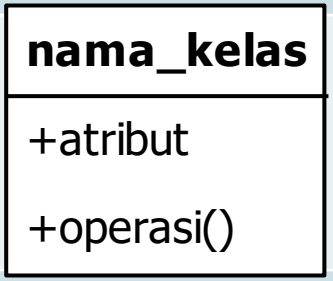
Definisi

- Diagram class 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.

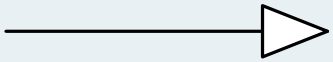
Komponen Kelas

- Objek adalah abstraksi dari sebuah entitas nyata atau tidak nyata yang informasinya harus diingat atau disimpan.
- Class adalah deskripsi lebih dari satu atau lebih objek dengan sejumlah atribut dan layanan yang sama termasuk deskripsi tentang cara membuat objek dari kelas tersebut.
- Atribut adalah variable data, yang dapat memberikan informasi keadaan dimana tiap objek dari suatu kelas mempunya nilai tersendiri. Atribut juga merupakan penjelasan dari item data
- Metoda adalah prosedur atau fungsi yang menjadi perilaku kelas dan objek dan menjadi tanggung jawab objek tersebut.

NOTASI

Deskripsi	Notasi
package merupakan sebuah bungkusan dari satu atau lebih kelas	- 
kelas pada struktur sistem	
sama dengan konsep *interface* dalam pemrograman berorientasi objek	 **nama_interface**
relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*	———————

NOTASI

Deskripsi	Notasi
relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*	
relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)	
relasi antar kelas dengan makna kebergantungan antar kelas	

Hubungan Antar Class

- **Association**

  Hubungan statis antar class. Pada umumnya menggambarkan class yang memiliki atribut berupa class lain, atau class yang harus mengetahui eksistensi class lain.

- ***Agregation***

  Hubungan secara keseluruhan antara aggregate class dengan component class.

- ***Inheritance dan Generalization***

  Inheritance adalah hubungan hirarkis antar class. Class dapat diturunkan dari class lain dan mewarisi semua atribut dan metode class asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari class yang mewarisinya. Kebalikan dari inheritance adalah Generalitation yang merupakan hubungan taksonomi antara class yang lebih umum dengan class yang lebih khusus.

- ***Hubungan dinamis***

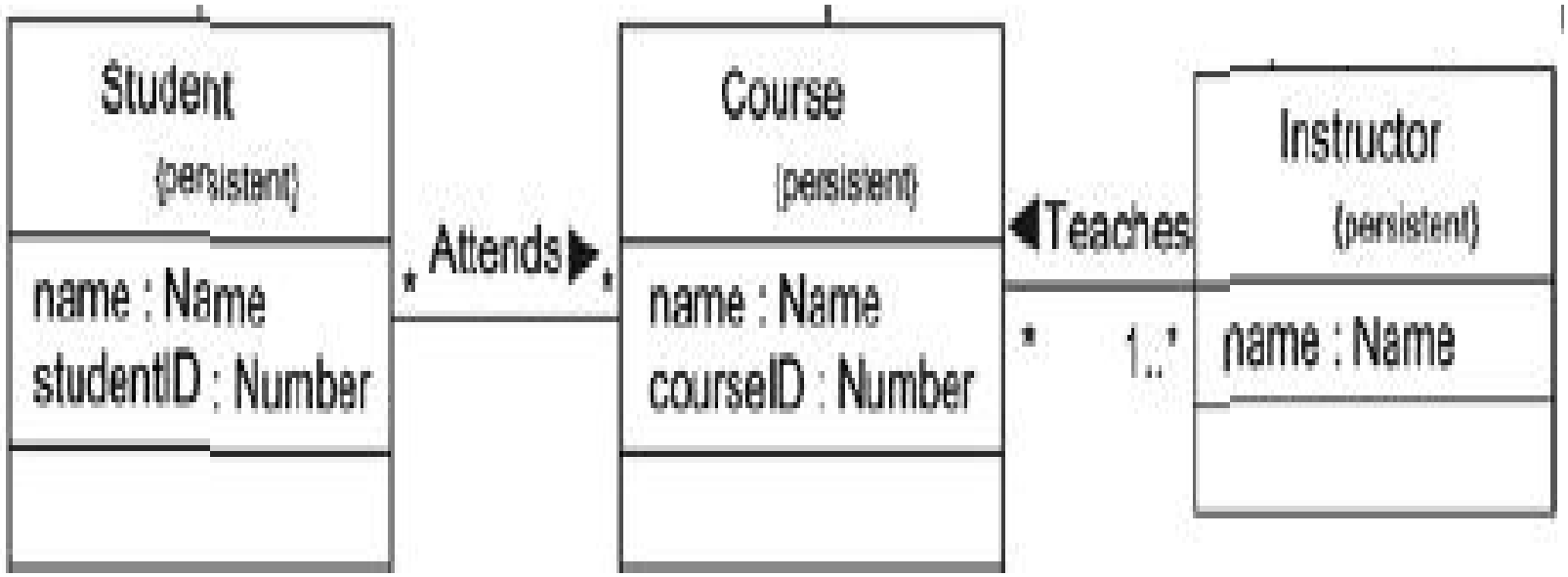
  Rangkaian pesan yang dikirim dari satu class kepada class lainnya. Hubungan dinamis dapat digambarkan dengan menggunakan sequence diagram.

Association

- Menggambarkan hubungan antar class dengan ditandai dengan anak panah dan seringkali ditambahkan label dan multiplicity untuk memperjelas hubungan

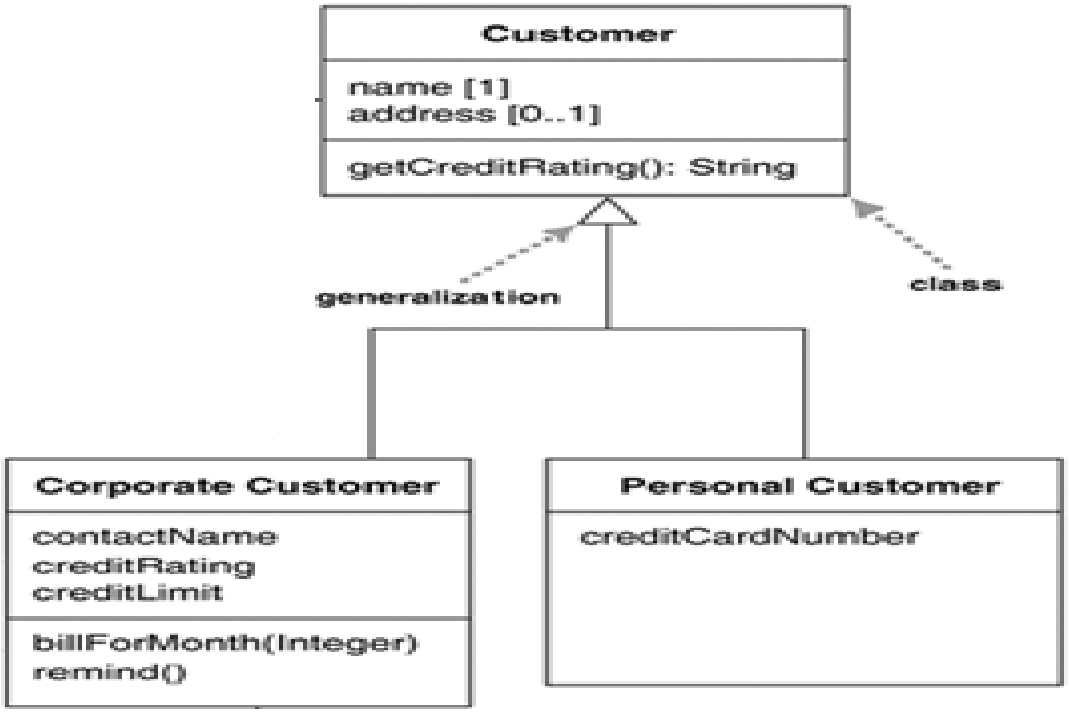
Multiplicity	Arti
N (default)	Banyak
0..0	Nol
0..1	Nol atau satu
0..n	Nol atau banyak
1..1	Tepat satu
1..n	Satu atau banyak

Association



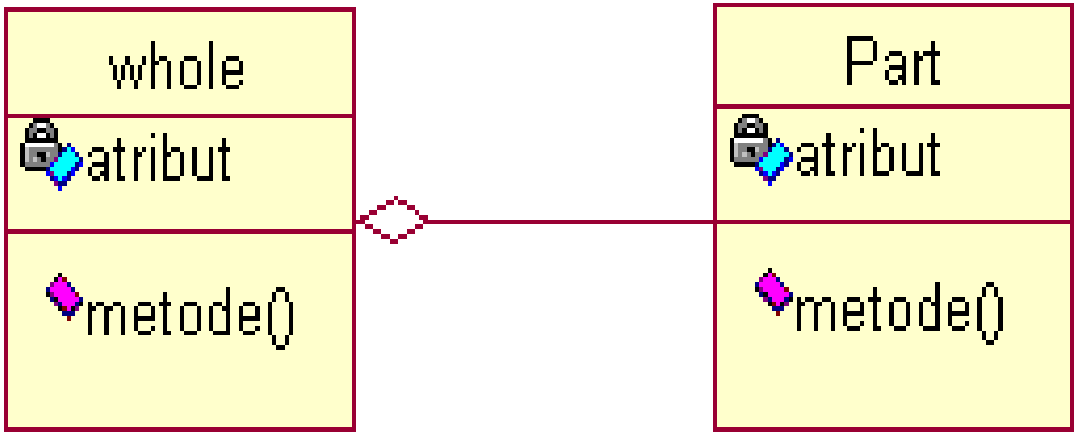
Generalization

- Generalization adalah inheritance pada UML dimana sub class mewarisi feature dari super classnya. Sub class mampu overriding metode super classnya. Generalization dinotasikan dengan anak panah mengacu ke super class.



Agregasi

- Sebuah aggregation adalah bentuk khusus association yang memodelkan hubungan whole-part antara sebuah aggregation dengan bagiannya.

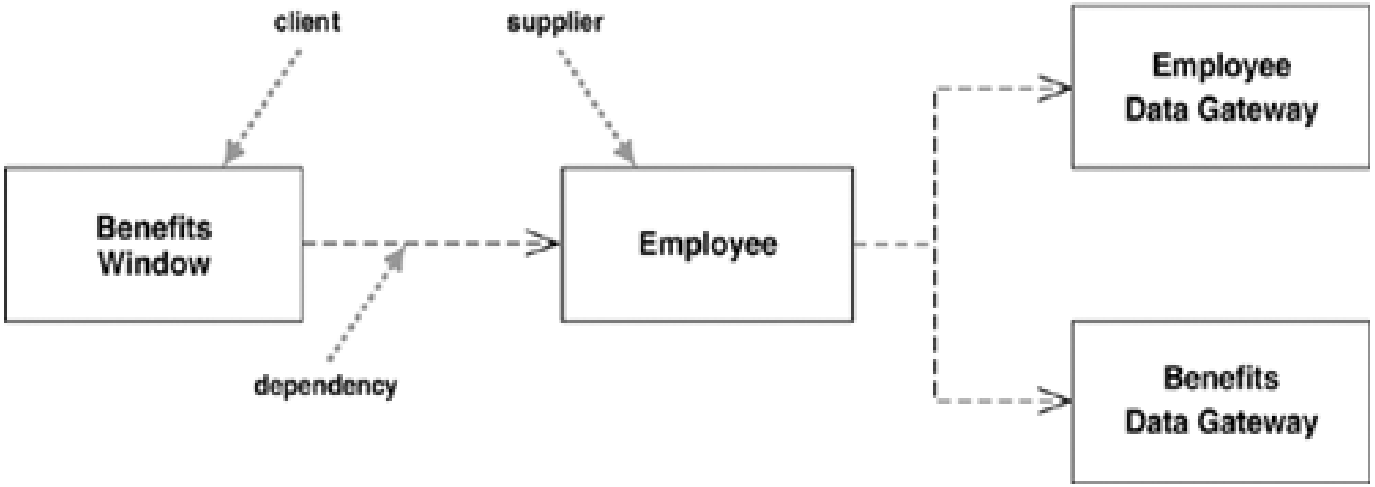


Composition

- Merupakan relasi Whole-part yang tidak boleh dipisahkan
- Ilustrasi :
  - Class Person dan Class Head
  - Menghapus person berarti juga menghapus kepalanya dan orang tidak bisa hidup tanpa kepala
  - Orang dan Kepala harus ada bersamaan

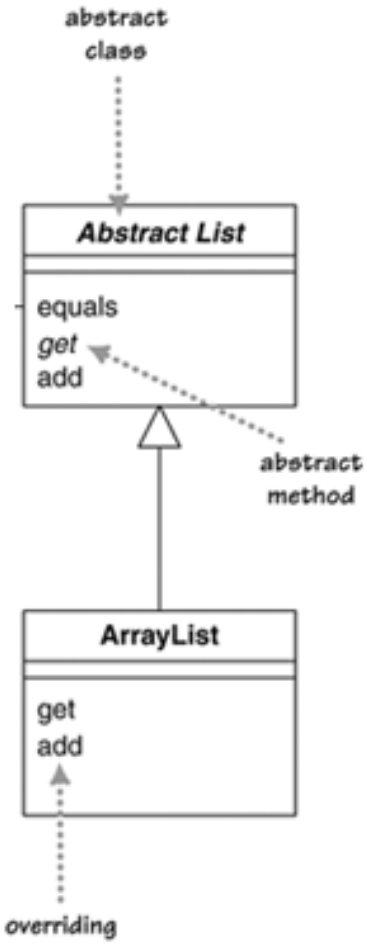
Depedency

- Dependency adalah perubahan pada salah satu elemen yang mengakibatkan perubahan pada elemen yang lain. Semakin kompleks sistem, maka dependency menjadi sesuatu yang harus dipertimbangkan. Dependency hanya berlaku satu arah. Bisa diperjelas dengan penggunaan keyword, seperti <<parameter>>, <<use>>, <<call>> dengan notasi anak panah dan garis putus-putus. Aturan umum dependency adalah mengurangi dependency antar modul (low coupling).



Abstract Class

Abstract Class digunakan pada class yang tidak bisa diinstantiasi, harus diturunkan kedalam class non-abstract. Memiliki satu atau lebih metode abstract sedangkan metode abstract tidak memiliki implementasi. Implementasi dilakukan oleh class yang menurunkan. Dinotasikan italics pada nama.



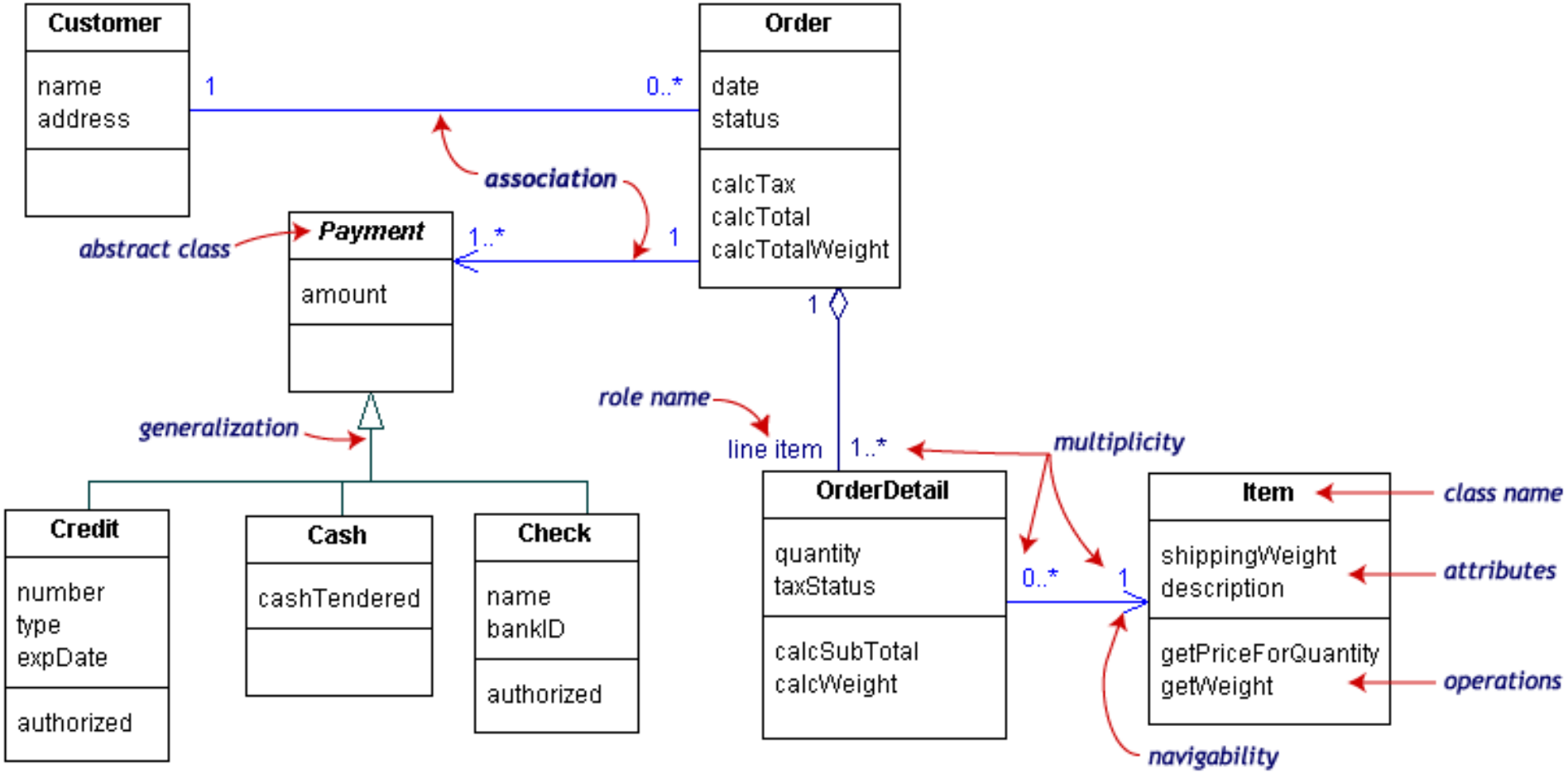
Sifat Atribut dan Metoda (Visibility)

- ***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


Customer
name
address
1
0..*
Order
date
status
calcTax
calcTotal
calcTotalWeight
association
abstract class
Payment
amount
1..*
1
1
role name
line item
1..*
multiplicity
generalization
Credit
number
type
expDate
authorized
Cash
cashTendered
Check
name
bankID
authorized
OrderDetail
quantity
taxStatus
calcSubTotal
calcWeight
0..*
1
Item
shippingWeight
description
getPriceForQuantity
getWeight
class name
attributes
operations
navigability

Contoh

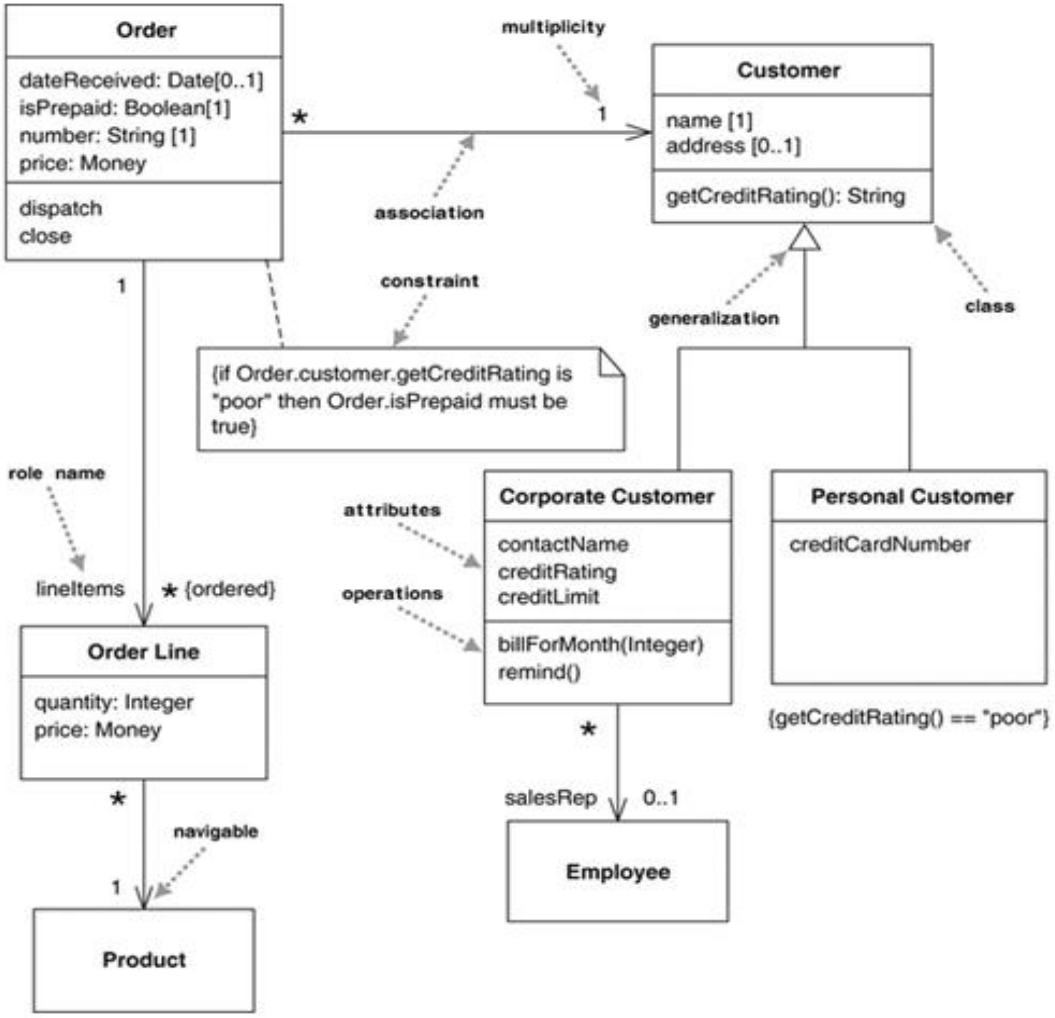
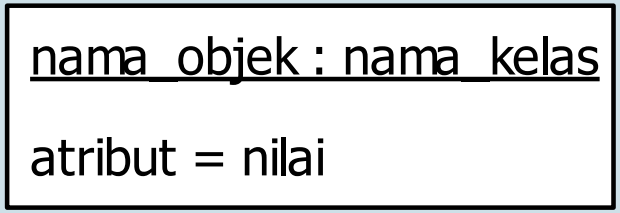

Order
dateReceived: Date[0..1]
isPrepaid: Boolean[1]
number: String [1]
price: Money
dispatch
close
multiplicity
1
*
association
Customer
name [1]
address [0..1]
getCreditRating(): String
1
constraint
generalization
class
{if Order.customer.getCreditRating is "poor" then Order.isPrepaid must be true}
role name
lineItems
* {ordered}
attributes
operations
Corporate Customer
contactName
creditRating
creditLimit
billForMonth(Integer)
remind()
Personal Customer
creditCardNumber
{getCreditRating() == "poor"}
*
salesRep
0..1
Employee
Order Line
quantity: Integer
price: Money
*
navigable
1
Product

Diagram Object

Diagram Objek

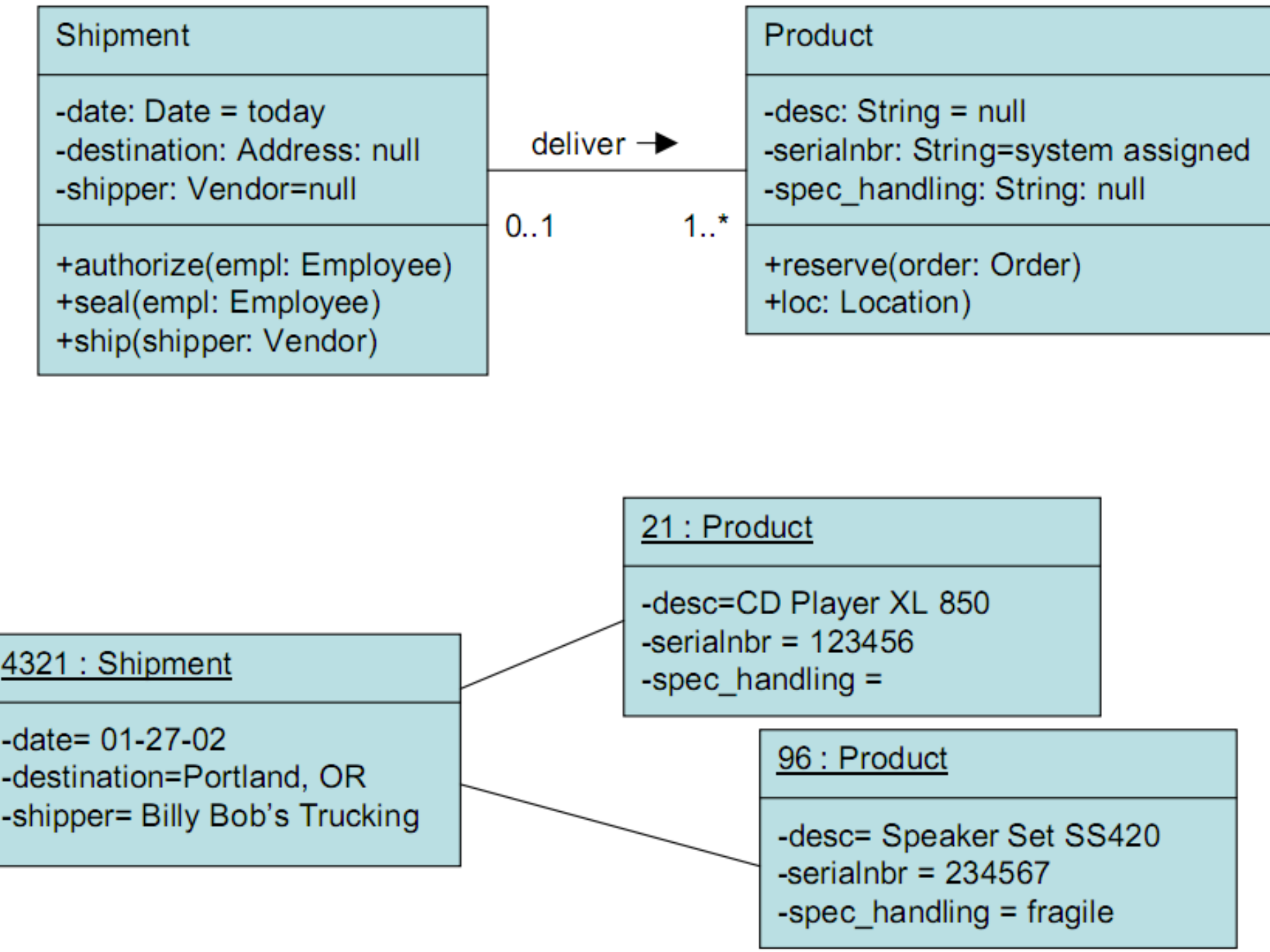
- Objek diagram berasal dari kelas objek diagram diagram sehingga tergantung pada diagram kelas.
- Konsep-konsep dasar serupa untuk kelas objek diagram dan diagram. Obyek diagram juga mewakili pandangan statis dari sebuah sistem tetapi pandangan statis ini merupakan sebuah snapshot dari sistem pada saat tertentu.
- Object diagram digunakan untuk membuat satu set benda dan hubungan mereka sebagai contoh.

NOTASI

Deskripsi	Notasi
objek dari kelas yang berjalan saat sistem dijalankan	
relasi antar objek	

Class Diagram vs Object Diagram

- Class mendefinisikan rule, object mendefinisikan fakta-fakta
- Class mendefinisikan “what can be”, object mendeskripsikan “what is”
- Keduanya membentuk object model
- Kegunaan Class :
  - terutama sebagai alat research dan testing
  - untuk memahami masalah dengan mendokumentasikan contoh-contoh dari problem domain sebagai object diagram
  - saat analisis & perancangan untuk memverifikasi keakuratan class diagram


Shipment
-date: Date = today
-destination: Address: null
-shipper: Vendor=null
+authorize(empl: Employee)
+seal(empl: Employee)
+ship(shipper: Vendor)
deliver
0..1
1..*
Product
-desc: String = null
-serialnbr: String=system assigned
-spec_handling: String: null
+reserve(order: Order)
+loc: Location)
21 : Product
-desc=CD Player XL 850
-serialnbr = 123456
-spec_handling =
4321 : Shipment
-date= 01-27-02
-destination=Portland, OR
-shipper= Billy Bob's Trucking
96 : Product
-desc= Speaker Set SS420
-serialnbr = 234567
-spec_handling = fragile

Perbedaan Class dan Component

- Kelas mempresentasikan abstraksi logic, komponen dapat merepresentasikan sesuatu yang fisik. Komponen dapat tinggal di node, sementara kelas tidak.
- Komponen merepresentasikan pemaketan fisik dari komponen logic lain dan pda beberapa level abtraksi.
- Kelas dapat mempunyai atribut dan operasi secara langsung. Komponen hanya mempunyai operasi yang dapat dicapai hanya lewat antarmukanya.

Diagram Component

Definisi

- Component diagram menggambarkan struktur dan hubungan antar komponen peranti lunak, termasuk ketergantungan (dependency) diantaranya.
- Komponen peranti 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.
- Pada umumnya komponen terbentuk dari bebrapa class dan/atau package, tapi dapat juga dari komponen-komponen yang lebih kecil.
- Komponen dapat juga berupa interface, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

3 Jenis Component

- Komponen deployment

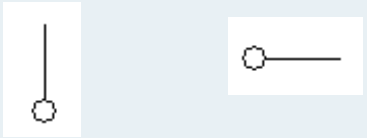
  Komponen yang diperlukan untuk membentuk system yang dapat dieksekusi, seperti DLL (Dynamic Link Library) dan executable (exe). Dapat juga untuk model seperti COM+, COBRA, Enterprise Java Beans, juga halaman web dinamis, dan sebagainya.
- Komponen produk, seperti file kode sumber dan file data.
- Komponen eksekusi

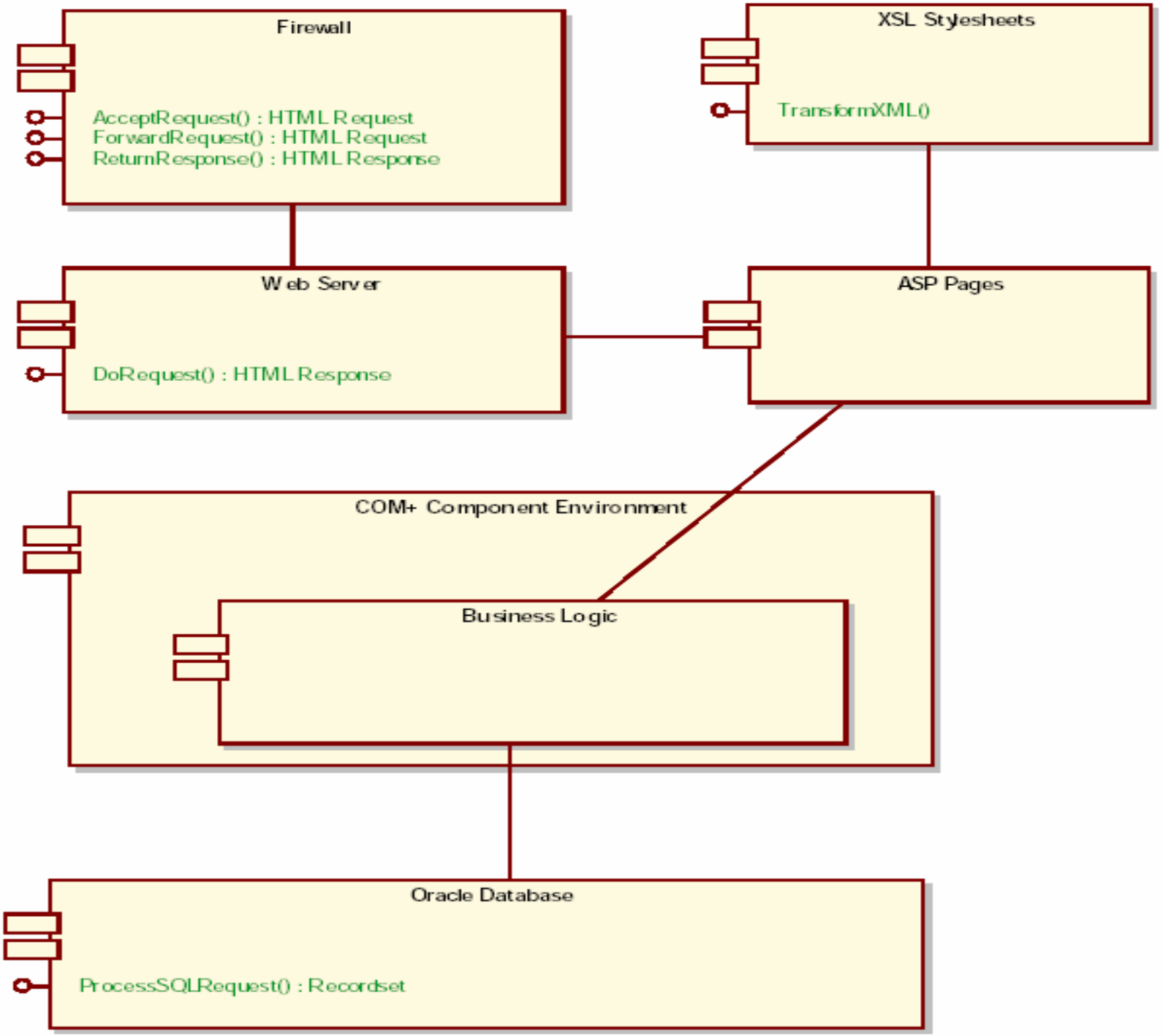
Tipe-tipe Component

- *Deployment Component*: Yang menjadi basis dari *executable system*. Contoh *deployment component* diantaranya: DAN LAIN-LAIN(*Dynamic Library Link*) *file exe*, *Active X Control*, *Java Bean* dan lain-lain.
- *Work Product Component*: Yaitu *file-file* yang dibutuhkan untuk pembuatan *deployment component*. Contoh untuk *component* kedua ini diantaranya *file* data, *file source code* dan lain-lain.
- *Execution Component*: Yang dibuat sebagai hasil dari sistem yang akan dijalankan.

Stereotype Component

- Executable, menspesifikasikan komponen yang dieksekusi di suatu node.
- Library, menspesifikasikan pustaka objek static atau dinamik.
- Table, menspesifikasikan komponen yang merepresentasikan table basis data.
- File, menspesifikasikan komponen yang merepresentasikan dokumen yang berisi kode sumber atau data.
- Document, menspesifikasikan komponen yang merepresentasikan dokumen.

Elemen dan descriptionnya	Simbol
Komponen adalah sebuah blok bangunan fisik dari sistem. Hal ini digambarkan sebagai persegi panjang dengan tab.	
Interface Sebuah antarmuka menggambarkan sekelompok operasi digunakan atau dibuat oleh komponen.	


Firewall
AcceptRequest() : HTML Request
ForwardRequest() : HTML Request
ReturnResponse() : HTML Response
XSL Stylesheets
TransformXML()
Web Server
DoRequest() : HTML Response
ASP Pages
COM+ Component Environment
Business Logic
Oracle Database
ProcessSQLRequest() : Recordset

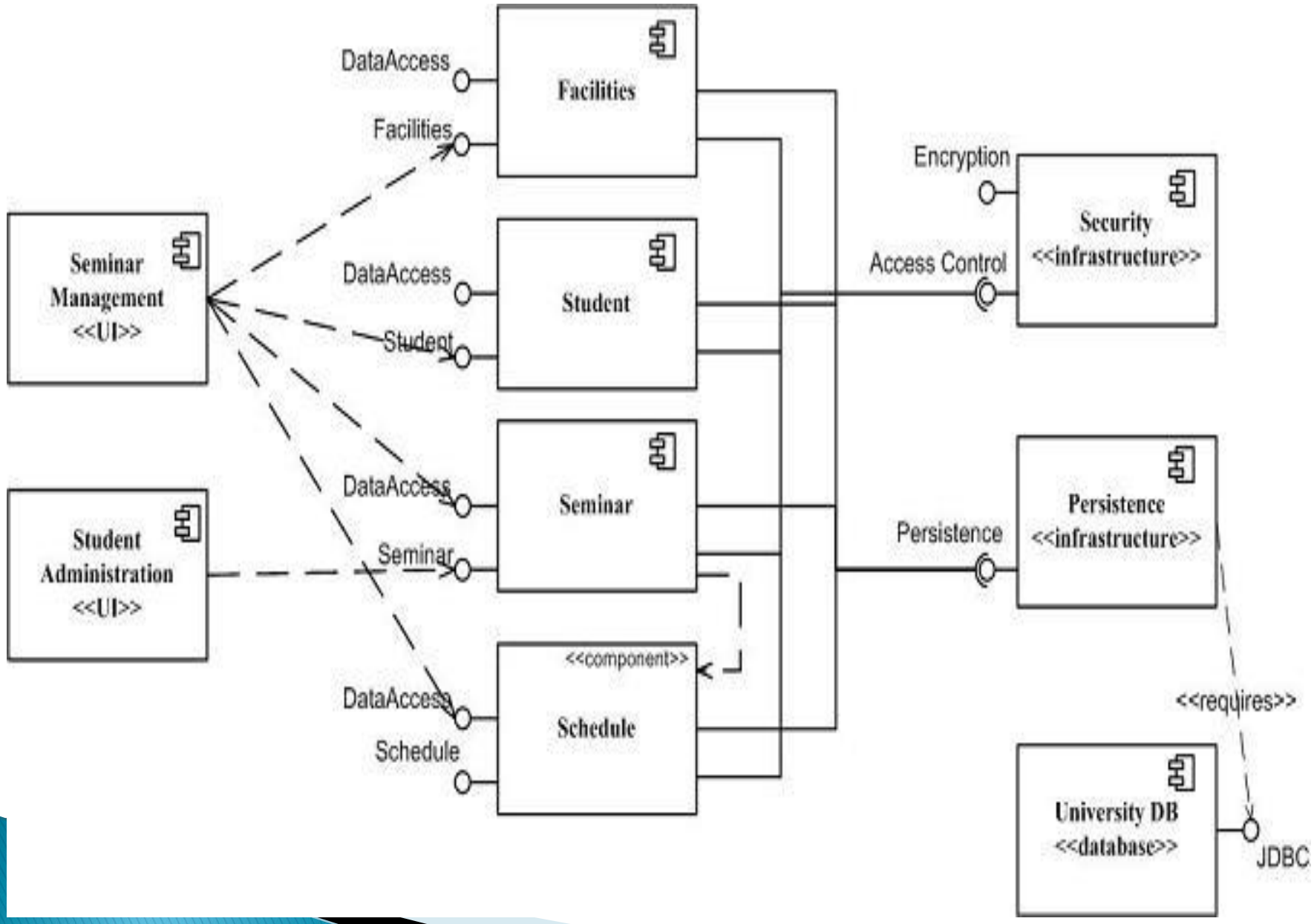
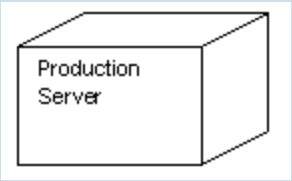

Facilities
DataAccess
Facilities
Seminar Management <<UI>>
Student
DataAccess
Student
Seminar
DataAccess
Seminar
Student Administration <<UI>>
<<component>>
Schedule
DataAccess
Schedule
Encryption
Security <<infrastructure>>
Access Control
Persistence <<infrastructure>>
Persistence
<<requires>>
University DB <<database>>
JDBC

Diagram Deployment

Fungsi

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.

Elemen dan descriptionnya	Simbol
Node: Elemen yang menyediakan lingkungan eksekusi untuk komponen-komponen sistem. Digambarkan oleh kubus dengan nama obyek di dalamnya, didahului oleh titik dua, dan digarisbawahi	
Koneksi: Serupa dengan relasi / asosiasi yang digunakan dalam diagram kelas untuk menentukan interkoneksi antar node.	

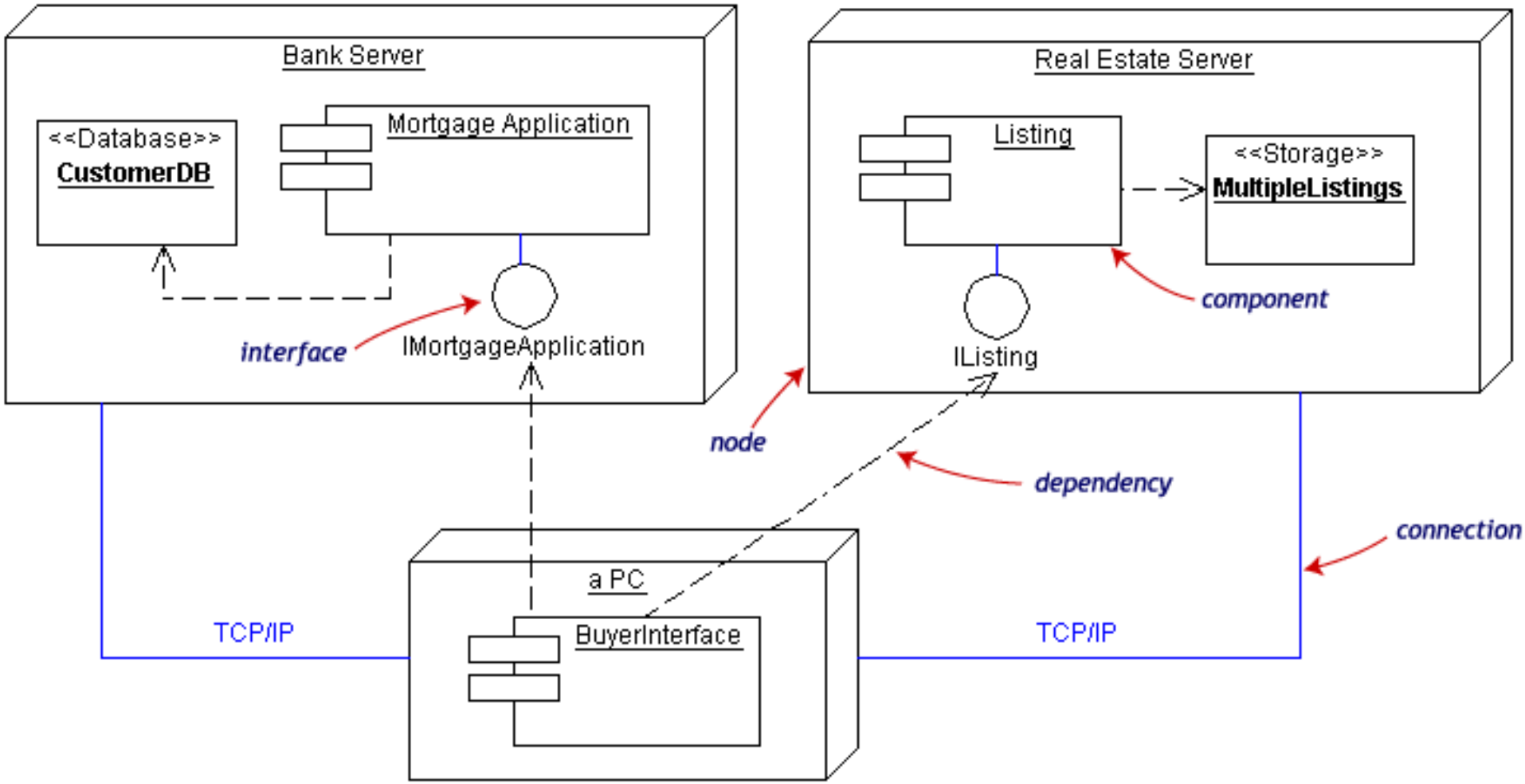

Bank Server
<<Database>>
CustomerDB
Mortgage Application
interface
IMortgageApplication
Real Estate Server
Listing
<<Storage>>
MultipleListings
component
IListing
node
dependency
connection
a PC
BuyerInterface
TCP/IP
TCP/IP

Diagram Deployment untuk Memodelkan Embedded System

- Identifikasi perangkatdan node yang unik terhadap sistem
- Menyediakan icon visual menggunakan mekanisme perluasan di UML untuk mendefinisikan stereotype dengan ikon yang cocok. Paling tidak kita harus membedakan antara pemroses (komponen yang memiliki perangkat lunak) dan perangkat (device).
- Memodelkan keterhubungan antara pemroses-pemroses dan perangkat-perangkat di diagram deployment. Serupa itu,spesifikasikan keterhubungan antara komponen-komponen di pandangan implementasi sistem dan node-node di pandangan deployment sistem.
- Bila diperlukan,perluas perangkat intelejen dengan memodelkan strukturnya dengan diagram deployment lebih rinci.

Diagram deployment untuk Memodelkan Sistem Client/Server

- Identifikasi node-node yang mempresentasikan pemroses client dan server.
- Beri perhatian pada perangkat-perangkat unik terhadap kelakuan sistem seperti credit card reader,badge reader,dan perangkat tampilan selain monitor karena penempatanya di topologi perangkat keras sistem berarti secara arsitektur.
- Sediakan ikon untuk pemroses dan perangkat ini dengan menggunakan fasilitas mekanisme stereotype UML.
- Memodelkan topologi node-node ini di diagram deployment.serupa itu,spesifikasikan keterhubungan antara komponen-komponen di pandangan implementasi sistem dan node-node pandangan deployment sistem.

Diagram Deployment untuk Memodelkan Sistem Tersebar Penuh

- Identifikasi perangkat-perangkat dan pemroses-pemroses sebagai sistem client/server lebih sederhana.
- Jika perlu kinerja jaringan sistem atau dampak perubahan ke jaringan, jamin kita memodelkan perangkat-perangkat komunikasi ke level ke cukup rinci untuk dapat memberi perhitungan.
- Beri perhatian ke pengelompokan node-node logic yang dapat dispesifikasikan menggunakan paket.
- Memodelkan perangkat-perangkat dan pemroses-pemroses ini menggunakan diagram deployment.Bila dimungkinkan,gunakan tool untuk menemukan topologi sistem dengan menelusuri jaringan sistem.
- Jika kita perlu focus pada dinamis sitem,pergunakan diagram use-case untuk menspesifikasikasn jenis kelakuan yang diperlukan dan lakukan perluasan perluasan use-case.