

Pengenalan Unified Modelling Language

Rahmatullah Arrizal S.Kom, M.T

S-1 Teknik Informatika



From West Java for Indonesia to the World through SDGs

www.unpad.ac.id



Tujuan Pembelajaran

- Memahami konsep UML
- Memahami perkembangan UML
- Memahami diagram-diagram dalam UML



Cakupan Materi

- ⊕ Pengertian UML
- ⊕ Asal Mula UML
- ⊕ Sejarah UML
- ⊕ Peran Penting UML
- ⊕ UML Overview:
structure diagram &
dynamic behavior





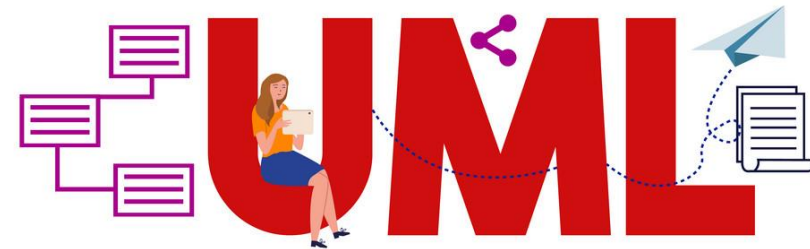
Pengertian UML

- UML (*Unified Modeling Language*) adalah sekumpulan diagram yang digunakan untuk melakukan abstraksi terhadap sebuah sistem atau perangkat lunak berbasis objek.
- Merupakan bahasa pemodelan standar yang terdiri dari sekumpulan diagram terintegrasi
- Dikembangkan untuk membantu pengembang sistem dan perangkat lunak untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan artefak sistem perangkat lunak, serta untuk pemodelan bisnis dan lainnya sistem non-perangkat lunak.
- UML merepresentasikan kumpulan praktik teknik terbaik yang telah terbukti berhasil dalam pemodelan sistem yang besar dan kompleks.



Tujuan UML

- Tujuan dari UML adalah untuk menyediakan notasi standar yang dapat digunakan oleh semua metode berorientasi objek dan untuk memilih serta mengintegrasikan elemen terbaik dari notasi prekursor.
- UML telah dirancang untuk berbagai aplikasi. Oleh karena itu, UML menyediakan konstruksi untuk berbagai sistem dan aktivitas (misalnya, sistem terdistribusi, analisis, desain dan penerapan sistem).





Asal Mula UML

UML merupakan notasi hasil penyatuan OMT dari:

- ⊕ Teknik Pemodelan Objek OMT [James Rumbaugh 1991] - paling baik untuk analisis dan sistem informasi intensif data.
- ⊕ Booch [Grady Booch 1994] - sangat bagus untuk desain dan implementasi. Grady Booch telah bekerja secara ekstensif dengan bahasa ADA, pemain utama dalam pengembangan teknik Berorientasi Objek untuk bahasa tersebut. Meskipun metode Booch kuat, notasinya kurang diterima dengan baik.
- ⊕ OOSE (Object Oriented Software Engineering - [Ivar Jacobson 1992]) - menampilkan model yang dikenal sebagai Use case. Use Case adalah teknik yang ampuh untuk memahami perilaku seluruh sistem (area di mana OO secara tradisional lemah).

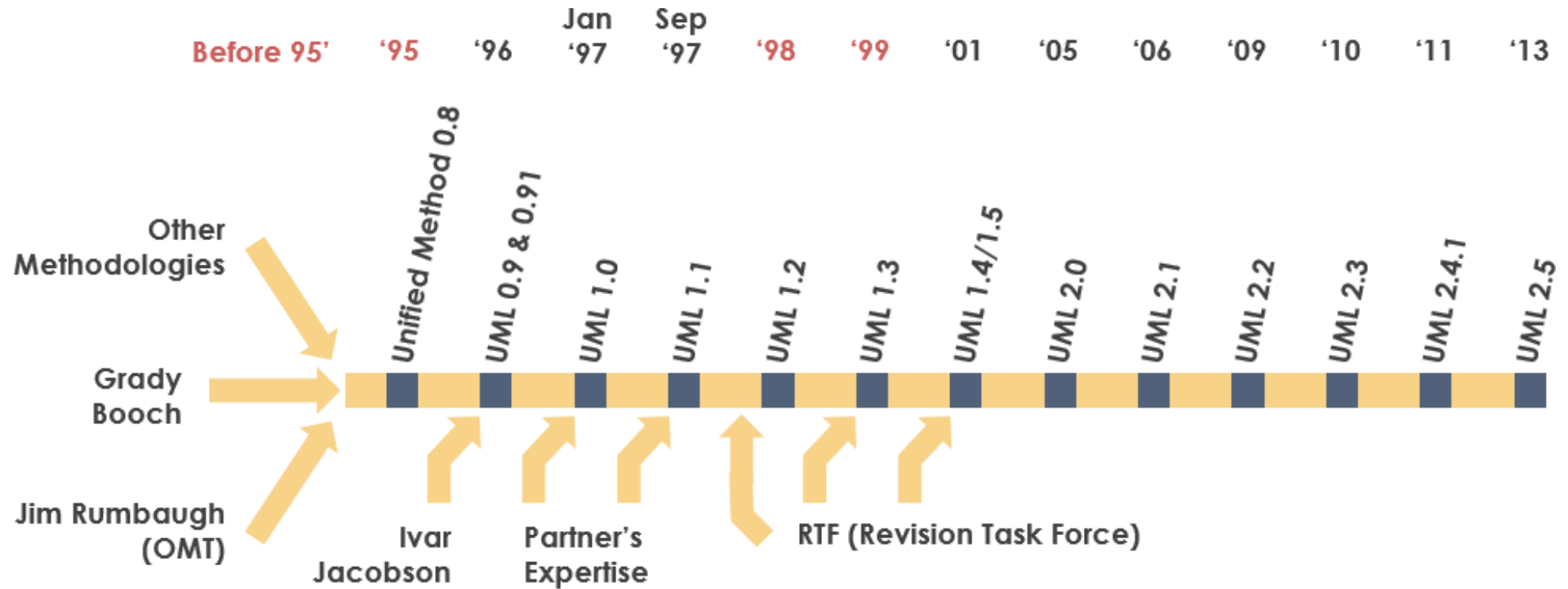


Asal Mula UML

- ⊕ UML juga telah dipengaruhi oleh notasi berorientasi objek lainnya:
 1. Mellor dan Shlaer [1998]
 2. Coad dan Yourdon [1995]
 3. Wirfs-Brock [1990]
 4. Martin dan Odell [1992]
- ⊕ UML juga mencakup konsep baru yang tidak ada dalam metode utama lainnya pada saat itu, seperti mekanisme ekstensi dan bahasa batasan.



Sejarah UML



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>



Kenapa Perlu UML?

- Karena nilai strategis *software* meningkat bagi banyak perusahaan, industri mencari teknik untuk mengotomatiskan produksi perangkat lunak dan untuk meningkatkan kualitas serta mengurangi biaya dan waktu pemasaran.
- Teknik ini termasuk teknologi komponen, pemrograman visual, pola dan kerangka kerja.
- Bisnis juga mencari teknik untuk mengelola kompleksitas sistem yang meningkat dalam lingkup dan skala.
- Adanya kebutuhan untuk memecahkan masalah arsitektur yang berulang, seperti distribusi fisik, konkurensi, replikasi, keamanan, penyeimbangan beban, dan toleransi kesalahan
- Perkembangan untuk World Wide Web, yang dapat membuat beberapa hal menjadi lebih sederhana, telah memperburuk masalah arsitektur ini.



Tujuan Spesifik UML

Menurut Page-Jones dalam Fundamental Object-Oriented Design, tujuan spesifik dari UML, antara lain:

- ⊕ Memberi pengguna bahasa pemodelan visual ekspresif yang siap digunakan sehingga mereka dapat mengembangkan dan bertukar model yang bermakna.
- ⊕ Menyediakan mekanisme perluasan dan spesialisasi untuk memperluas konsep inti.
- ⊕ Bebas dari bahasa pemrograman dan proses pengembangan tertentu.
- ⊕ Berikan dasar formal untuk memahami bahasa pemodelan.
- ⊕ Mendorong pertumbuhan pasar tools Object Oriented (OO).
- ⊕ Mendukung konsep pengembangan tingkat yang lebih tinggi seperti kolaborasi, kerangka kerja, pola, dan komponen.
- ⊕ Mengintegrasikan *best practice*.



Overview UML

- Terdapat banyak model diagram yang digunakan dalam UML.
- Alasannya adalah mengakomodasi kemungkinan untuk melihat suatu sistem dari banyak sudut pandang yang berbeda
- Pengembangan perangkat lunak akan melibatkan banyak pemangku kepentingan, seperti: Analis, Desainer, Programmer, Penguji/QA, User, dan Penulis Teknis
- Semua orang yang terlibat tertarik pada aspek sistem yang berbeda, dan masing-masing memerlukan tingkat detail yang berbeda

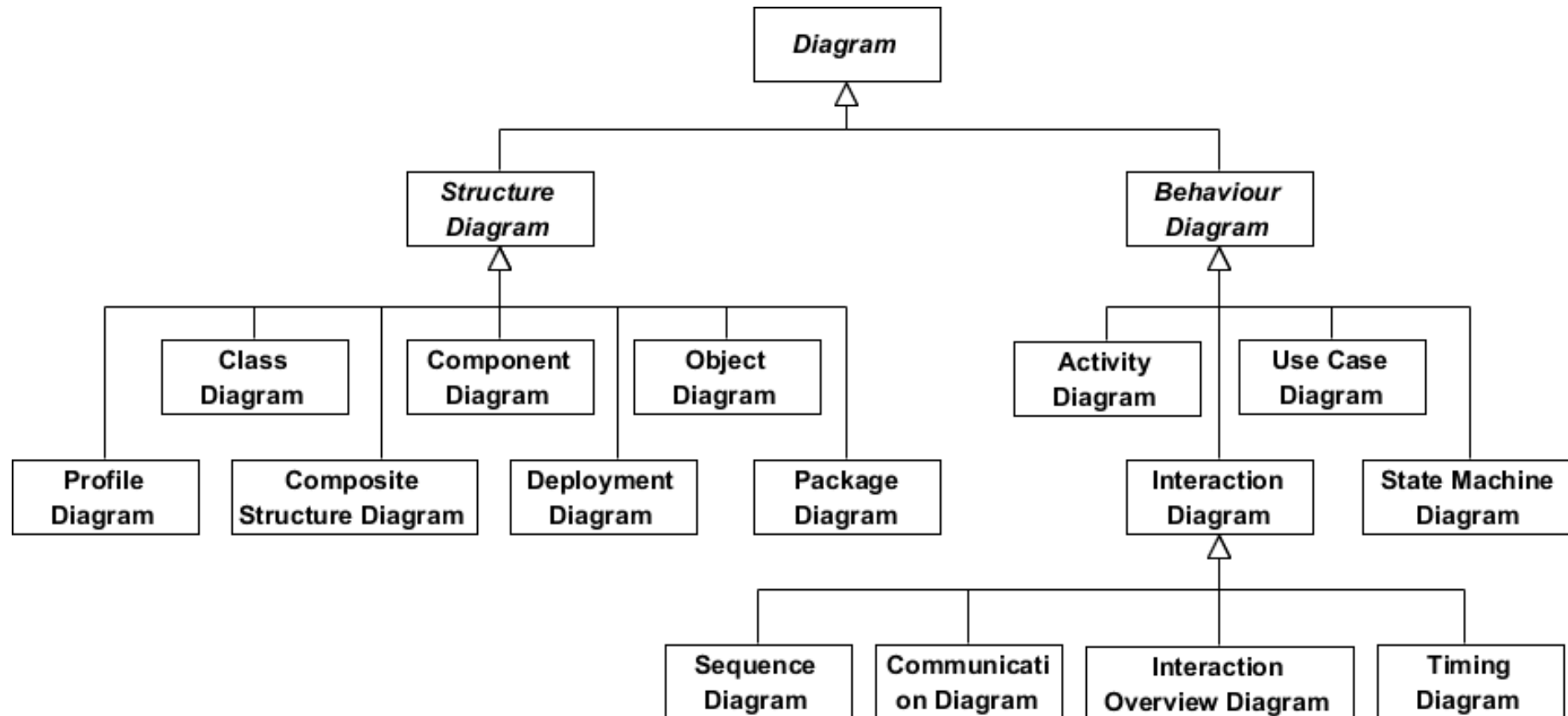


Overview UML

- Terdapat 2 pengelompokan diagram berdasarkan kebutuhan sudut pandang, yaitu **structure diagram** dan **behavior diagram**
- Structure diagram dibutuhkan untuk memahami desain sistem dan dapat mengubah desain ke kode level rendah
- Behavior diagram diperlukan untuk mengetahui perilaku sistem secara keseluruhan, dan perlu memahami bagaimana fungsi produk



Taksonomi Diagram UML



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>



Structural Diagram



Structural Diagram

- ⊕ Diagram struktur menunjukkan struktur statis dari sistem dan bagian-bagiannya pada tingkat abstraksi dan implementasi yang berbeda dan bagaimana keterkaitannya satu sama lain.
- ⊕ Elemen-elemen dalam diagram struktur merepresentasikan konsep yang bermakna dari suatu sistem, dan dapat mencakup konsep abstrak, dunia nyata, dan implementasi
- ⊕ Terdapat tujuh jenis diagram struktur sebagai berikut:
 1. Class Diagram
 2. Component Diagram
 3. Deployment Diagram
 4. Object Diagram
 5. Package Diagram
 6. Composite Structure Diagram
 7. Profile Diagram



Structural Diagram

Class Diagram

- ⊕ Class Diagram: sebuah diagram yang menjelaskan hubungan antar class dalam sebuah sistem yang sedang dibuat dan menjelaskan bagaimana caranya agar mereka saling berkolaborasi.
- ⊕ Class diagram: digunakan untuk menampilkan kelas-kelas maupun paket-paket yang ada pada suatu sistem yang nantinya akan digunakan.
- ⊕ Diagram ini dapat memberikan sebuah gambaran mengenai sistem maupun relasi-relasi yang terdapat pada sistem tersebut.



Structural Diagram

Relationship di Class Diagram

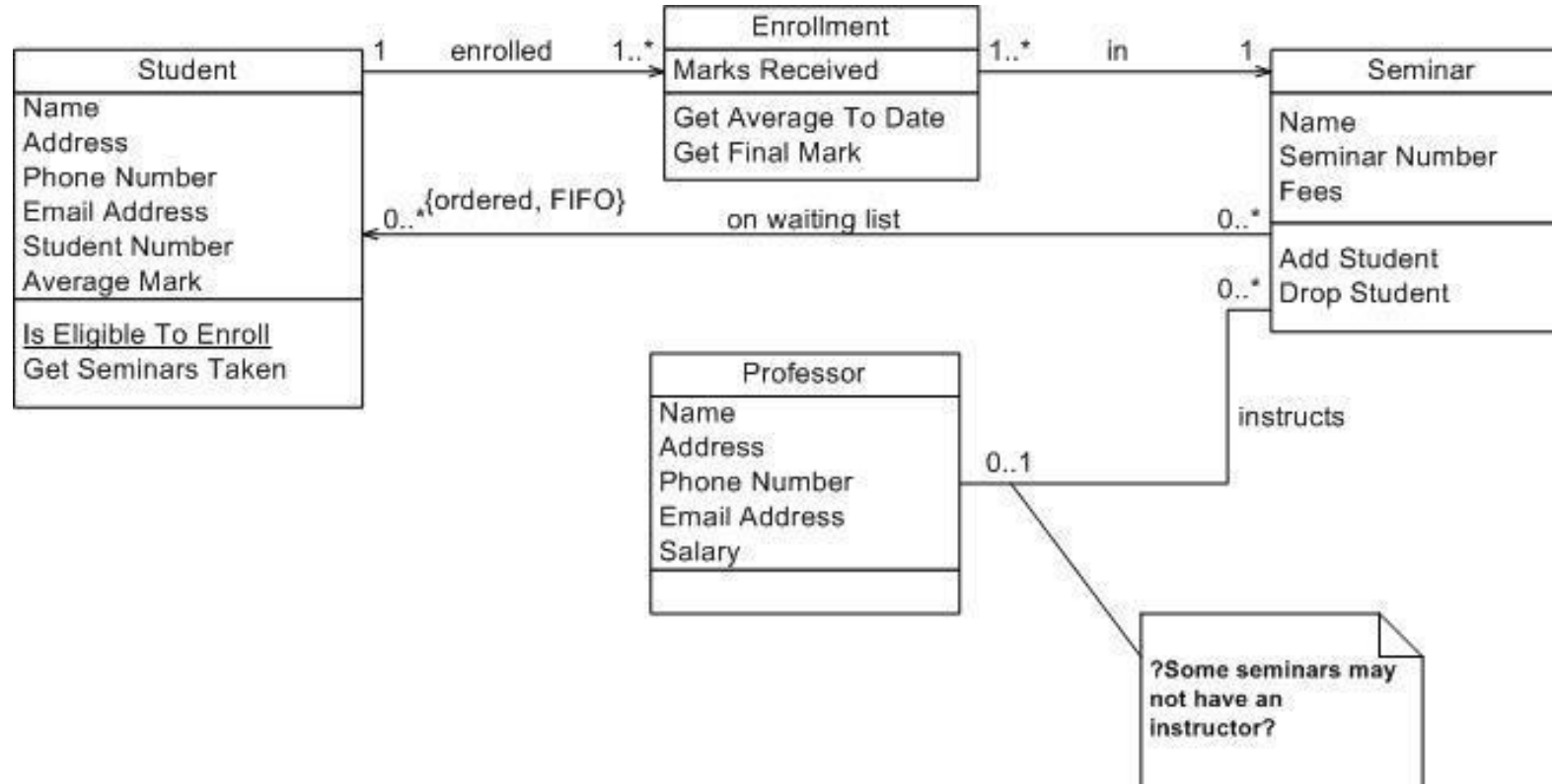
Terdapat tiga jenis hubungan utama yang penting:

- **Association** - mewakili hubungan antara contoh jenis (seseorang bekerja untuk perusahaan, perusahaan memiliki sejumlah kantor.)
- **Inheritance** - tambahan paling jelas untuk diagram ER untuk digunakan dalam OO. Ini memiliki korespondensi langsung dengan pewarisan dalam desain OO.
- **Aggregation** - Agregasi, suatu bentuk komposisi objek dalam desain berorientasi objek.



Structural Diagram

Contoh Class Diagram



Sumber : https://miro.medium.com/max/673/0*tEf7upkjOjGpCYvj.



Structural Diagram

Notasi Class Diagram

Simbol untuk class dan visibility dari class member (Class Diagram)

Class name
Attributes
Operations

Sumber :
<https://www.edrawsoft.com/symbols/class-and-interface.png>

+	<i>For Public</i>
-	<i>For Private</i>
#	<i>For Protected</i>
/	<i>For Derived</i>
~	<i>For Package</i>

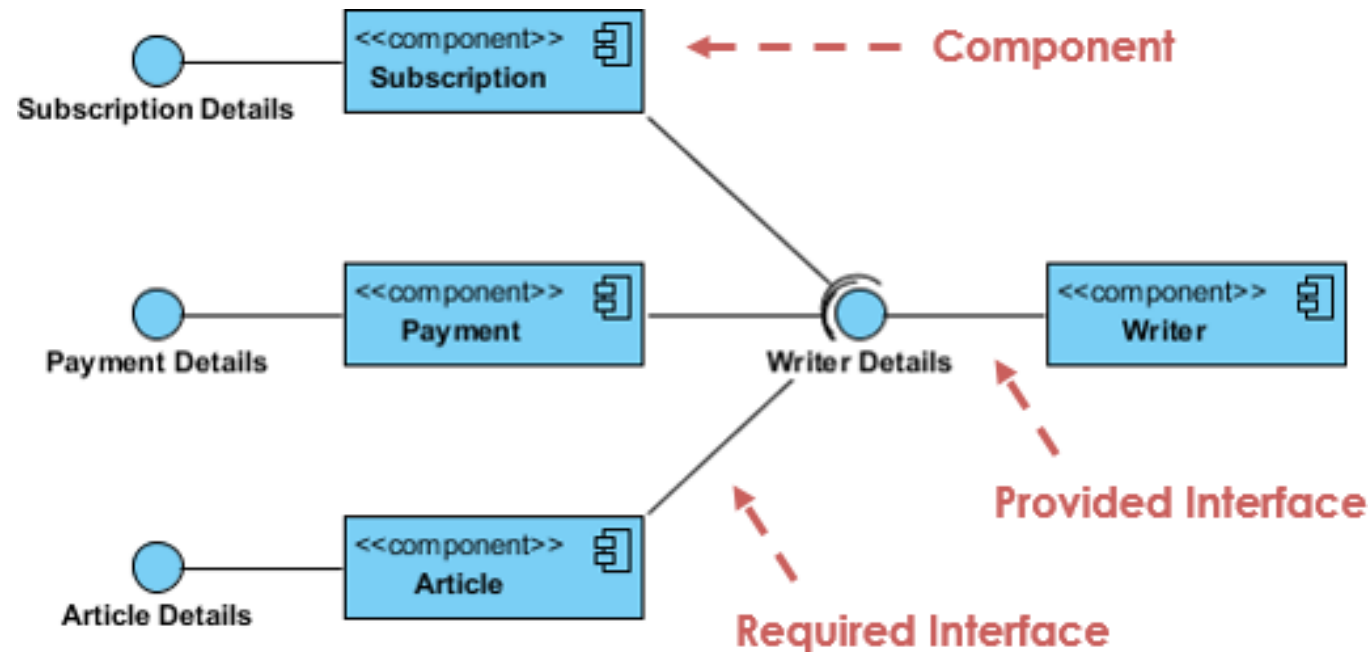
Sumber :
<https://wcs.smartdraw.com/uml-diagram/img/uml-class-members-notations.png?bn=1510011166>



Structural Diagram

Component Diagram

Component Diagram: diagram yang menampilkan komponen dalam sistem dan hubungan antara mereka.



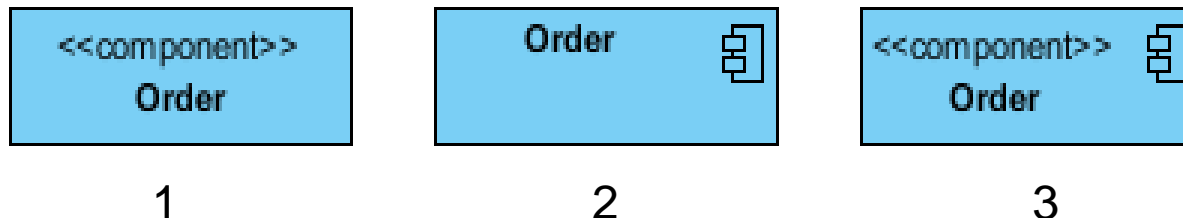
Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/>



Structural Diagram

Notasi Component Diagram

- ⊕ Dalam UML 2, sebuah komponen digambar sebagai persegi panjang dengan kompartemen opsional yang ditumpuk secara vertikal.
- ⊕ Tampilan level tinggi yang diabstraksi dari komponen di UML 2 dapat dimodelkan sebagai:
 1. Persegi panjang dengan nama komponen
 2. Persegi panjang dengan ikon komponen
 3. Persegi panjang dengan teks dan / atau ikon stereotip



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/>

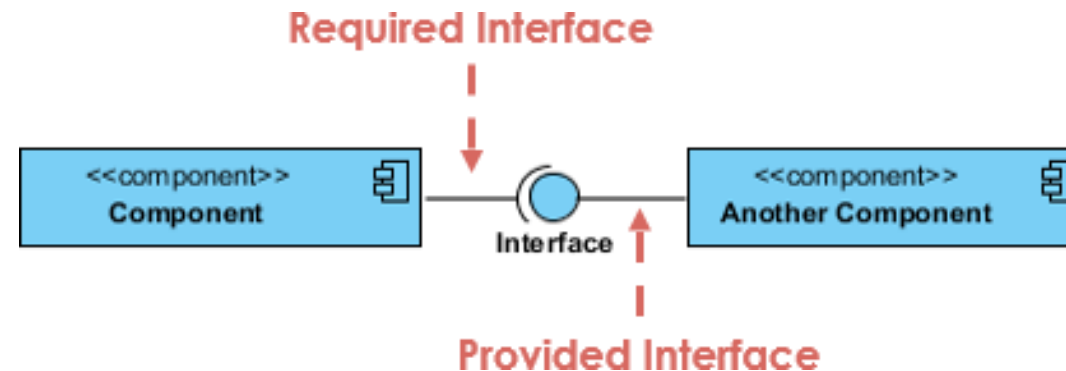


Structural Diagram

Notasi Component Diagram

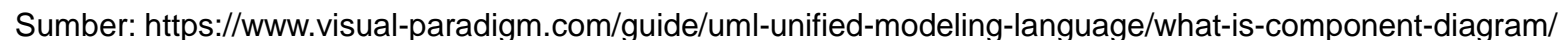
Interface

- ⊕ **Provided Interface** disimbolkan dengan lingkaran lengkap di ujungnya mewakili antarmuka yang disediakan komponen - simbol "lolipop" ini adalah singkatan dari hubungan realisasi pengklasifikasi antarmuka.
- ⊕ **Required Interface** disimbolkan dengan hanya setengah lingkaran di ujungnya (soket alias) mewakili antarmuka yang diperlukan komponen (dalam kedua kasus, nama antarmuka ditempatkan di dekat simbol antarmuka itu sendiri).



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/>

- ⊕ Pengklasifikasi subsistem adalah versi khusus dari pengklasifikasi komponen.
- ⊕ Elemen notasi subsistem mewarisi semua aturan yang sama dengan elemen notasi komponen.
- ⊕ Satu-satunya perbedaan adalah bahwa elemen notasi subsistem memiliki kata kunci subsistem, bukan komponen.



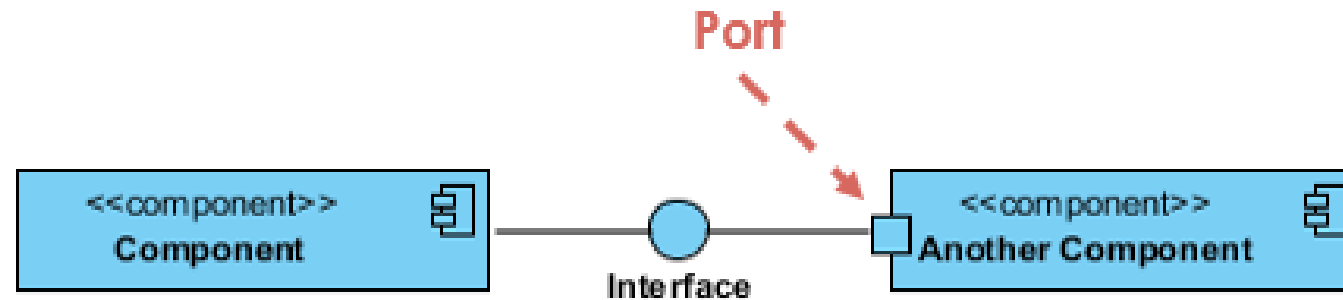


Structural Diagram

Notasi Component Diagram

Port

- ⊕ Port direpresentasikan menggunakan kotak di sepanjang tepi sistem atau komponen.
- ⊕ Port sering digunakan untuk membantu mengekspos antarmuka yang diperlukan dan disediakan dari suatu komponen.






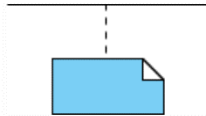
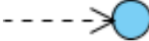

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/>



Structural Diagram

Notasi Component Diagram

Relationship

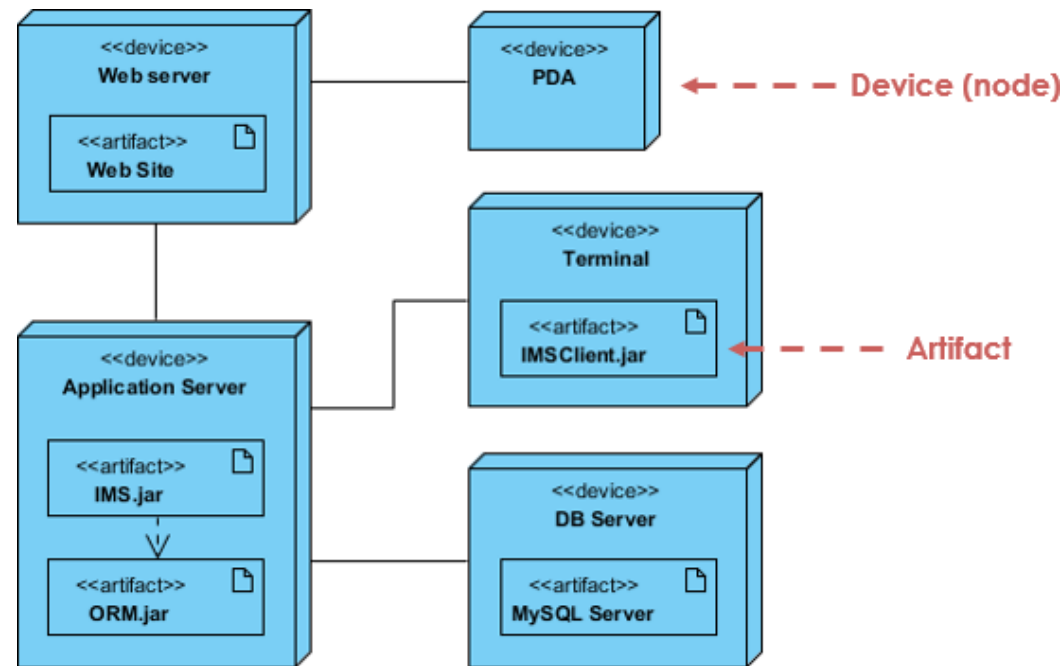
Jenis Hubungan	Notasi
Association	
Composition	
Aggregation	
Constraint	
Dependency	
Link	

Structural Diagram

Deployment Diagram



Deployment Diagram: Diagram yang menunjukkan tata letak sebuah sistem secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware* yang digunakan untuk mengimplementasikan sebuah sistem dan keterhubungan antara komponen-komponen *hardware*



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>



Structural Diagram

Kapan Menggunakan Deployment Diagram?

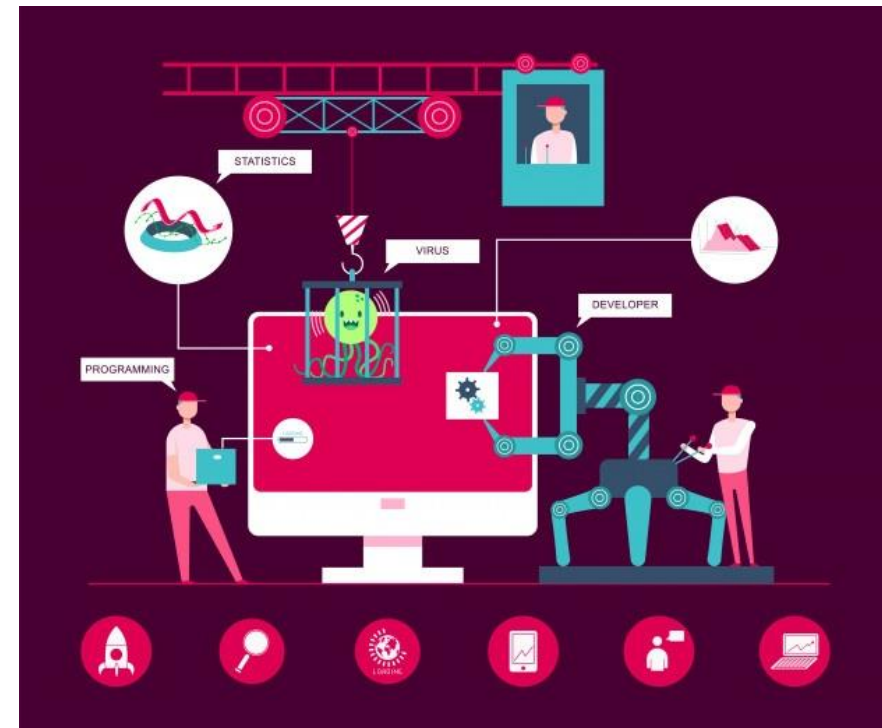
- ⊕ Ketika ingin menintegrasikan sistem baru dengan sistem yang ada
- ⊕ Ingin mengetahui seberapa kuat sistem yang dibutuhkan (contoh: berapa banyak hardware yang bisa terlibat/dikontrol)
- ⊕ Untuk mengetahui siapa dan bagaimana berinteraksi dengan sistem
- ⊕ Untuk mengetahui middleware, termasuk sistem operasi dan pendekatan serta protokol komunikasi, yang akan digunakan sistem
- ⊕ Untuk menggambarkan hardware dan software yang akan berinteraksi langsung dengan pengguna
- ⊕ Mengetahui cara memantau sistem setelah diterapkan
- ⊕ Untuk mengetahui kebutuhan mengenai tingkat keamanan sistem



Structural Diagram

Tujuan Deployment Diagram

- + Menunjukkan struktur sistem run-time
- + Menginformasikan perangkat keras yang akan digunakan untuk mengimplementasikan sistem dan hubungan antara berbagai item perangkat keras.
- + Memodelkan elemen perangkat keras fisik dan jalur komunikasi di antara mereka
- + Digunakan untuk merencanakan arsitektur suatu sistem.
- + Mendokumentasikan penyebaran komponen perangkat lunak atau node





Structural Diagram

Notasi Deployment Diagram

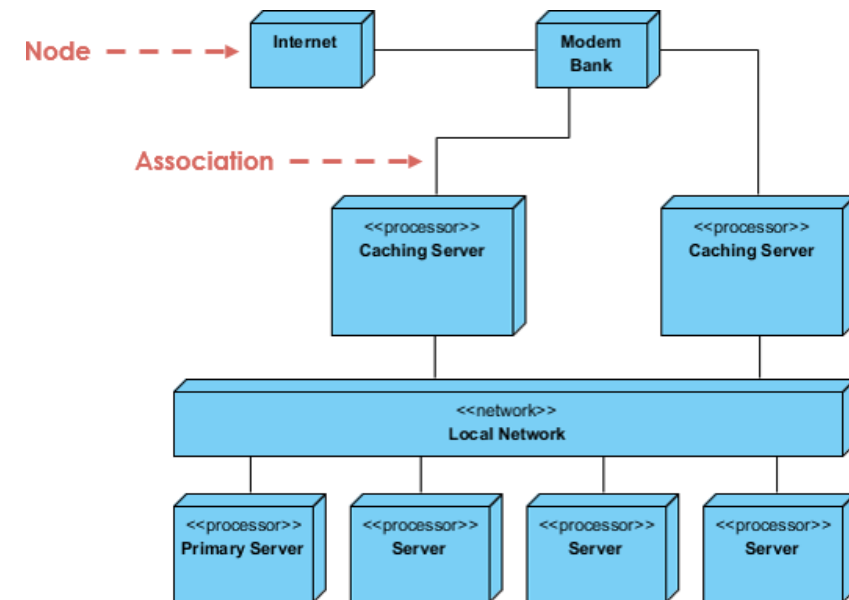
- Deployment diagram adalah jenis class diagram khusus, yang berfokus pada node sistem.
- Secara grafis, Deployment diagram adalah kumpulan simpul dan busur. Diagram penyebaran biasanya berisi:

1. Nodes

- 3-D box merepresentasikan sebuah node, baik software maupun hardware
- Node HW dapat disimbolkan <<stereotype>>
- Koneksi antar node disimbolkan dengan sebuah garis, dengan <<stereotype>> opsional
- Node dapat berada di dalam sebuah node

2. Notasi Relationship

- Dependency
- Association
- Dapat mengandung notes dan constraints.





Structural Diagram

Object Diagram

- ⊕ Object diagram adalah graf dari instance, termasuk objek dan nilai data.
- ⊕ Object diagram statis adalah turunan dari class diagram yang menunjukkan snapshot dari keadaan rinci dari suatu sistem pada suatu titik waktu.
- ⊕ Perbedaannya adalah bahwa class diagram merepresentasikan model abstrak yang terdiri dari kelas-kelas dan hubungannya.
- ⊕ Namun, object diagram mewakili sebuah instance pada momen tertentu, yang sifatnya konkret.
- ⊕ Penggunaan diagram objek cukup terbatas yaitu untuk menunjukkan contoh struktur data.
- ⊕ Class diagram dan Object Diagram mewakili dua aspek berbeda dari basis kode



Structural Diagram

Tujuan Object Diagram

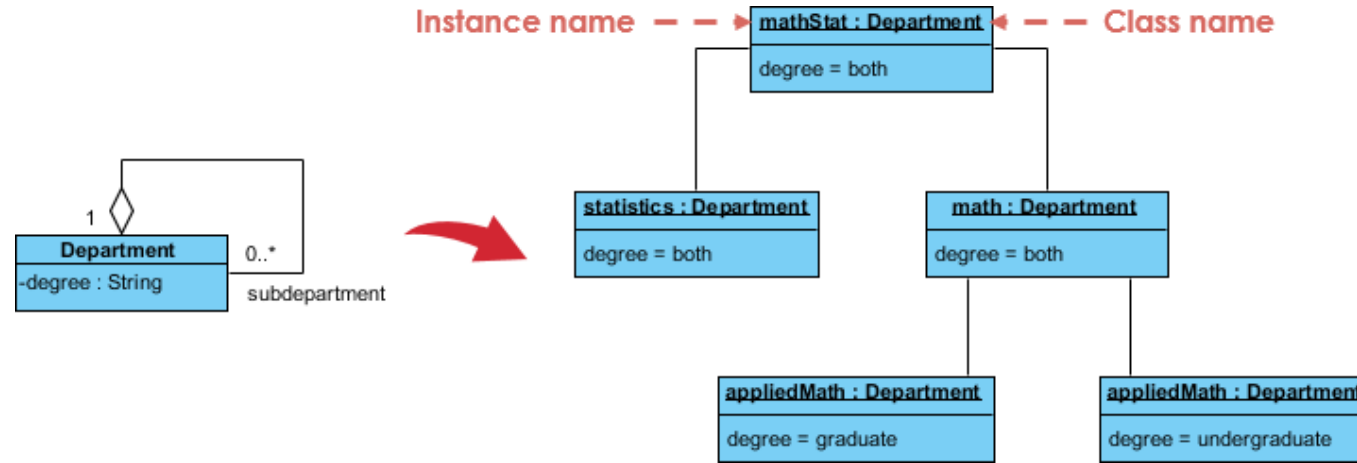
Penggunaan diagram objek cukup terbatas, terutama untuk menunjukkan contoh struktur data:

- ⊕ Selama tahap analisis proyek, analis dapat membuat class diagram untuk menggambarkan struktur sistem dan kemudian membuat satu set object diagram sebagai kasus uji untuk memverifikasi keakuratan dan kelengkapan class diagram.
- ⊕ Sebelum membuat class diagram, Analis dapat membuat diagram objek untuk menemukan fakta tentang elemen model tertentu dan tautannya, atau untuk menggambarkan contoh spesifik pengklasifikasi yang diperlukan.



Structural Diagram

Contoh & Notasi Object Diagram



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-object-diagram/>

Nama Notasi	Simbol
Object Name	
Object Attributes tidak seperti class, atribut objek harus memiliki nilai yang diberikan	
Links Link merupakan contoh yang terkait dengan asosiasi.	



Structural Diagram

Package Diagram

- ⊕ Package diagram, semacam structure diagram, menunjukkan penataan dan organisasi elemen model dalam proyek skala menengah hingga besar.
- ⊕ Diagram ini dapat menunjukkan struktur dan dependensi antara sub-sistem atau modul, menunjukkan tampilan sistem yang berbeda, misalnya, sebagai aplikasi multi-layered (alias multi-tier) - model aplikasi multi-layer.

Tujuan Package Diagram:

- ⊕ Package Diagram digunakan untuk menyederhanakan class diagram yang kompleks, diagram ini dapat mengelompokkan kelas menjadi paket.
- ⊕ Paket adalah kumpulan elemen UML yang terkait secara logis.
- ⊕ Paket digambarkan sebagai folder file dan dapat digunakan pada diagram UML mana pun.

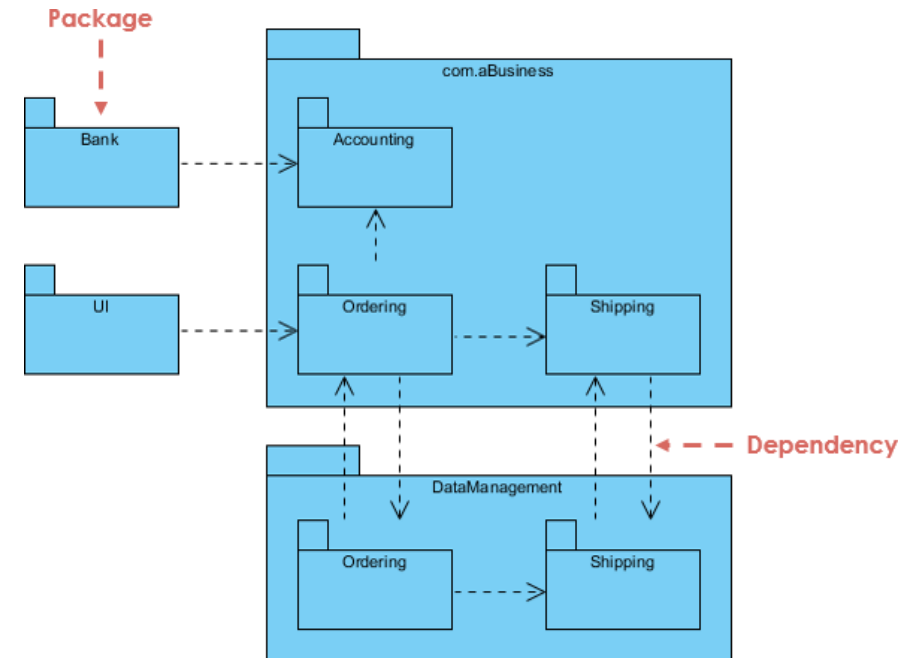


Structural Diagram

Notasi Package Diagram

Diagram berikut adalah model bisnis di mana kelas-kelas dikelompokkan ke dalam paket:

- ⊕ Paket muncul sebagai persegi panjang dengan tab kecil di bagian atas.
- ⊕ Nama paket ada di tab atau di dalam persegi panjang.
- ⊕ Panah putus-putus adalah dependency.
- ⊕ Satu paket bergantung pada paket lain jika perubahan di paket lain mungkin bisa memaksa perubahan di paket pertama.



Contoh Package Diagram

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/>



Structural Diagram

Notasi Package Diagram

- ⊕ Package Diagram mengikuti struktur hierarki dari paket bersarang.
- ⊕ Modul atom untuk paket bersarang biasanya berupa class diagram. Ada beberapa kendala saat menggunakan package diagram, yaitu sebagai berikut:
 1. Nama paket tidak boleh sama untuk suatu sistem, namun kelas di dalam paket yang berbeda dapat memiliki nama yang sama.
 2. Paket dapat mencakup seluruh diagram, nama komponen saja, atau tanpa komponen sama sekali.
 3. Nama paket yang sepenuhnya memenuhi syarat memiliki sintaks berikut.

Name owing the package :: Name of the package

java :: util :: Date

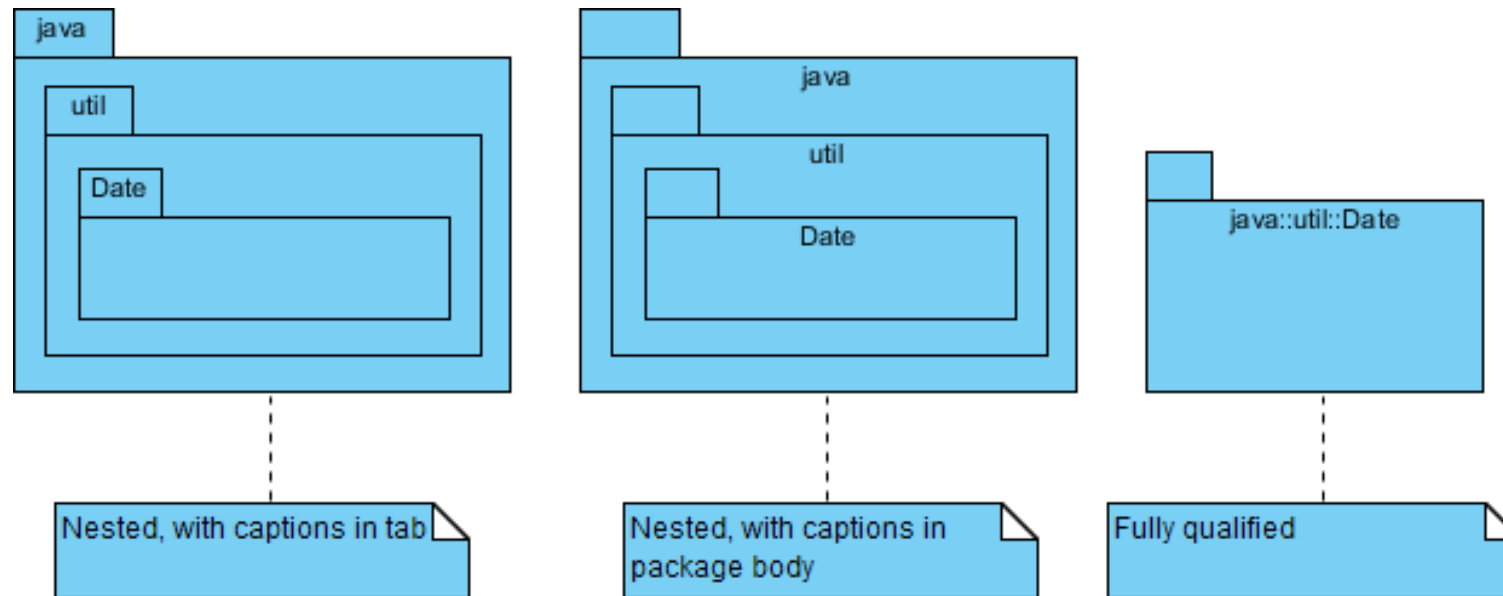
Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/>



Structural Diagram

Notasi Package Diagram

Paket dapat diwakili oleh notasi dengan beberapa contoh yang ditunjukkan di bawah ini:



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/>

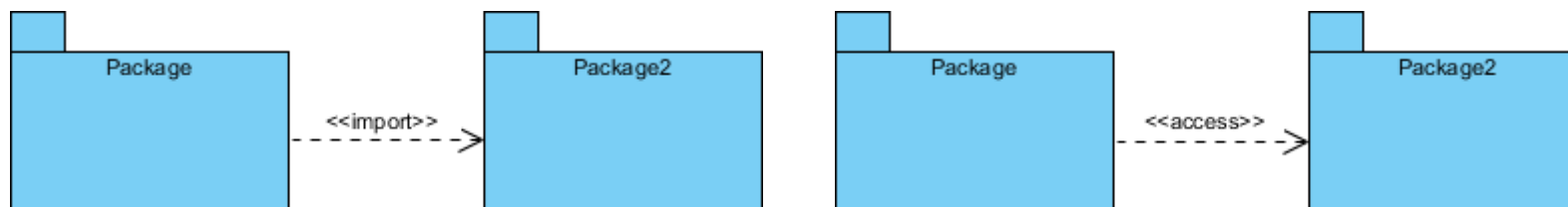


Structure Diagram

Notasi Package Diagram

Package Diagram – Notasi Dependency

- ⊕ Ada 2 tipe notasi dependency dalam package diagram: `<<import>>` & `<<access>>`.
- ⊕ Meskipun ada dua stereotip, pengguna dapat menggunakan stereotip sendiri untuk merepresentasikan jenis ketergantungan antara dua paket.



Package Import

Satu paket mengimpor fungsionalitas paket lainnya

Package Access

Satu paket membutuhkan bantuan dari fungsi paket lainnya.

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/>



Structural Diagram

Composite Structure Diagram

- ⊕ Composite Structure Diagram adalah salah satu artefak baru yang ditambahkan ke UML 2.0.
- ⊕ Diagram ini mirip dengan diagram kelas dan merupakan jenis diagram komponen yang terutama digunakan dalam pemodelan sistem pada sudut pandang mikro, tetapi diagram tersebut menggambarkan bagian-bagian individual, bukan seluruh kelas.
- ⊕ Ini adalah jenis diagram struktur statis yang menunjukkan struktur internal kelas dan kolaborasi yang dimungkinkan oleh struktur ini.



Structural Diagram

Composite Structure Diagram

- ⊕ Diagram ini dapat mencakup bagian internal, port tempat bagian-bagian tersebut berinteraksi satu sama lain atau melalui instance kelas yang berinteraksi dengan bagian-bagian dan dengan dunia luar, dan konektor antar bagian atau port.
- ⊕ Struktur komposit adalah sekumpulan elemen yang saling berhubungan yang berkolaborasi pada waktu proses untuk mencapai beberapa tujuan.
- ⊕ Setiap elemen memiliki beberapa peran yang ditentukan dalam kolaborasi.



Structural Diagram

Tujuan Composite Structure Diagram

- ⊕ Diagram ini memungkinkan pengguna untuk "Mengintip Ke Dalam" suatu objek untuk melihat dengan tepat terdiri dari apa.
- ⊕ Tindakan internal kelas, termasuk hubungan kelas bertingkat, bisa dirinci.
- ⊕ Objek ditampilkan sebagai komposisi dari objek rahasia lainnya.



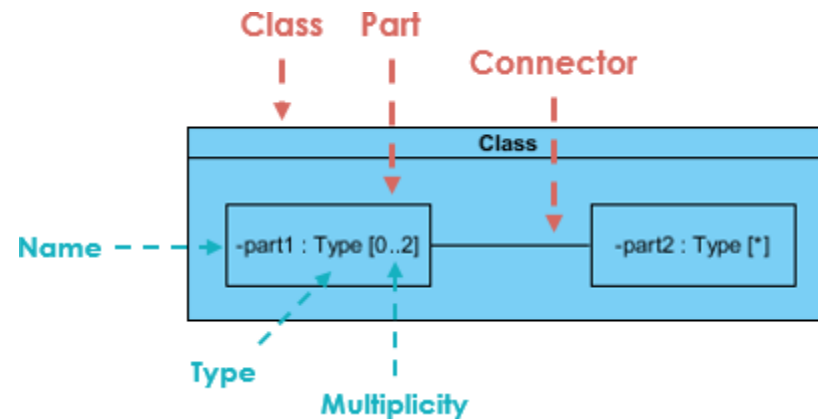
designed by  freepik



Structural Diagram

Notasi Composite Structure Diagram

- ⊕ Composite Structure Diagram menunjukkan bagian internal sebuah kelas.
- ⊕ Bagian diberi nama: `partName: partType [multiplicity]`
- ⊕ Kelas teragregasi adalah bagian dari kelas tetapi bagian belum tentu kelas, bagian adalah elemen apa pun yang digunakan untuk membuat kelas yang memuatnya.



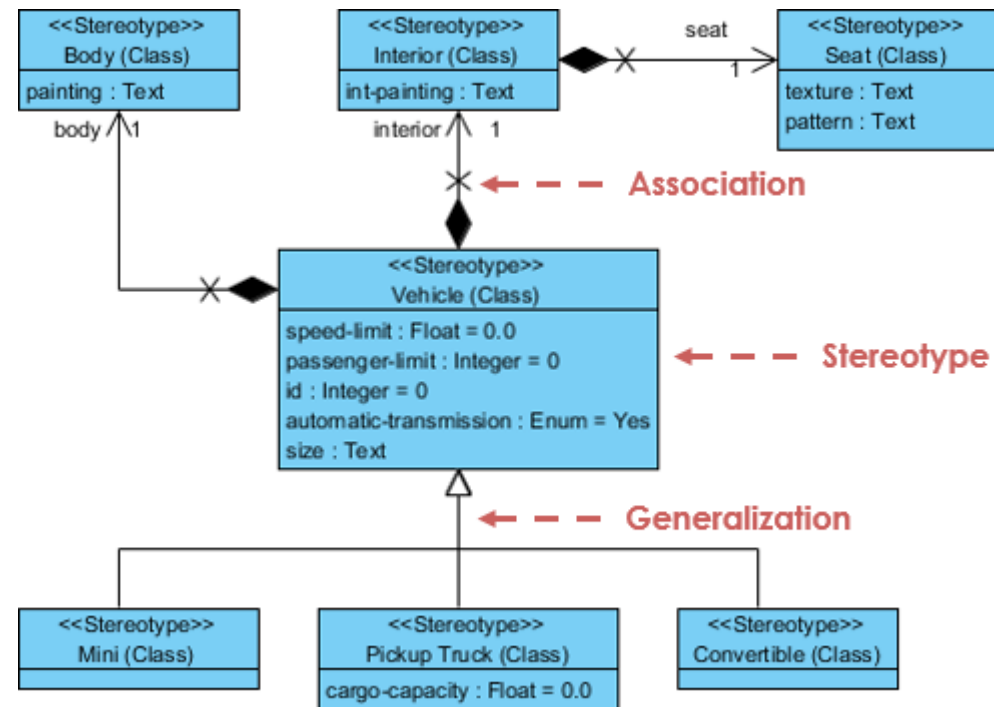
Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-composite-structure-diagram/>



Structural Diagram

Profile Diagram

- + Semacam diagram struktural dalam UML yang menyediakan mekanisme ekstensi generik untuk menyesuaikan model UML untuk domain dan platform tertentu.
- + Profile adalah kumpulan ekstensi yang secara kolektif menyesuaikan UML untuk domain tertentu (mis., Dirgantara, perawatan kesehatan, keuangan) atau platform (J2EE, .NET).



Contoh Profile Diagram

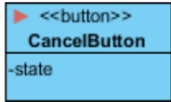
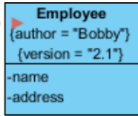
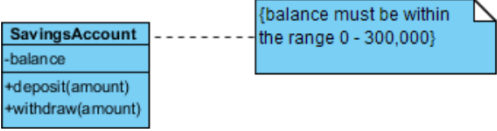
Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-profile-diagram/>



Structural Diagram

Notasi Profile Diagram

- ➊ Diagram profil pada dasarnya adalah mekanisme perpanjangan yang memungkinkan Anda untuk memperluas dan menyesuaikan UML dengan menambahkan blok penyusun baru, membuat properti baru, dan menentukan semantik baru untuk membuat bahasa sesuai dengan domain masalah spesifik Anda.
- ➋ Diagram profil memiliki tiga jenis mekanisme perpanjangan:

Jenis Mekanisme	Simbol
Stereotypes: Elemen model baru, yang berasal dari yang sudah ada tetapi memiliki properti spesifik yang sesuai dengan domain masalah.	Stereotype — — — 
Tagged Values: digunakan untuk memperluas properti UML sehingga informasi tambahan bisa ditambahkan dalam spesifikasi elemen model.	Two tagged values — — — 
Constraints: properti untuk menentukan semantik atau kondisi yang harus dipegang benar setiap saat.	



Behavioral Diagram



Behavioral Diagram

- ⊕ Behavioral Diagram menunjukkan perilaku dinamis dari objek dalam suatu sistem, yang dapat digambarkan sebagai rangkaian perubahan sistem dari waktu ke waktu
- ⊕ Terdapat tujuh jenis behavioral diagram sebagai berikut:
 1. Use Case Diagram
 2. Activity Diagram
 3. State Machine Diagram
 4. Sequence Diagram
 5. Communication Diagram
 6. Interaction Overview Diagram
 7. Timing Diagram



Behavioral Diagram

Use Case Diagram

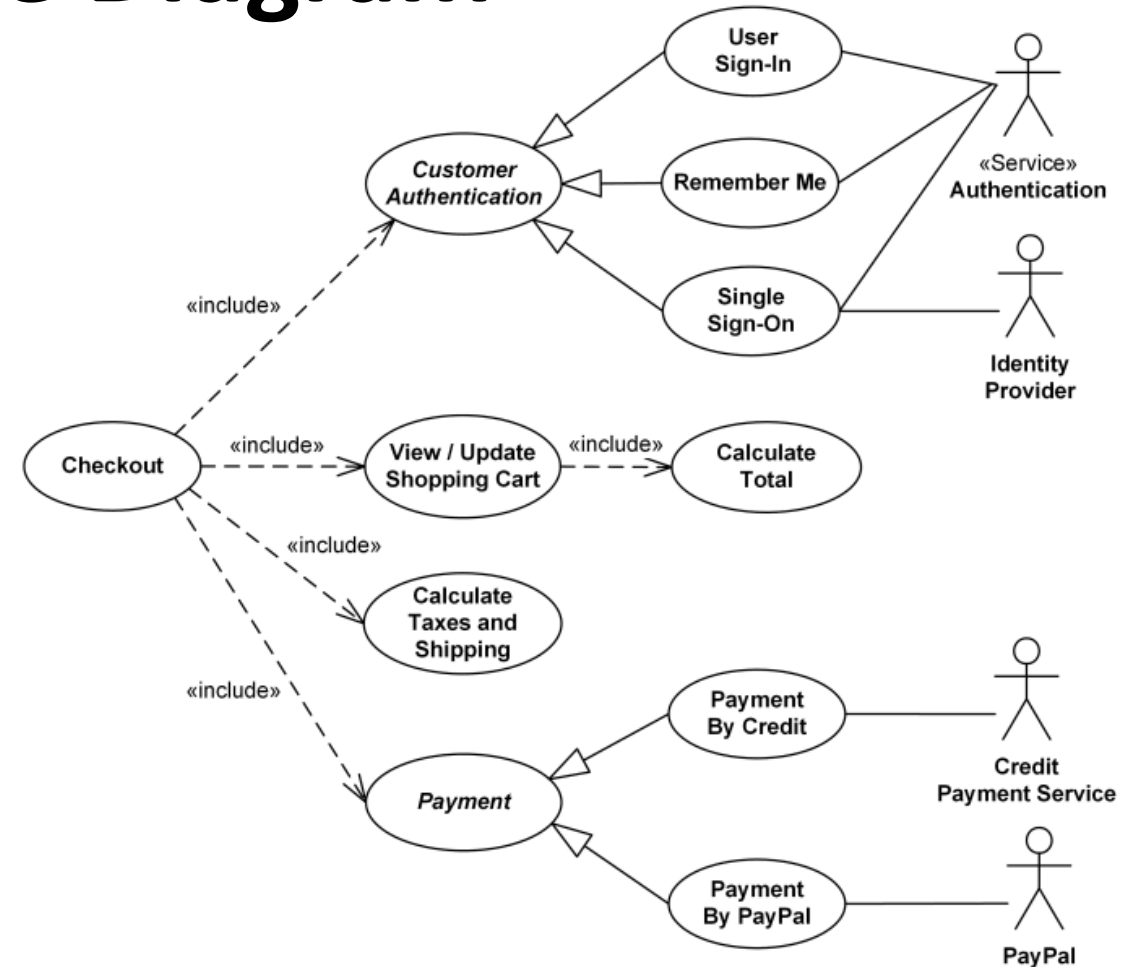
- ⊕ Tahun 1986, Ivar Jacobson pertama kali merumuskan teknik pemodelan tekstual dan visual untuk menentukan kasus penggunaan.
- ⊕ Tahun 1992, bukunya yang ditulis bersama Object-Oriented Software Engineering - A Use Case Driven Approach membantu mempopulerkan teknik untuk menangkap persyaratan fungsional, terutama dalam pengembangan perangkat lunak.
- ⊕ Diagram yang menunjukkan peran *user* dan bagaimana peran tersebut ketika menggunakan sistem.
- ⊕ Use case diagram juga dapat digunakan untuk mempresentasikan interaksi *user* dengan sistem dan menggambarkan spesifikasi kasus penggunaan.



Behavioral Diagram

Contoh Use Case Diagram

Use case diagram:
mendeskripsikan tipe
interaksi antara si
pemakai sistem dengan
sistemnya.



Sumber : https://miro.medium.com/max/680/0*9wQpTu9IZEDAWEs9.



Behavioral Diagram

Tujuan Use Case Diagram





Use Case Diagram biasanya dikembangkan pada tahap awal pengembangan dan orang sering menerapkan pemodelan use case untuk tujuan berikut:

- ⊕ Menentukan konteks sistem
- ⊕ Menangkap kebutuhan sistem
- ⊕ Memvalidasi arsitektur sistem
- ⊕ Mendorong penerapan dan buat kasus uji
- ⊕ Dikembangkan oleh analis bersama dengan pakar domain



Behavioral Diagram

Notasi Use Case Diagram

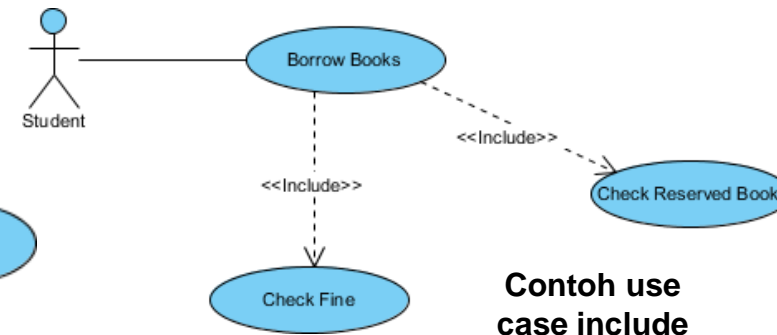
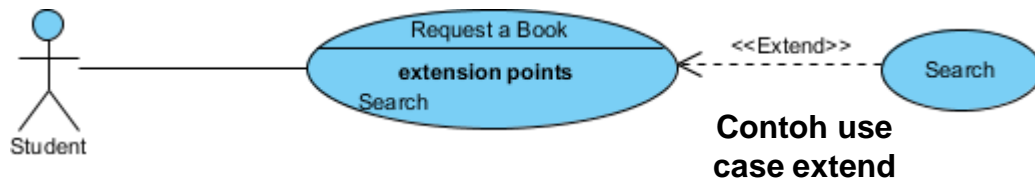
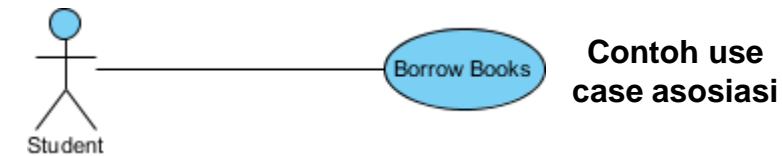
Deskripsi Notasi	Simbol
Actor: Seseorang berinteraksi dengan kasus penggunaan (fungsi sistem); kata benda; aktor berperan dalam bisnis; mirip dengan konsep pengguna, tetapi pengguna dapat memainkan peran yang berbeda;	
Use Case: Fungsi sistem (proses - otomatis atau manual); Dinamakan dengan kata kerja + Kata Benda (atau Frasa Kata Benda); mis. Lakukan sesuatu; Setiap Aktor harus ditautkan ke use case, sementara beberapa use case mungkin tidak terkait dengan aktor.	
Communication Link: Partisipasi seorang aktor dalam use case ditunjukkan dengan menghubungkan seorang aktor ke use case melalui tautan yang kuat; Aktor dapat dihubungkan ke use case oleh asosiasi, yang menunjukkan bahwa pelaku dan use case berkomunikasi satu sama lain menggunakan pesan.	
Boundary of system: Batas sistem berpotensi menjadi keseluruhan sistem seperti yang ditentukan dalam dokumen persyaratan; Untuk sistem yang besar dan kompleks, setiap modul mungkin merupakan batasan sistem.	

Behavioral Diagram

Notasi Use Case Diagram



Deskripsi Relasi	Simbol
Extends: Menunjukkan bahwa kasus penggunaan "Invalid Password" dapat mencakup (sesuai dengan yang ditentukan dalam ekstensi) perilaku yang ditentukan oleh use case dasar "Login Account".	
Include: Ketika kasus penggunaan digambarkan sebagai menggunakan fungsionalitas kasus penggunaan lain, hubungan antara kasus penggunaan disebut sebagai hubungan penyertaan atau penggunaan.	
Generalization: Hubungan generalisasi adalah hubungan induk-anak antara kasus penggunaan; Kasus penggunaan anak merupakan peningkatan dari kasus penggunaan orang tua.	

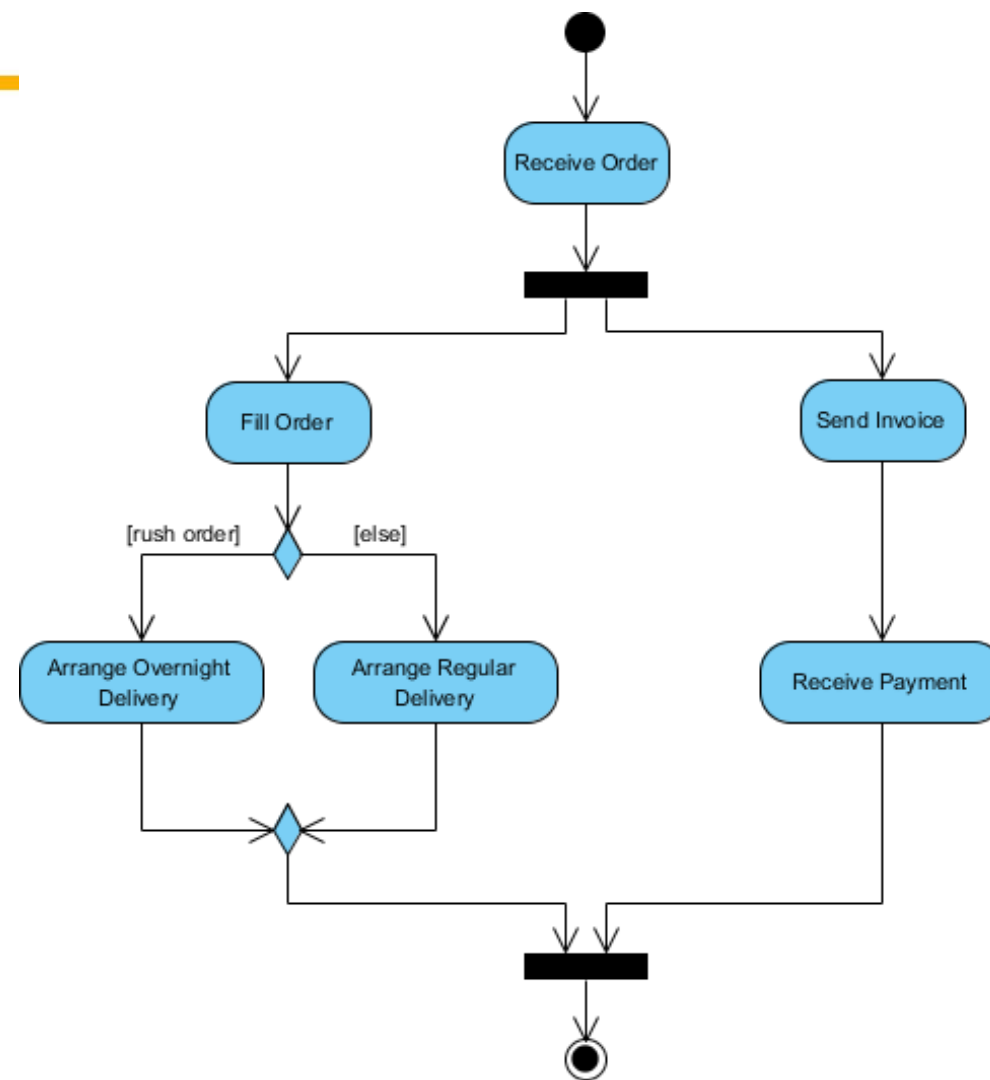




Behavioral Diagram

Activity Diagram

- + Activity Diagram: diagram yang menggambarkan alur kerja dari berbagai aktivitas *user* atau sistem, orang yang melakukan aktivitas, dan aliran berurutan dari aktivitas ini.
- + Activity Diagram pada dasarnya adalah diagram alir versi lanjutan yang memodelkan aliran dari satu aktivitas ke aktivitas lainnya.

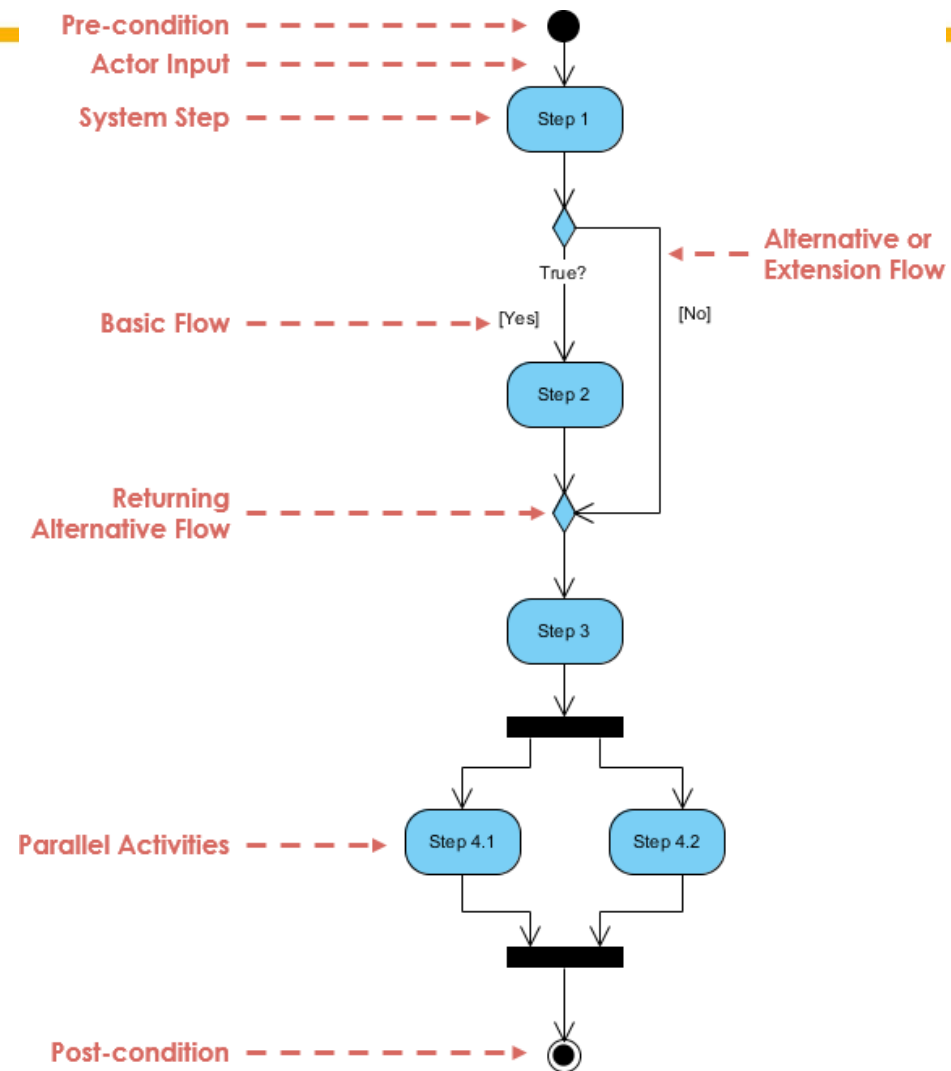


Contoh Activity Diagram Order hingga Transaksi

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>

Behavioral Diagram Notasi Activity Diagram

Activity Diagram
dasar - seperti
diagram alir



Notasi Activity Diagram

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>

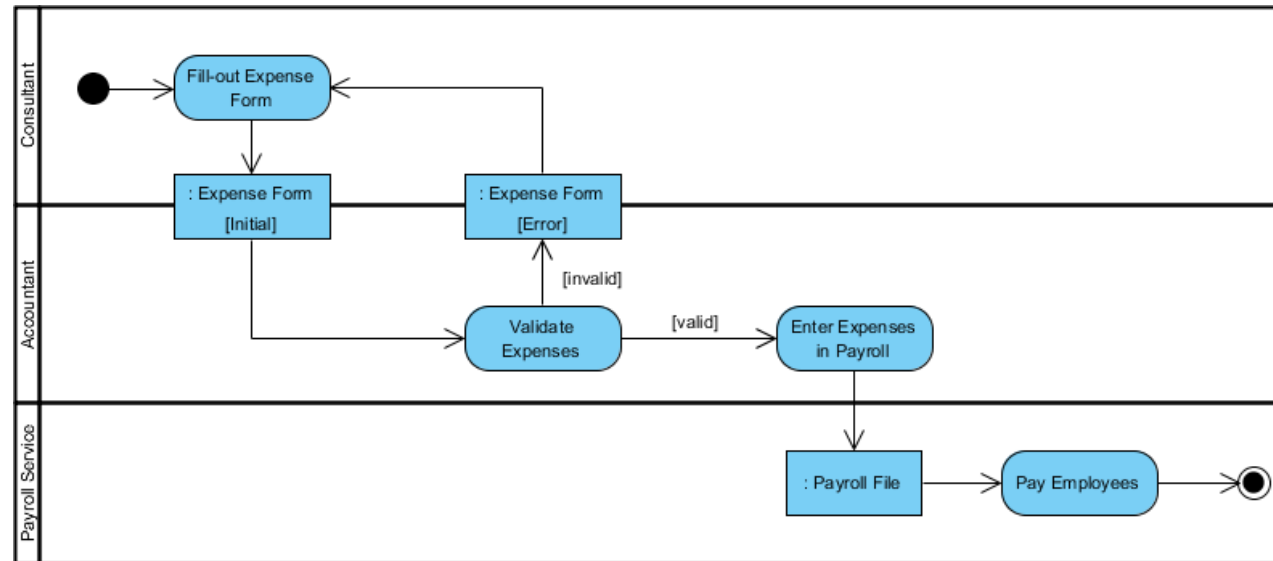




Behavioral Diagram

Activity Diagram - Swinlane

- Swimlane adalah cara untuk mengelompokkan aktivitas yang dilakukan oleh aktor yang sama pada diagram aktivitas atau diagram aktivitas atau untuk mengelompokkan aktivitas dalam satu rangkaian.
- Berikut adalah contoh diagram aktivitas swinlane untuk pemodelan Staff Expenses Submission:



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/>



Behavioral Diagram

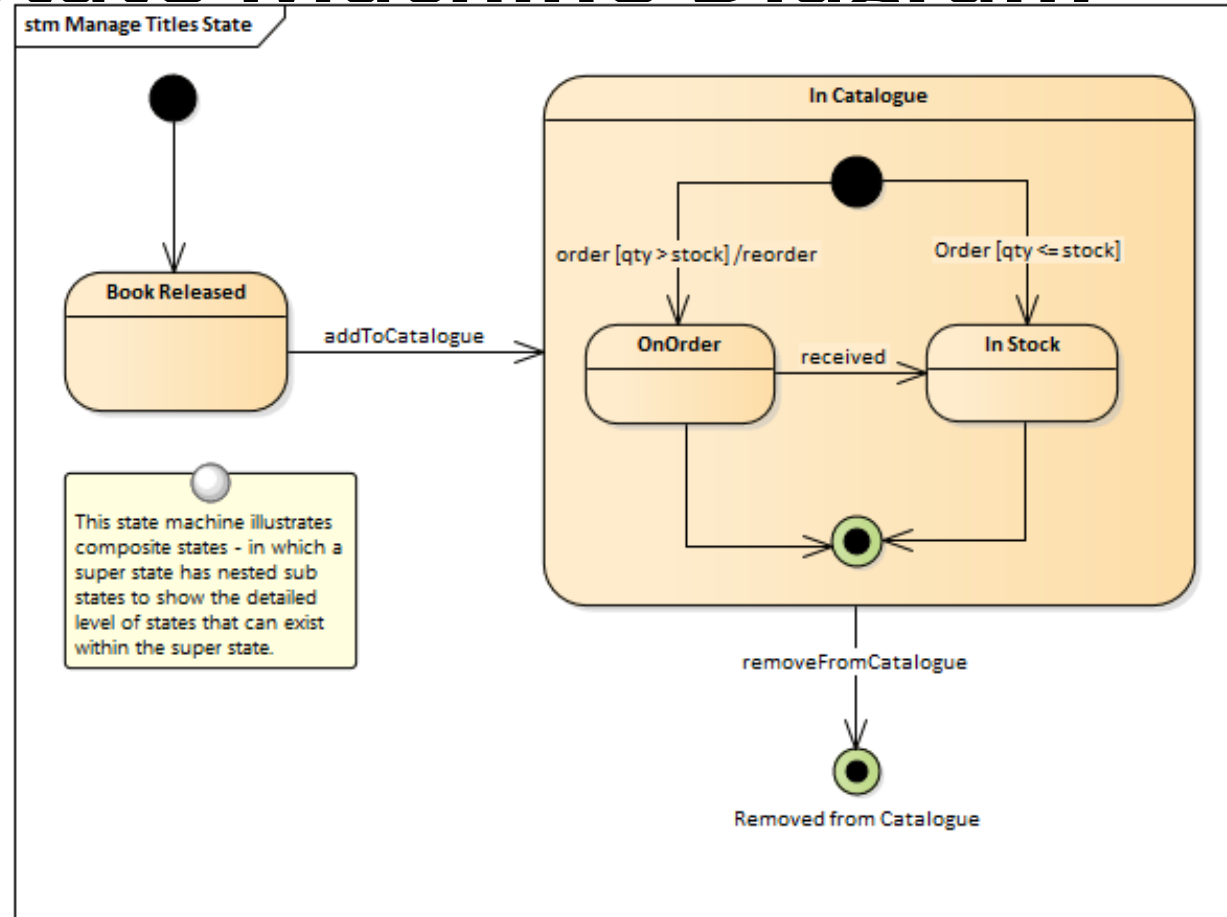
State Machine Diagram

- ⊕ State Diagram adalah jenis diagram yang digunakan dalam UML untuk menggambarkan perilaku sistem yang didasarkan pada konsep diagram keadaan oleh David Harel.
- ⊕ Diagram ini menggambarkan keadaan dan transisi yang diizinkan serta peristiwa yang mempengaruhi transisi ini.
- ⊕ Ini membantu untuk memvisualisasikan seluruh siklus hidup objek dan dengan demikian membantu memberikan pemahaman yang lebih baik tentang sistem state-based.



Behavioral Diagram

Contoh State Machine Diagram



Sumber : https://miro.medium.com/max/591/0*yTydHLClyiCaCPan.

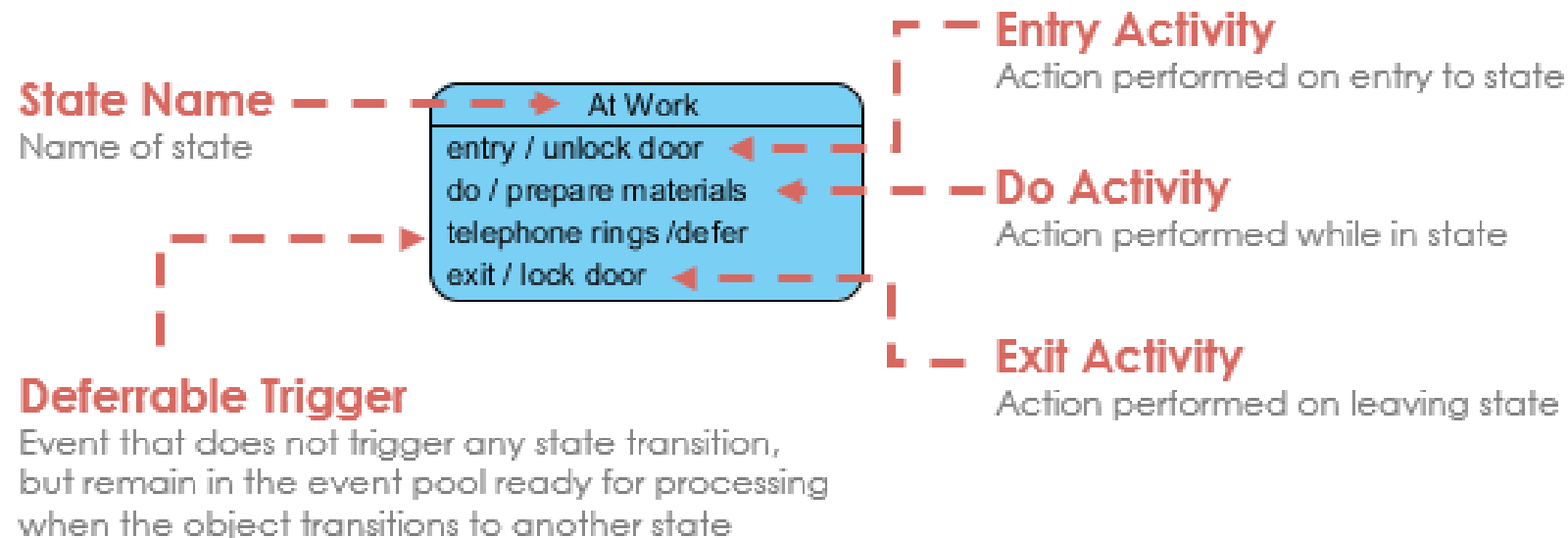


Behavioral Diagram

Notasi State Machine Diagram

Menurut Rumbaugh:

"State adalah abstraksi dari nilai atribut dan tautan dari suatu objek. Kumpulan nilai dikelompokkan menjadi satu status sesuai dengan properti yang memengaruhi perilaku kasar objek."



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/>



Interaction Diagram



Behavioral Diagram

Interaction Diagram

- ⊕ Interaction diagram merupakan sub-klasifikasi dari behavioral diagram
- ⊕ Interaction diagram digunakan dalam UML untuk membangun komunikasi antar objek.
- ⊕ Diagram ini tidak memanipulasi data yang terkait dengan jalur komunikasi tertentu.
- ⊕ Terdapat 3 jenis interaction diagram:
 1. Sequence Diagram
 2. Communication Diagram
 3. Interaction Overview Diagram
 4. Timing Diagram



Behavioral – Interaction Diagram

Sequence Diagram

- ⊕ Sequence Diagram: diagram yang menggambarkan interaksi *user* dengan sistem secara sekuensial (berurutan).
- ⊕ Sequence diagram: menjelaskan interaksi objek yang berdasarkan urutan waktu, sequence diagram juga dapat menggambarkan urutan atau tahapan yang harus dilakukan untuk dapat menghasilkan sesuatu.

Sequence Diagram menangkap:

- ⊕ Interaksi yang terjadi dalam kolaborasi yang mewujudkan kasus penggunaan atau operasi (diagram instance atau diagram umum)
- ⊕ Interaksi tingkat tinggi antara pengguna sistem dan sistem, antara sistem dan sistem lain, atau antara subsistem (kadang-kadang dikenal sebagai diagram urutan sistem)



Behavioral – Interaction Diagram

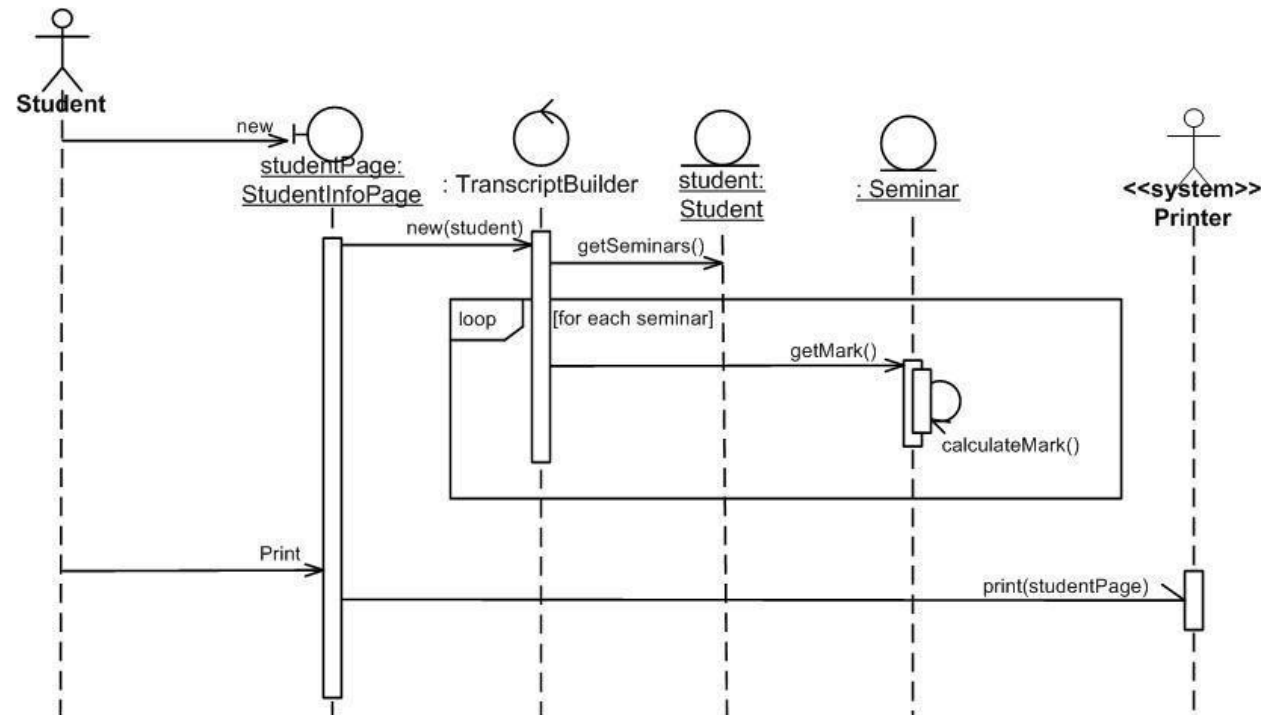
Tujuan Sequence Diagram

- ⊕ Model interaksi tingkat tinggi antara objek aktif dalam sistem
- ⊕ Buat model interaksi antara contoh objek dalam kolaborasi yang mewujudkan kasus penggunaan
- ⊕ Buat model interaksi antara objek dalam kolaborasi yang mewujudkan operasi
- ⊕ Salah satu model interaksi umum (menampilkan semua kemungkinan jalur melalui interaksi) atau contoh spesifik dari interaksi (hanya menampilkan satu jalur melalui interaksi)



Behavioral – Interaction Diagram

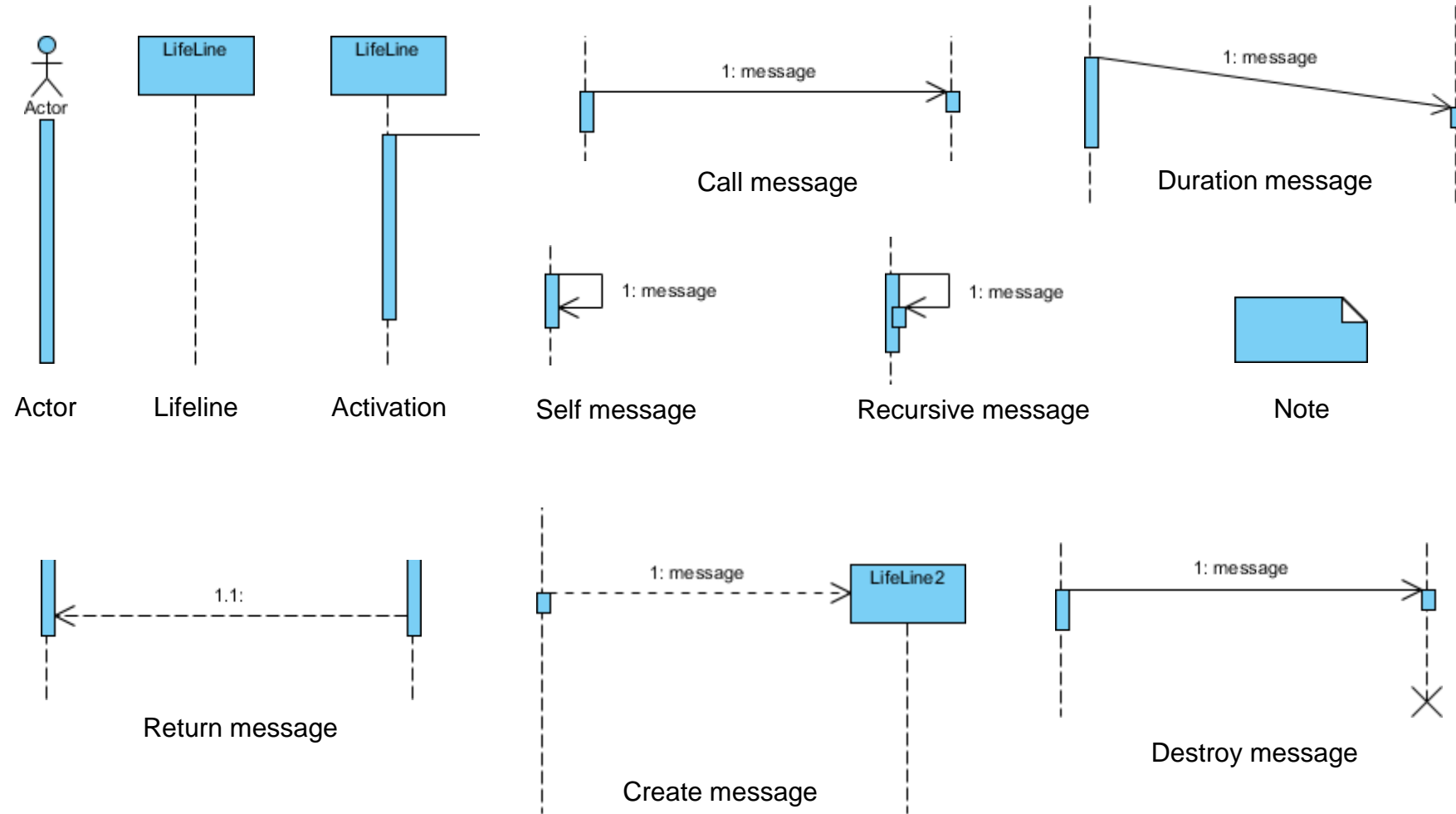
Contoh Sequence Diagram



Sumber : https://miro.medium.com/max/700/0*dKjh7DrsAke5fOcS.



Notasi Sequence Diagram

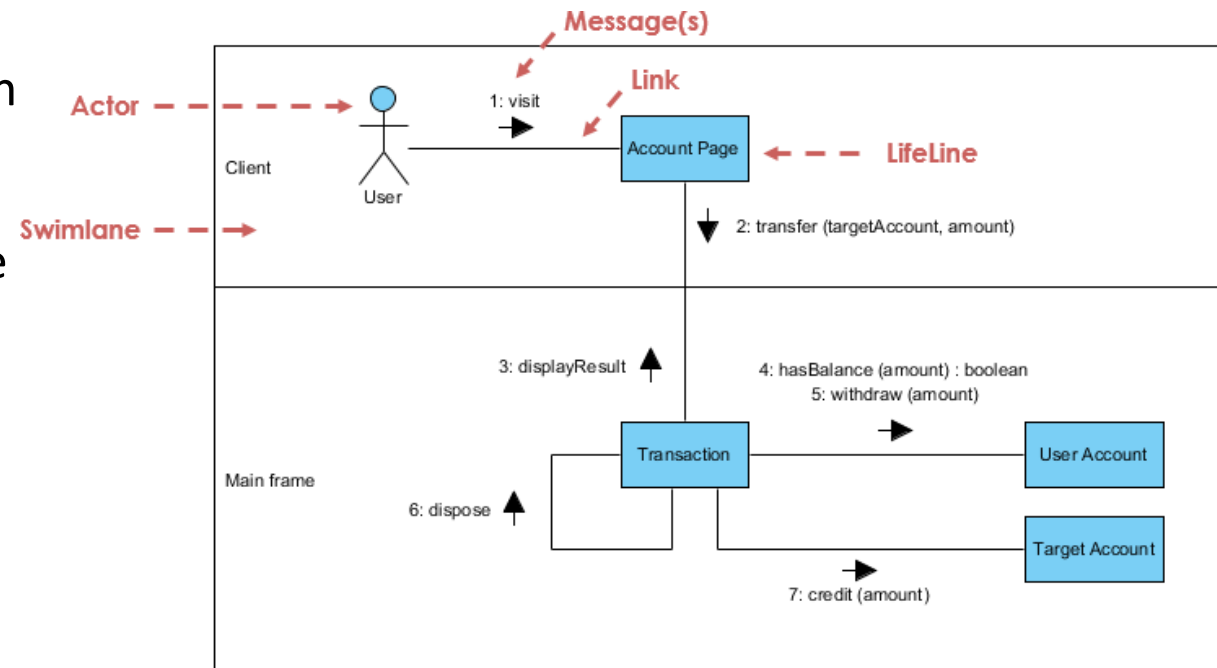




Behavioral – Interaction Diagram

Communication Diagram

- + Mirip dengan Sequence Diagram, Communication Diagram juga digunakan untuk memodelkan perilaku dinamis dari use case.
- + Diagram ini lebih difokuskan pada menampilkan kolaborasi objek daripada urutan waktu



Contoh Communication Diagram

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-communication-diagram/>

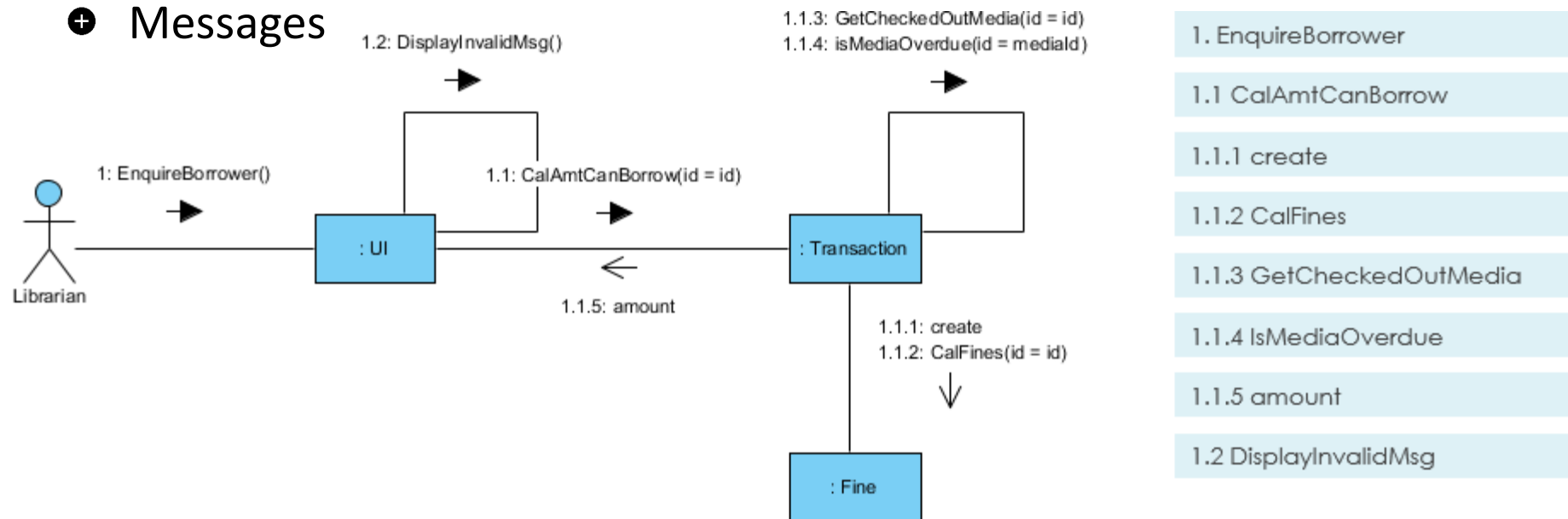


Behavioral – Interaction Diagram

Elemen Communication Diagram

Terdapat beberapa elemen di Communication Diagram:

- + Objects
- + Links
- + Messages



Ikuti nomor urut pada diagram untuk menentukan urutan pesan antar objek

Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-communication-diagram/>



Behavioral – Interaction Diagram

Interaction Overview Diagram

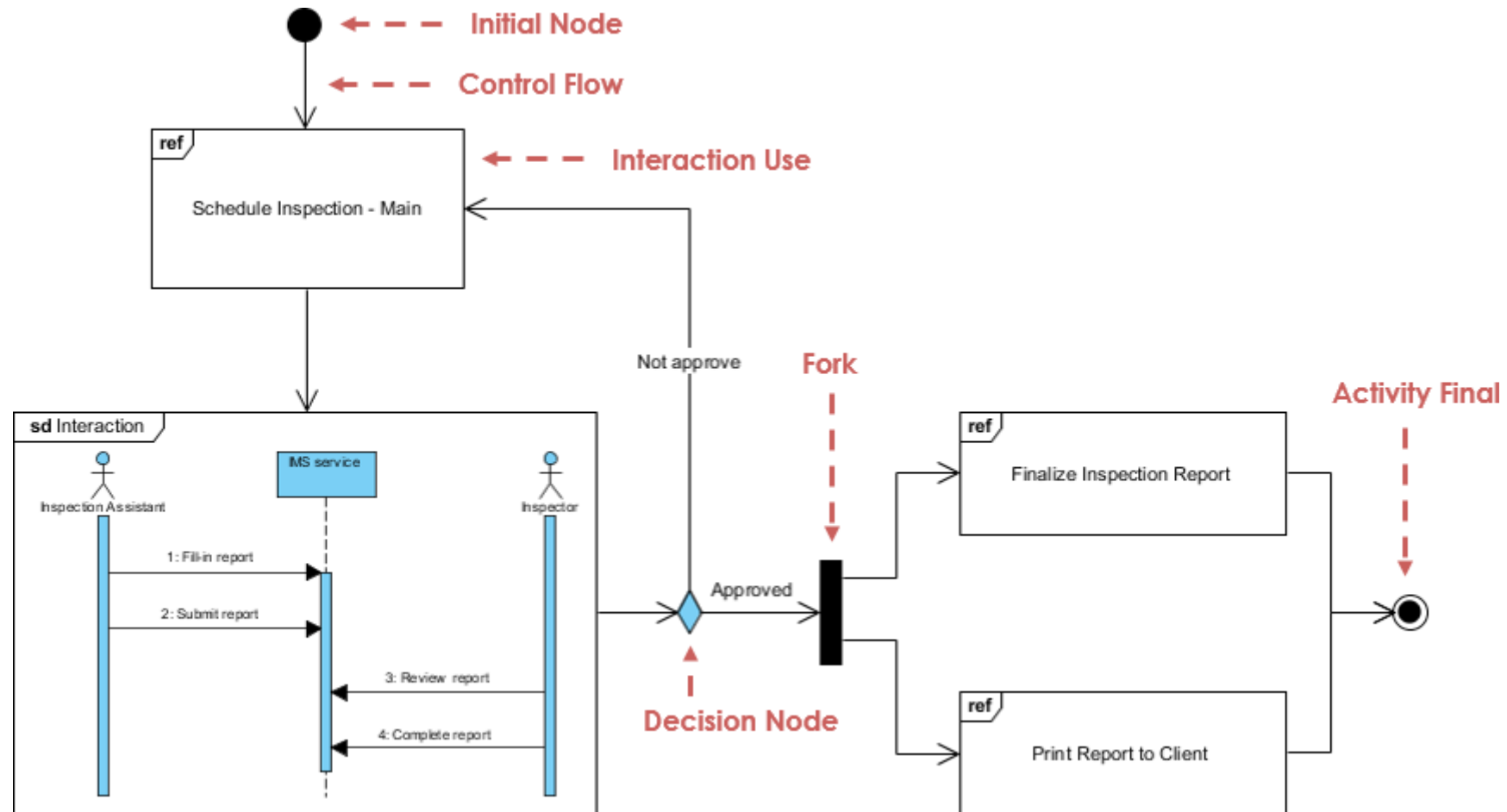
- + Interaction Overview Diagram berfokus pada gambaran umum aliran kontrol interaksi.
- + Diagram ini adalah varian dari activity diagram di mana node adalah interaksi atau kejadian interaksi.
- + Diagram ini menjelaskan interaksi di mana pesan dan jalur hidup disembunyikan.





Behavioral – Interaction Diagram

Contoh Interaction Overview Diagram



Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-interaction-overview-diagram/>



Behavioral – Interaction Diagram

Notasi Interaction Overview Diagram

Tipe Node	Simbol
Interaction Interaction Diagram dalam bentuk apa pun dapat muncul sebaris sebagai Activity Invocation.	<pre>sequenceDiagram participant Student participant Registrar Student->>Registrar: 1: register</pre>
Interaction Use Sequence Diagram yang besar dan kompleks dapat disederhanakan dengan penggunaan interaksi. Juga umum untuk menggunakan kembali beberapa interaksi antara beberapa interaksi lainnya.	<pre>sequenceDiagram participant Student participant Registrar Student->>Registrar: 1: register Student->>Registrar: 2: register Student->>Registrar: 3: register Student->>Registrar: 4: register Student->>Registrar: 5: register Student->>Registrar: 6: register Student->>Registrar: 7: register Student->>Registrar: 8: register Student->>Registrar: 9: register Student->>Registrar: 10: register Student->>Registrar: 11: register Student->>Registrar: 12: register Student->>Registrar: 13: register Student->>Registrar: 14: register Student->>Registrar: 15: register Student->>Registrar: 16: register Student->>Registrar: 17: register Student->>Registrar: 18: register Student->>Registrar: 19: register Student->>Registrar: 20: register Student->>Registrar: 21: register Student->>Registrar: 22: register Student->>Registrar: 23: register Student->>Registrar: 24: register Student->>Registrar: 25: register Student->>Registrar: 26: register Student->>Registrar: 27: register Student->>Registrar: 28: register Student->>Registrar: 29: register Student->>Registrar: 30: register Student->>Registrar: 31: register Student->>Registrar: 32: register Student->>Registrar: 33: register Student->>Registrar: 34: register Student->>Registrar: 35: register Student->>Registrar: 36: register Student->>Registrar: 37: register Student->>Registrar: 38: register Student->>Registrar: 39: register Student->>Registrar: 40: register Student->>Registrar: 41: register Student->>Registrar: 42: register Student->>Registrar: 43: register Student->>Registrar: 44: register Student->>Registrar: 45: register Student->>Registrar: 46: register Student->>Registrar: 47: register Student->>Registrar: 48: register Student->>Registrar: 49: register Student->>Registrar: 50: register Student->>Registrar: 51: register Student->>Registrar: 52: register Student->>Registrar: 53: register Student->>Registrar: 54: register Student->>Registrar: 55: register Student->>Registrar: 56: register Student->>Registrar: 57: register Student->>Registrar: 58: register Student->>Registrar: 59: register Student->>Registrar: 60: register Student->>Registrar: 61: register Student->>Registrar: 62: register Student->>Registrar: 63: register Student->>Registrar: 64: register Student->>Registrar: 65: register Student->>Registrar: 66: register Student->>Registrar: 67: register Student->>Registrar: 68: register Student->>Registrar: 69: register Student->>Registrar: 70: register Student->>Registrar: 71: register Student->>Registrar: 72: register Student->>Registrar: 73: register Student->>Registrar: 74: register Student->>Registrar: 75: register Student->>Registrar: 76: register Student->>Registrar: 77: register Student->>Registrar: 78: register Student->>Registrar: 79: register Student->>Registrar: 80: register Student->>Registrar: 81: register Student->>Registrar: 82: register Student->>Registrar: 83: register Student->>Registrar: 84: register Student->>Registrar: 85: register Student->>Registrar: 86: register Student->>Registrar: 87: register Student->>Registrar: 88: register Student->>Registrar: 89: register Student->>Registrar: 90: register Student->>Registrar: 91: register Student->>Registrar: 92: register Student->>Registrar: 93: register Student->>Registrar: 94: register Student->>Registrar: 95: register Student->>Registrar: 96: register Student->>Registrar: 97: register Student->>Registrar: 98: register Student->>Registrar: 99: register Student->>Registrar: 100: register</pre>



Behavioral – Interaction Diagram

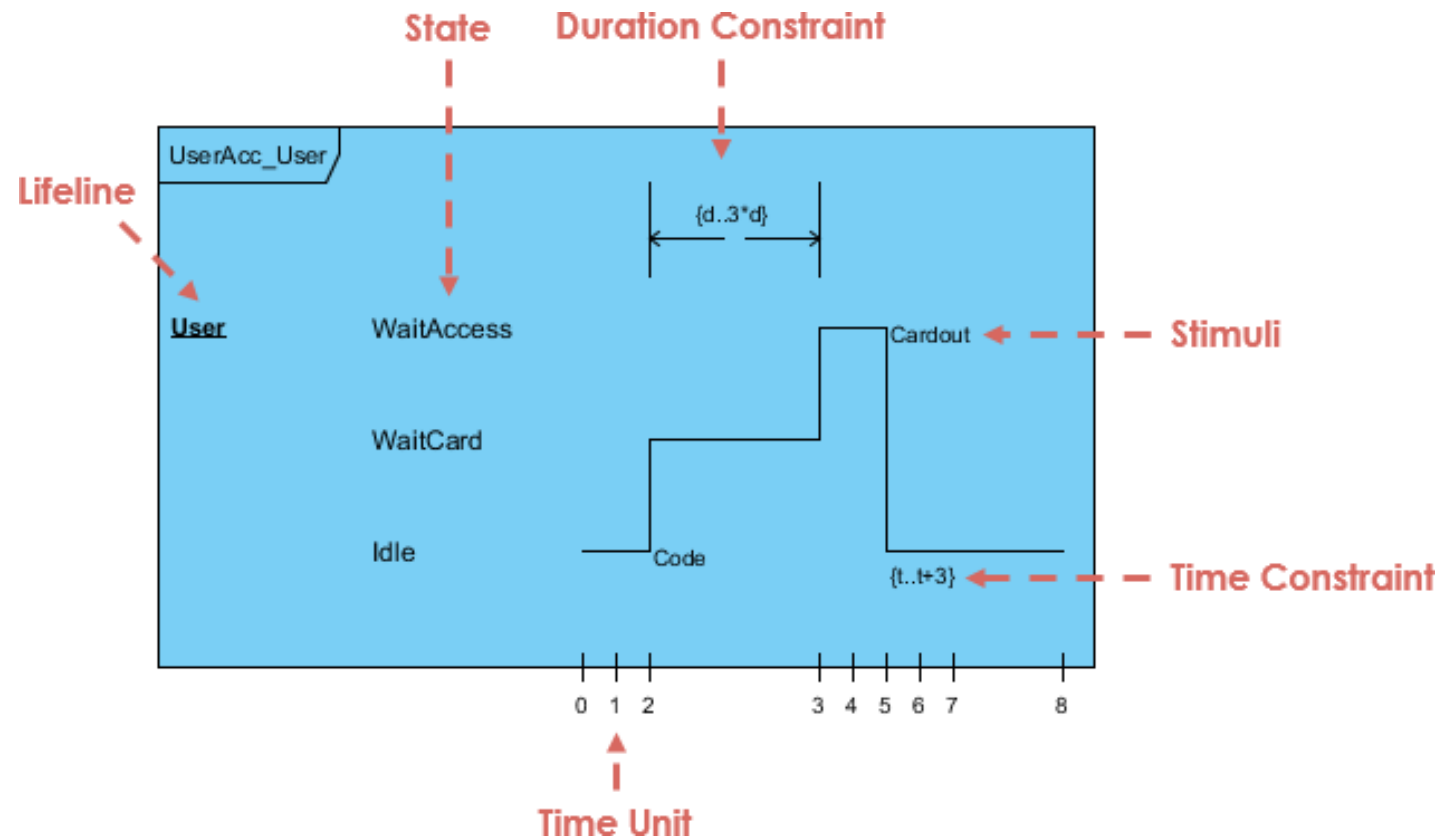
Timing Diagram

- ⊕ Timing Diagram menunjukkan perilaku objek dalam periode waktu tertentu.
- ⊕ Diagram ini adalah bentuk khusus dari sequence diagram.
- ⊕ Perbedaan antara timing diagram dan sequence diagram adalah sumbu-sumbu yang dibalik sehingga waktu bertambah dari kiri ke kanan dan garis kehidupan ditampilkan dalam kompartemen terpisah yang disusun secara vertikal.



Behavioral – Interaction Diagram

Contoh Timing Diagram



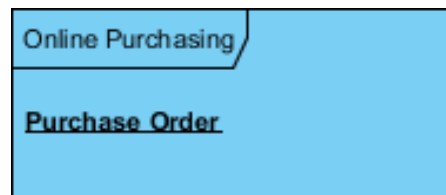
Sumber: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-timing-diagram/>



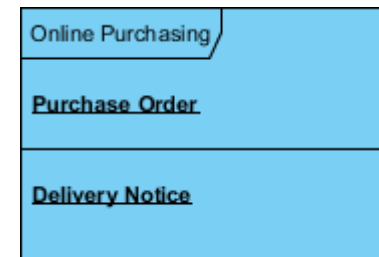
Behavioral – Interaction Diagram

Notasi Timing Diagram

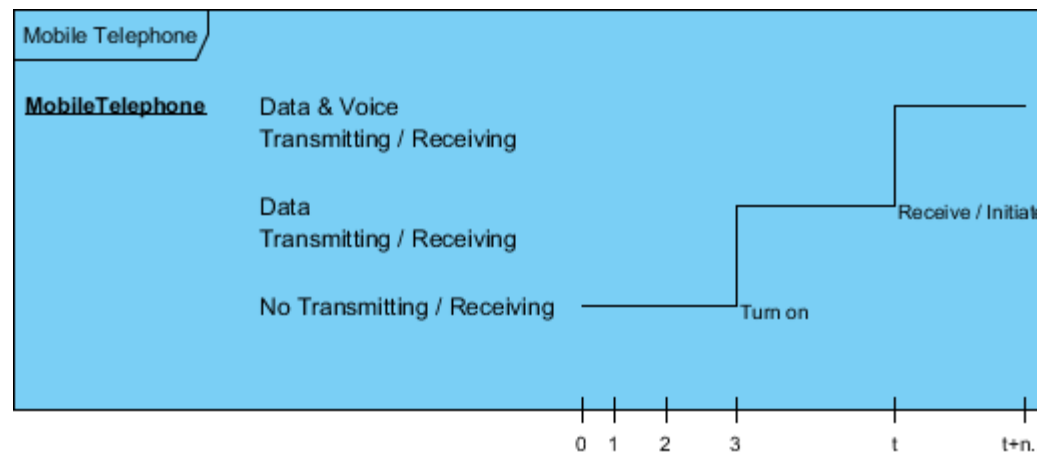
Elemen utama dari timing diagram adalah: lifeline, timeline, state or condition, message, duration constraint, timing ruler.



Lifeline



Multiple Lifeline

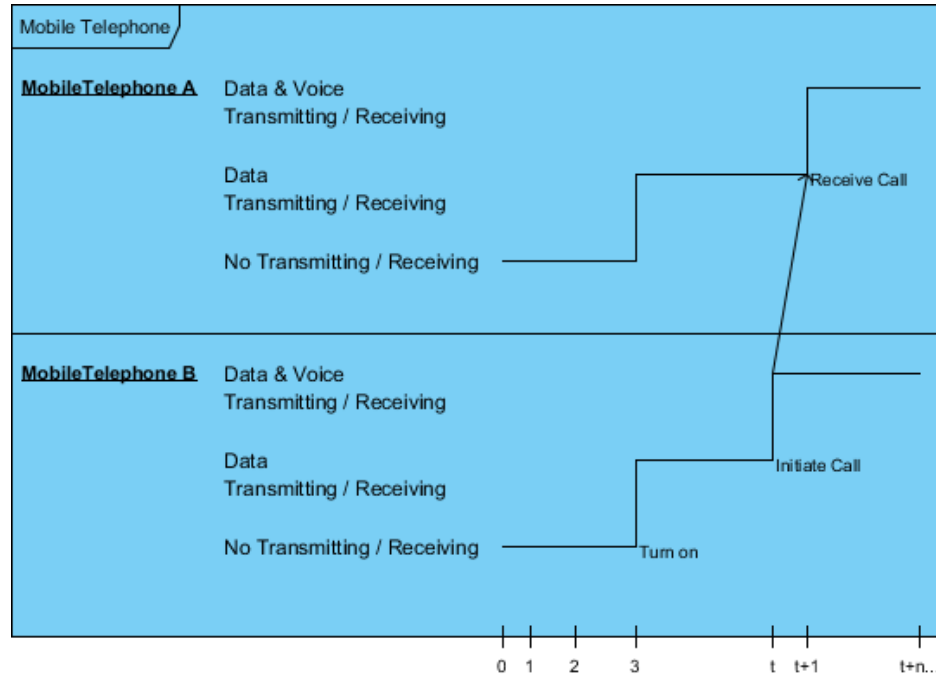


State Timeline

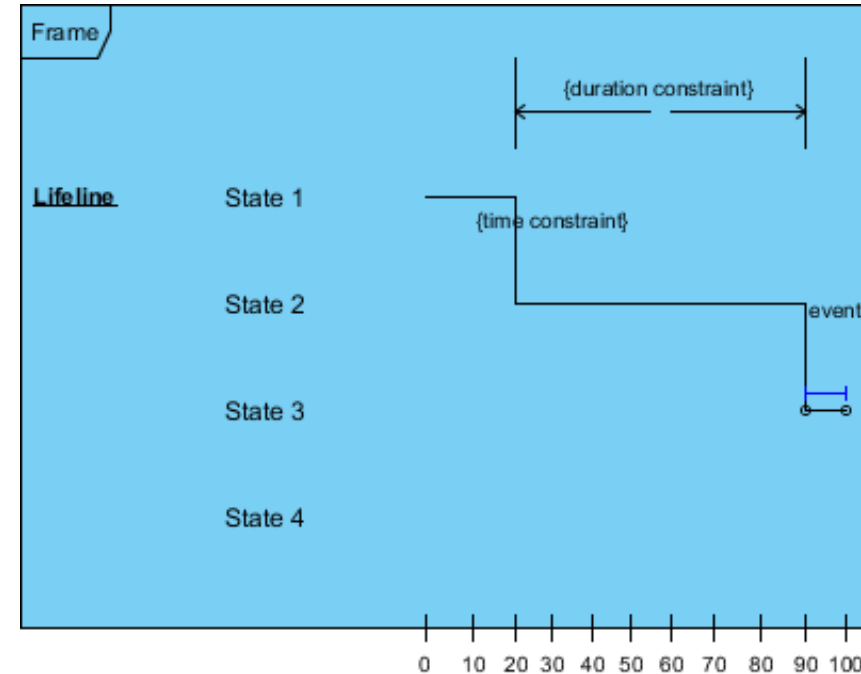


Behavioral – Interaction Diagram

Notasi Timing Diagram



Multiple Compartments

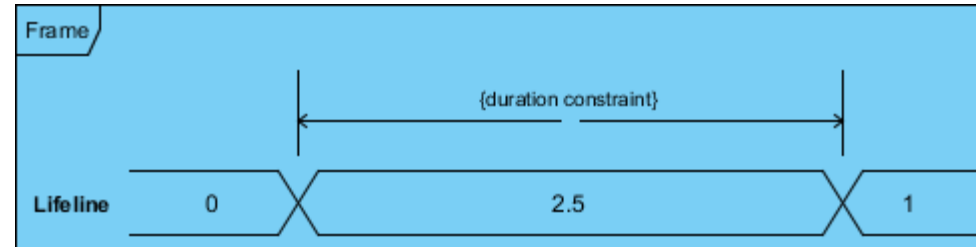


State Lifeline

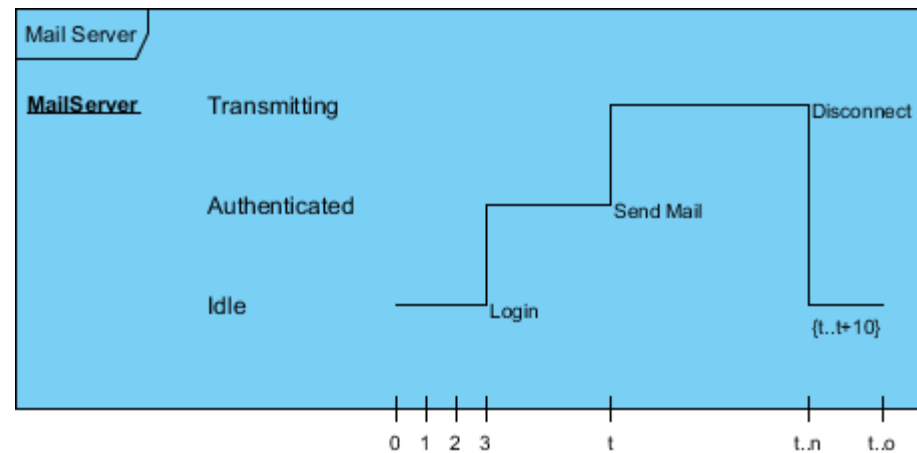


Behavioral – Interaction Diagram

Notasi Timing Diagram



Value Lifeline



Timeline dan Constraints



Referensi

- ⊕ Apa itu UML?. A. Penjelasan UML | by Andre Rahardjo | Medium. (n.d.). Retrieved August 2, 2020, from <https://medium.com/@andrerahardjo/apa-itu-uml-b8f2a8f70b89>
- ⊕ Activity Diagram in UML: Symbol, Components & Example. (n.d.). Retrieved August 2, 2020, from <https://www.guru99.com/uml-activity-diagram.html>
- ⊕ Beekman, G., & Beekman, B. (2012). *Digital Planet: Tomorrow's Technology and You 10th Edition*. Prentice Hall.
- ⊕ Model-model Diagram UML. (n.d.). Retrieved August 2, 2020, from <https://sis.binus.ac.id/2019/05/15/model-model-diagram-uml/>
- ⊕ Package Diagram Tutorial | Lucidchart. (n.d.). Retrieved August 2, 2020, from <https://www.lucidchart.com/pages/uml-package-diagram>
- ⊕ Roger S. Pressman. *Software Engineering: A Practitioner's Approach 7th Edition*. McGraw Hill Higher Education. 2010.
- ⊕ UML Diagram - Everything You Need to Know About UML Diagrams. (n.d.). Retrieved August 2, 2020, from <https://www.smartdraw.com/uml-diagram/>
- ⊕ UML Use Case Diagram Tutorial | Lucidchart. (n.d.). Retrieved August 2, 2020, from <https://www.lucidchart.com/pages/uml-use-case-diagram>
- ⊕ Visual Paradigm. What is UML? Retrieved 17 August 2020, Online: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>



**ANY
QUESTIONS?**



Tugas

Buat 3 UML (2 dari structure diagram, 1 dari behaviour diagram) pilih dari 11 jenis UML yang dijelaskan pada slide.

- Kerjakan secara berkelompok yang terdiri dari minimal 2 orang sampai 3 orang anggota
- Pilih satu studi kasus dari yang disediakan
- Didalam judul ada aktor aktor yang terlibat dalam sistem (dalam kurung) boleh lebih dari itu jika dibutuhkan aktor tambahan, tetapi yang didalam kurung wajib terlibat
- Submit dengan format PDF, format judul / topik, anggota kelompok, UML - UML yang dibuat
- Submit satu orang anggota kelompok saja di halaman live, dengan mencatat juga siapa yang mengumpulkan di spreadsheet
- Deadline sebelum pertemuan minggu depan



Sesi Berakhir
TERIMA KASIH