UNIKOM
Universitas Komputer Indonesia
Quality is Our Tradition

MATERI 2 - PENGANTAR RUP & UML

ANALISIS & PERANCANGAN SISTEM INFORMASI -2 (APSI-2)

ANNISA PARAMITHA F., S.KOM., M.KOM

PRODI SI & MI – UNIKOM

Pengantar RUP

- Definisi RUP
- Kelebihan RUP
- Fase RUP

UML

- Definisi UML
- Klasifikasi UML
- Diagram UML

DEFINISI RUP

Rational Unified Process atau dikenal dengan proses *iterative* dan *incremental* merupakan sebuah proses pengembangan perangkat lunak yang dilakukan secara *iterative* (berulang) dan inkrementasl (bertahap dengan proses menaik).

RUP (*Rational Unified Process*) adalah tahapan pengembangan sistem secara iteratif khusus untuk pemrograman berorientasi objek

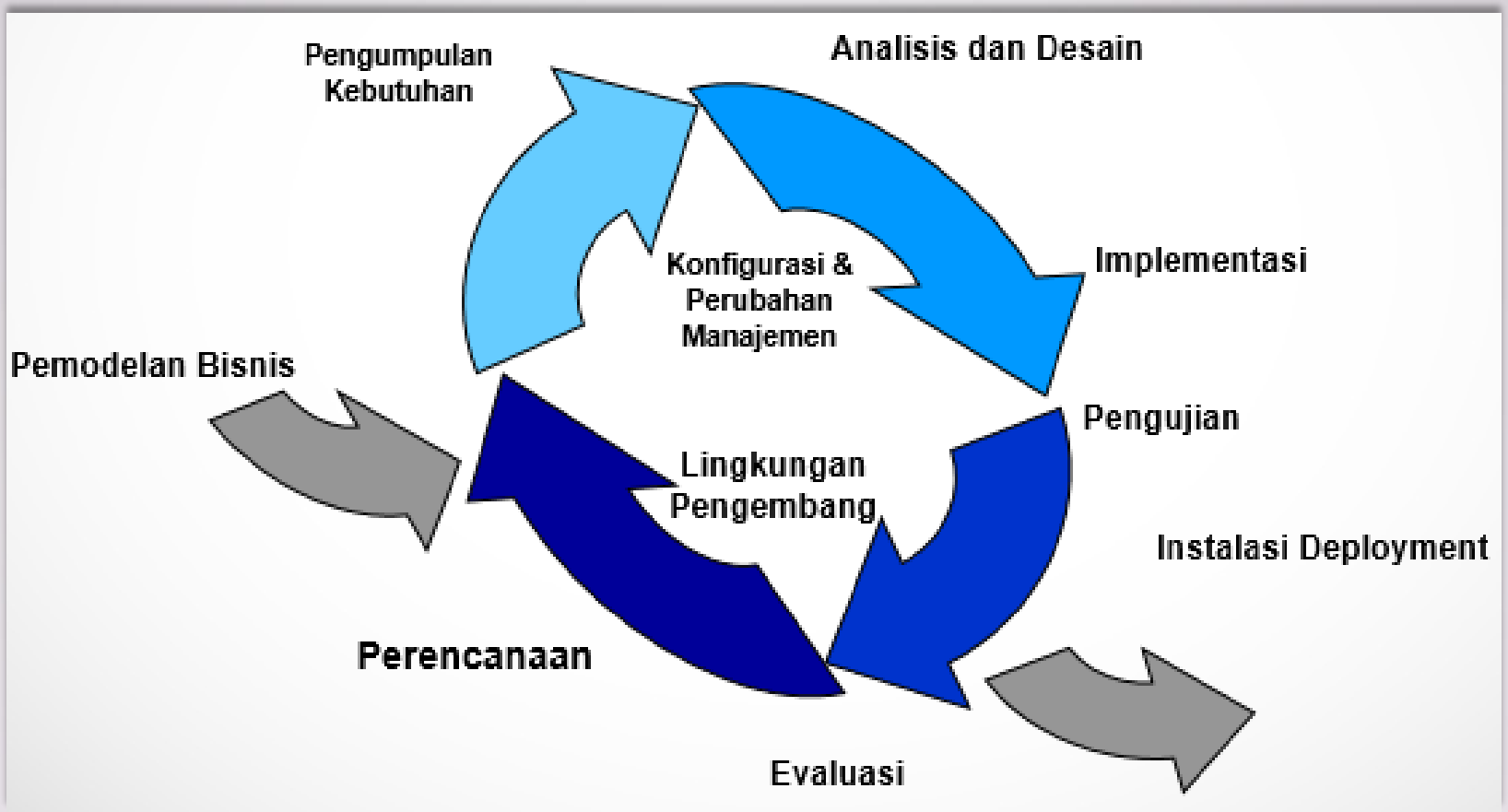
RUP menggunakan konsep *object oriented, dengan* aktifitas yang berfokus pada pengembangan model dengan menggunakan *Unified Model Language (UML)*

KELEBIHAN RUP

KELEBIHAN RUP

1. RUP mengakomodasi perubahan kebutuhan perangkat lunak.
2. Integrasi bukanlah sebuah proses besar dan cepat ('big bang')
3. Risiko biasanya ditemukan atau dialamatkan selama pada proses integrasi awal.
4. Manajemen berarti membuat perubahan taktik pada produk.
5. Mendukung fasilitas penggunaan kembali.
6. Kecacatan dapat ditemukan dan diperbaiki pada beberapa iterasi
7. Lebih baik menggunakan 'anggota proyek' dibandingkan susunan secara seri pada tim proyek
8. Anggota tim belajar selama proses proyek berjalan
9. Pengembangan perangkat lunak dapat diperbaiki seiring proses pengembangan perangkat lunak.

FASE RUP


Pengumpulan Kebutuhan
Analisis dan Desain
Konfigurasi & Perubahan Manajemen
Implementasi
Pemodelan Bisnis
Pengujian
Lingkungan Pengembang
Instalasi Deployment
Perencanaan
Evaluasi

FASE RUP

Metode dari RUP ada empat fase sebagai berikut :

1. Inception

Pada tahap ini pengembang mendefinisikan batasan kegiatan :

- Melakukan analisis kebutuhan user.
- Melakukan perancangan awal perangkat lunak (perancangan arsitektural dan use case).
- Menentukan kebutuhan akan model bisnis diterapkan. (Biaya)
- Menentukan batasan – batasan dalam sebuah project.
- Memahami ruang lingkup dari proyek (termasuk biaya, waktu, kebutuhan resiko dsb)
- Hasil yg diharapkan : Pendefinisian Ruang lingkup, Perkiraan Biaya, dan perkiraan Jadwal.

FASE RUP (LANJUTAN)

2. *Elaboration* (perluasan/ perencanaan)

- Pada tahap ini dilakukan perancangan perangkat lunak mulai dari menspesifikasi fitur perangkat lunak hingga perilisan prototype versi Betha dari perangkat lunak.
- Pengimplementasian use case, sebagai perwujudan dari arsitektur system software.
- Menghilangkan kemungkinan-kemungkinan terbesar yang memungkinkan timbulnya sebuah resiko dalam proses perkembangan project itu sendiri, dikarenakan jika sampai terjadi perubahan terhadap project dalam fase berikutnya,akan menyulitkan pengembang project tersebut untuk kembali meninjau ulang.
- Merencanakan fase berikutnya yaitu construction

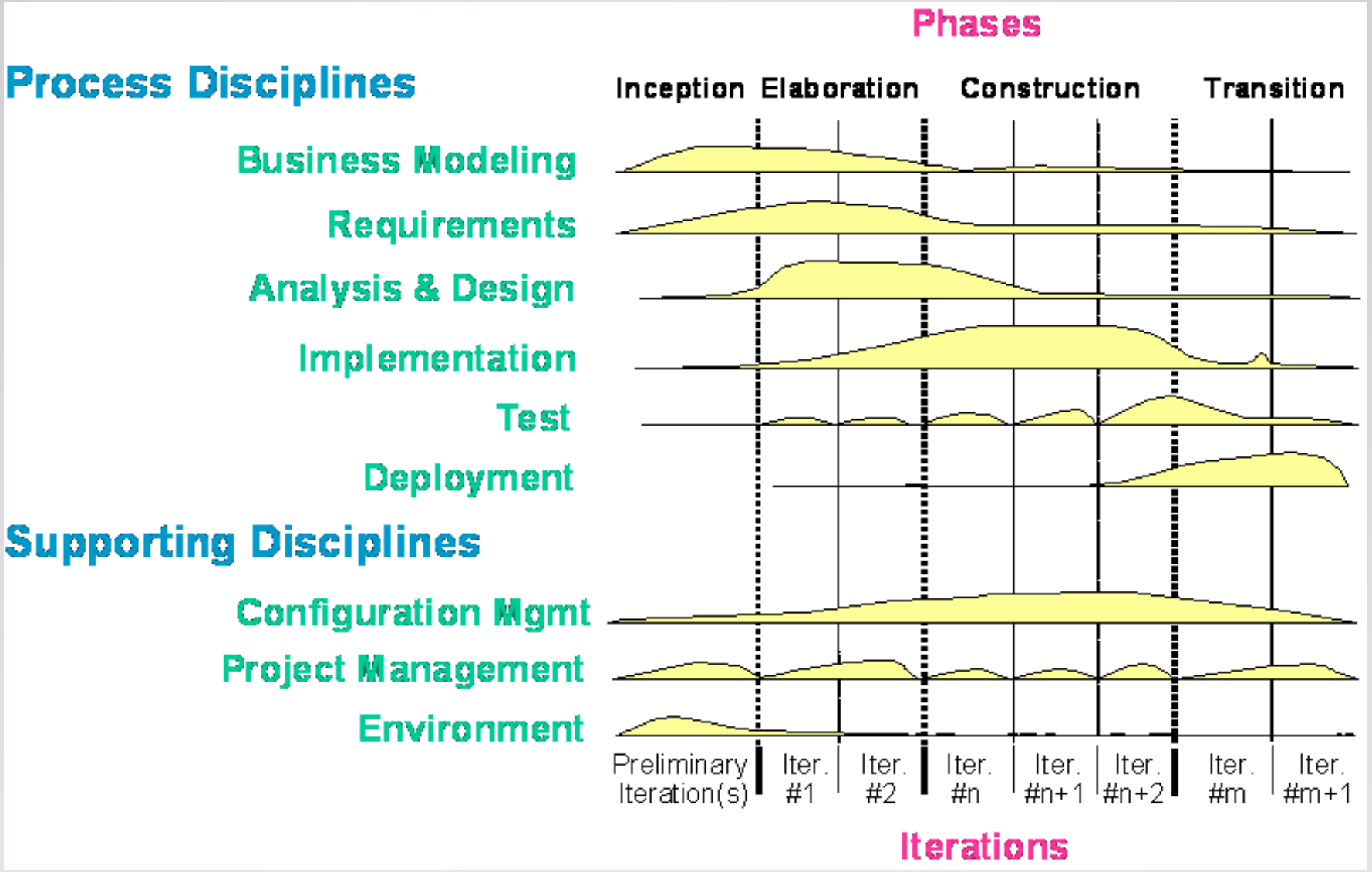
FASE RUP (LANJUTAN)

3. *Construction* (konstruksi)

- Aktivitas yang terjadi dalam fase ini antara lain design, implementasi, dan serta pengujian software tersebut
- Tahap ini fokus pada pengembangan komponen dan fitur -fitur sistem.
- Tujuan utama dalam fase ini adalah perkembangan sebuah project dalam bentuk coding.

4. *Transition* (transisi)

- Tahap ini lebih pada *deployment* atau instalasi sistem agar dapat dimengerti oleh user.
- Sosialisasi perangkat lunak
- Aktifitas pada tahap ini termasuk pada pelatihan user, pemeliharaan dan pengujian sistem apakah sudah memenuhi harapan user.


Phases
Process Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Supporting Disciplines
Configuration Mgmt
Project Management
Environment
Preliminary Iteration(s)
Iter. #1
Iter. #2
Iter. #n
Iter. #n+1
Iter. #n+2
Iter. #m
Iter. #m+1
Iterations

Bagian Business Modeling

Tujuan utama dalam business modeling di sini adalah untuk memungkinkan adanya komunikasi dan pengertian yang lebih baik dari business engineering dan software engineering.

Fase-fase yang terlibat dalam business modeling :

a. **Inception** : pertama kalinya business modeling dideklarasikan dan difenisikan.

b. **Elaboration** : peninjauan kembali terhadap requirement bisnis untuk meminimalisasikan terjadinya perubahan pada tahap selanjutnya yaitu construction.

c. **Construction** penerapan dari business modeling yang telah terdefinisi dalam bentuk coding.

d. **Transition** : dimungkinkan apablia terjadi kesepakatan antara developer dengan end users dalam perawatan software yang telah dibuat.

Bagian Requirement

Tujuan dari tahap ini adalah mendeskripsikan apa yang harus dilakukan oleh system dan memungkinkan terjadinya kesepakatan antara customer dan developer tentang hal itu.

Fase-fase yang terlibat antara lain :

1. **Inception** : requirement dari software pertama kali dibahas. Lebih terfokus pada requirement pengembangan software yang akan dipakai.
2. **Elaboration** : mengurangi / meninjau kembali requirement dari software, dan dimungkinkan terjadi pergantian requirement dalam software yang akan dikembangkan.
3. **Construction** : perwujudan requirement yang ada dalam bentuk coding dari software yang dikembangkan beserta pengujian apakah software sudah memenuhi requirement awal.
4. **Transition** : requirement dalam fase ini berupa requirement dari end users untuk menambah aplikasi software,perawatan software.

Bagian Analysis dan Design

Tujuan dalam tahap ini adalah untuk menunjukkan bagaimana project akan diwujudkan dalam fase implementasi kelak. Hasil dari tahap ini adalah model design.

Fase-fase yang terlibat :

1. **Inception** : pembahasan tentang business modeling dan requirement.
2. **Elaboration** : Fase inilah yang menjadi pusat perkembangan, peninjauan kembali dilakukan pada fase ini.
3. **Construction**: Project dikembangkan dalam bentuk coding

Bagian Implementasi

Tujuan dari implementasi di sini adalah mendefinisikan pengkodean secara terorganisasi, mengimplimentasikan classes dan objects dalam bentuk komponen-komponen, menguji perkembangan komponen-komponen dalam bentuk kesatuan, dan mengintegrasikan hasil-hasil dari tiap-tiap kelompok yang mengerjakan project.

Fase-fase yang terlibat :

1. **Inception** : Pada tahap ini implementasi berlaku dengan terjadinya percakapan antara end users dan developer mengenai software yang akan dikembangkan.
2. **Elaboration** : selain implementasi terhadap pembuatan use case, tahap ini juga memuat implementasi dari perkembangan perencanaan arsitektural dan sebagainya.
3. **Construction** : Implementasi dari rancangan software dilaksanakan.
4. **Transition** : Implementasi yang terjadi pada tahap ini adalah penyerahan software terhadap end users dan implementasi pada penerapan aplikasi software yang telah dikembangkan .

Bagian Testing

Tujuan dari testing adalah memverifikasi interaksi antar object, memverifikasi integrasi yang sesuai antar komponen, memverfikasi apakah semua requirement telah terpenuhi dengan benar, mengidentifikasi dan memastikan defect telah ditangani secara benar.

Fase-fase yang terlibat :

1. **Inception** : Dalam fase ini testing dilakukan apabila pemodelan bisnis dan requirement telah teridentifikasi. Testing dilakukan dengan tujuan menghasilkan kesepakatan antara end users dengan developer.
2. **Elaboration** : testing di sini merupakan testing setelah use case diimplementasikan, masih seputar tercapainya kesepakatan antara end users dengan developer.
3. **Construction** : testing kebanyakan dilakukan di akhir fase construction, karena setelah penyelesaian program-lah, testing baru dilaksanakan.
4. **Transition** : testing dilakukan sebelum penyerahan software kepada end users dengan keadaan yang sebenarnya.

Bagian Pengembangan
Tujuan dari pengembangan di sini adalah suksesnya menghasilkan suatu project dan mengantarkan project tersebut pada end users.

Fase-fase yang terlibat :

1. **Elaboration** : mulailah pengembangan tentang realitas dari software itu akan seperti apa dalam fase ini.
2. **Construction** : dalam fase ini pengembangan software secara nyata terjadi dengan adanya coding.
3. **Transition** : fase yang paling berpengaruh karena adanya penyerahan software dari developer kepada end users.

Bagian Managemen Konfigurasi dan Perubahan

Area-area spesifik dalam bagian ini adalah management konfigurasi, merubah management requirement, status dan management pengukuran.

Fase-fase yang terlibat :

1. **Inception** : terjadi diskusi mengenai konfigurasi dari system software yang diinginkan.
2. **Elaboration** : masih membahas seputar konfigurasi software, ditambah dengan perubahan-perubahan yang terjadi, terkait dengan diminimalisasikannya perubahan dalam fase selanjutnya.
3. **Construction** : dalam fase inilah akan terlihat jelas penerapan dari konfigurasi yang telah ditentukan, dan mungkin tidaknya konfigurasi yang diinginkan terpenuhi.
4. **Transition** : konfigurasi yang ada adalah mengenai konfigurasi system dalam keadaan yang sesungguhnya.

PENGANTAR UML

UML ***(Unified Modelling Language)*** adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem berorientasi objek.

UML disebut sebagai bahasa pemodelan bukan metode. Bahasa pemodelan (sebagian besar grafik) merupakan notasi dari metode yang digunakan untuk mendesain secara cepat.

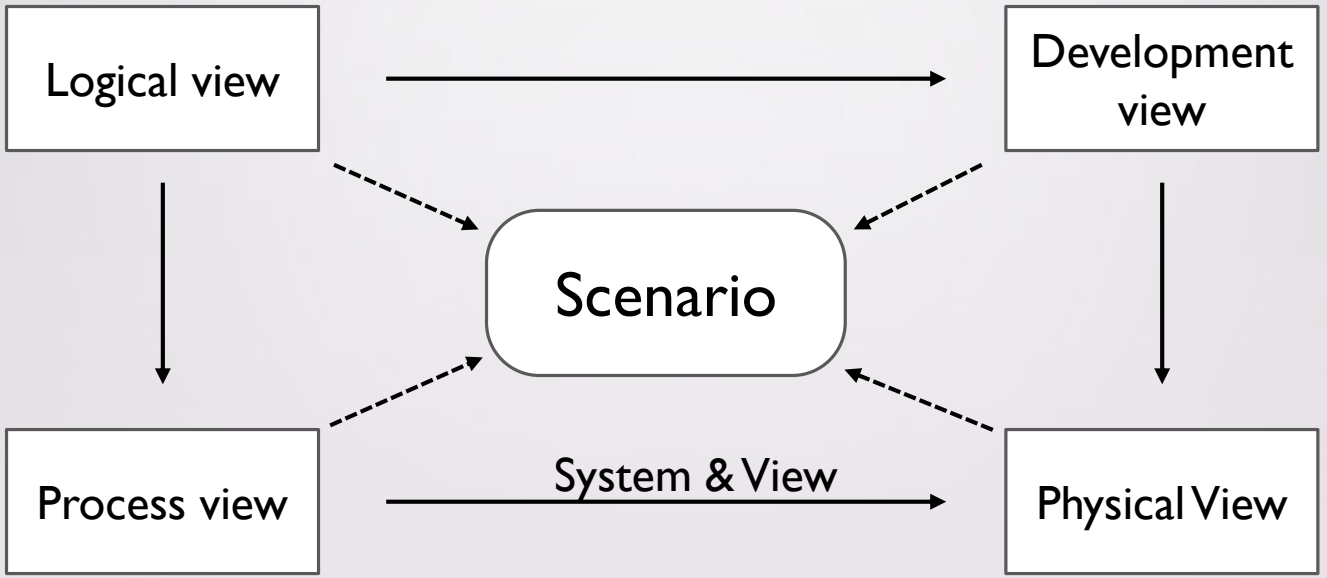
UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek.

Perbedaan model dan diagram.

Diagram menggambarkan atau mendokumentasikan beberapa aspek dari sebuah sistem. Sedangkan sebuah model menggambarkan pandangan tentang suatu sistem pada suatu tahapan tertentu dan dari perspektif tertentu. Sebuah model mungkin mengandung satu atau lebih diagram

PENGANTAR UML (LANJUTAN..)

UML dibangun atas model 4+ 1 view. Model ini didadasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam 5 *view* dimana salah satu diantaranya adalah scenario. Skenario ini memegang peran khusus untuk mengintegrasikan *content* ke *view* yang lain.



PENGANTAR UML (LANJUTAN..)

Scenario menggambarkan interaksi antar objek dan proses. Scenario ini biasa disebut juga dengan *usecase view*. *Usecase view* ini mendefinisikan kebutuhan sistem karena mengandung semua view yang lain yang mendeskripsikan aspek-aspek tertentu dari rancangan sistem.

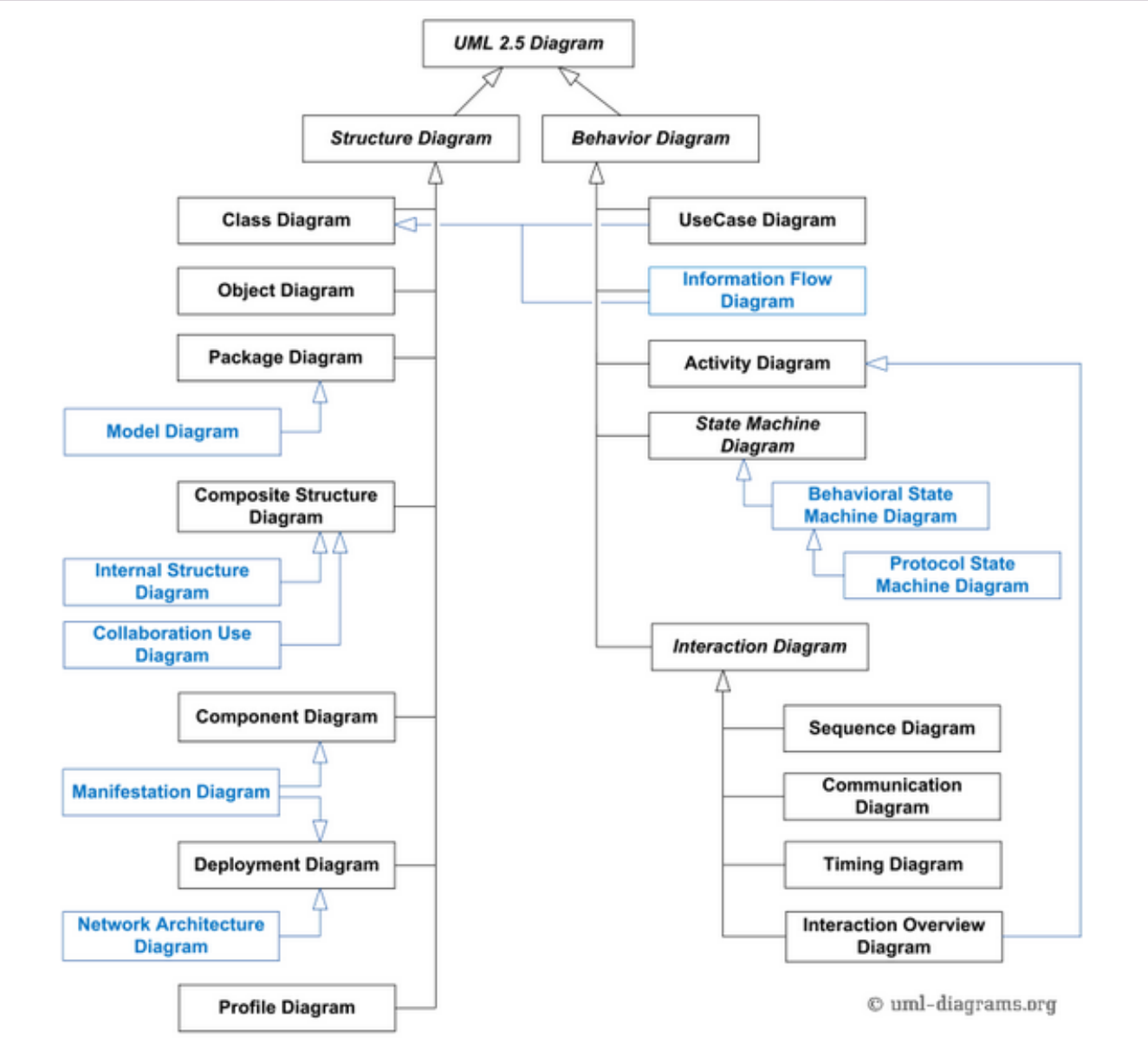
Development view menjelaskan sebuah sistem dari perspektif programmer dan terkonsentrasi ke manajemen perangkat lunak. Diagram UML yang termasuk dalam *development view* diantaranya adalah *component diagram dan package diagram*

Logical view terkait dengan fungsionalitas sistem yang dipersiapkan untuk pengguna akhir. *Logical view* berisi *object diagram*, *class diagram*, dan *composite structure diagram*.

PENGANTAR UML (LANJUTAN..)

Physical view menggambarkan sistem dari perspektif system engineer. Fokus dari ***physical view*** adalah topologi sistem perangkat lunak.

Process view berhubungan erat dengan aspek dinamis dari sistem, proses yang terjadi di sistem dan bagaimana komunikasi yang terjadi di sistem serta tingkh laku sistem saat dijalankan. Yang termasuk dalam ***process view*** adalah ***activity diagram***, ***communication diagram***, ***sequence diagram***, dan ***interaction overview diagram***.



Sumber : https://www.uml-diagrams.org/uml-25-diagrams.html

Klasifikasi UML 2.5

Pada halaman sebelumnya dapat dilihat stuktur UML 2.5

Spesifikasi UML mendefinisikan dua jenis utama diagram UML : **diagram struktur** dan **diagram perilaku.**

Diagram struktur ***(Structure Diagram)*** menunjukkan struktur statis sistem dan bagian-bagiannya pada abstraksi yang berbeda dan tingkat implementasi dan bagaimana mereka saling terkait. Elemen-elemen dalam diagram struktur mewakili konsep yang bermakna dari suatu sistem, dan dapat mencakup abstrak, dunia nyata dan konsep implementasi.

Behavior diagram menunjukkan perilaku dinamis objek dalam suatu sistem, yang dapat digambarkan sebagai serangkaian perubahan pada sistem dari waktu ke waktu.

Klasifikasi UML 2.5 (Lanjutan ....)

Diagram struktur menunjukkan struktur statis sistem dan bagian-bagiannya pada abstraksi yang berbeda dan tingkat implementasi dan bagaimana bagian-bagian tersebut saling terkait. Elemen-elemen dalam diagram struktur mewakili konsep yang bermakna dari suatu sistem, dan dapat mencakup abstrak, dunia nyata dan konsep implementasi.

Diagram struktur tidak memanfaatkan konsep yang berkaitan dengan waktu, tidak menunjukkan rincian perilaku dinamis. Namun, mereka dapat menunjukkan hubungan dengan perilaku pengklasifikasi yang ditunjukkan dalam diagram struktur.

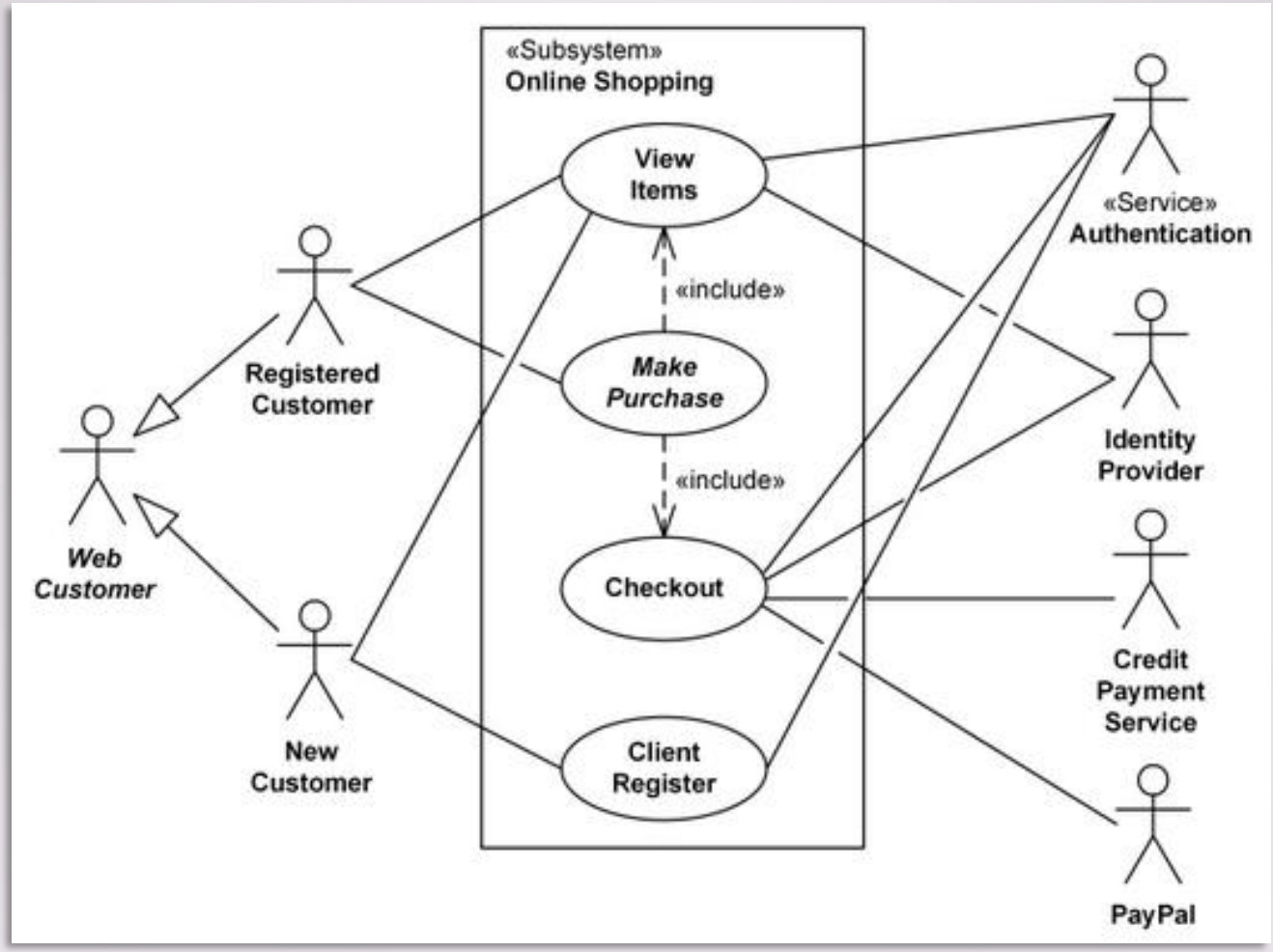
Klasifikasi UML 2.5 - (Lanjutan ....)

Diagram struktur menunjukkan struktur statis sistem dan bagian-bagiannya pada abstraksi yang berbeda dan tingkat implementasi dan bagaimana bagian-bagian tersebut saling terkait. Elemen-elemen dalam diagram struktur mewakili konsep yang bermakna dari suatu sistem, dan dapat mencakup abstrak, dunia nyata dan konsep implementasi.

Diagram struktur tidak memanfaatkan konsep yang berkaitan dengan waktu, tidak menunjukkan rincian perilaku dinamis. Namun, mereka dapat menunjukkan hubungan dengan perilaku pengklasifikasi yang ditunjukkan dalam diagram struktur.

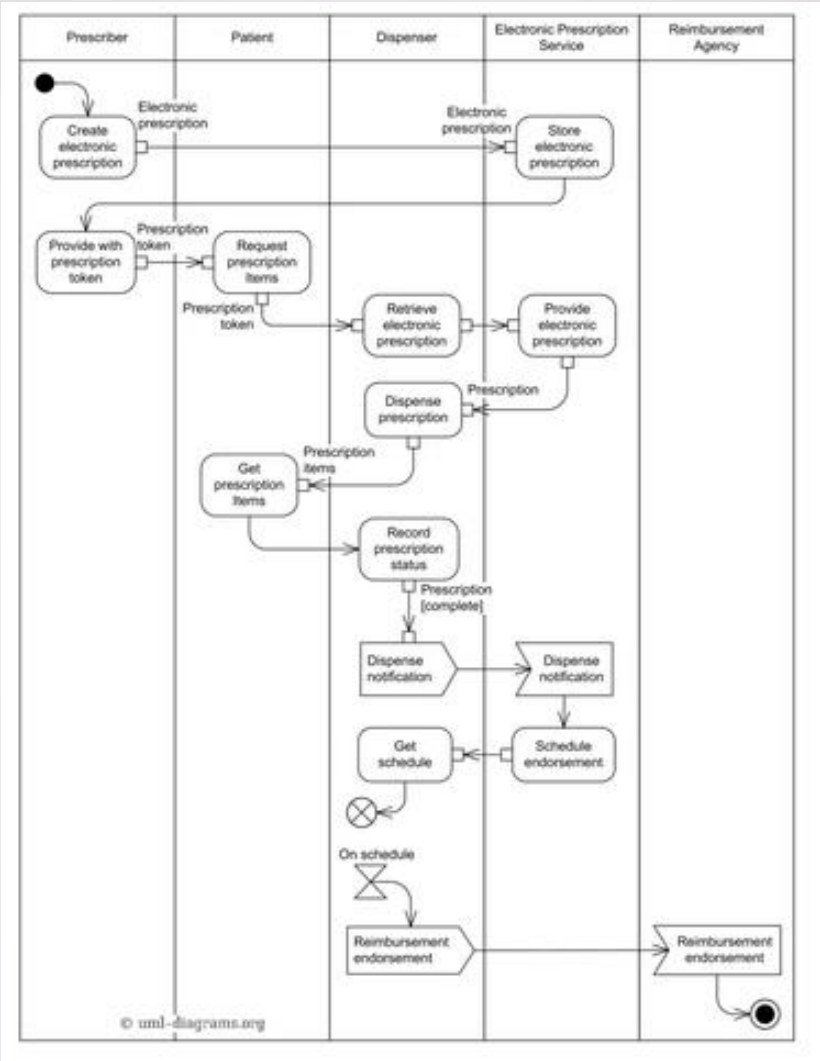
Behavior diagram menunjukkan perilaku dinamis objek dalam suatu sistem, yang dapat digambarkan sebagai serangkaian perubahan pada sistem dari waktu ke waktu.

Usecase Diagram



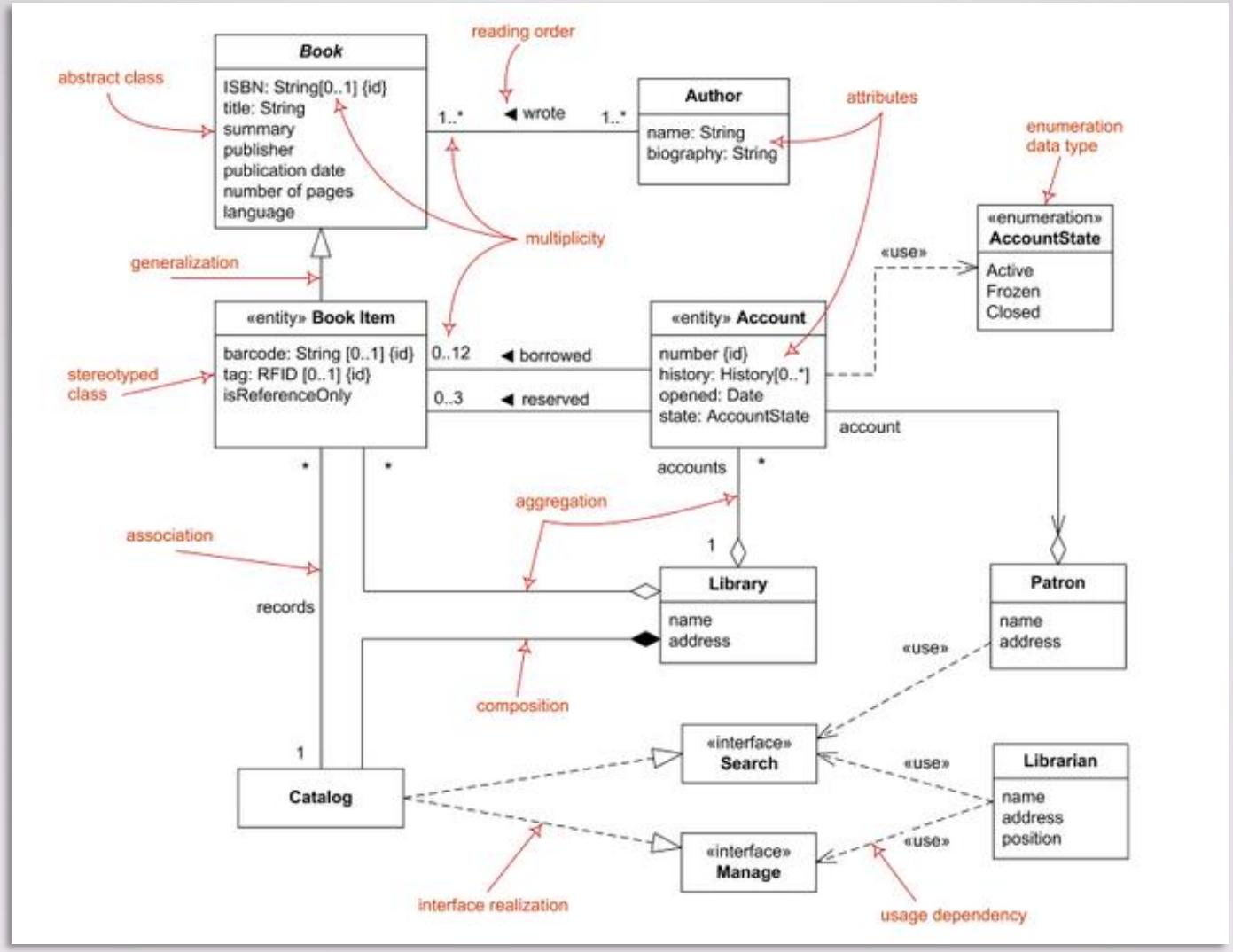
Sumber : https://www.uml-diagrams.org/uml-25-diagrams.html

Activity Diagram



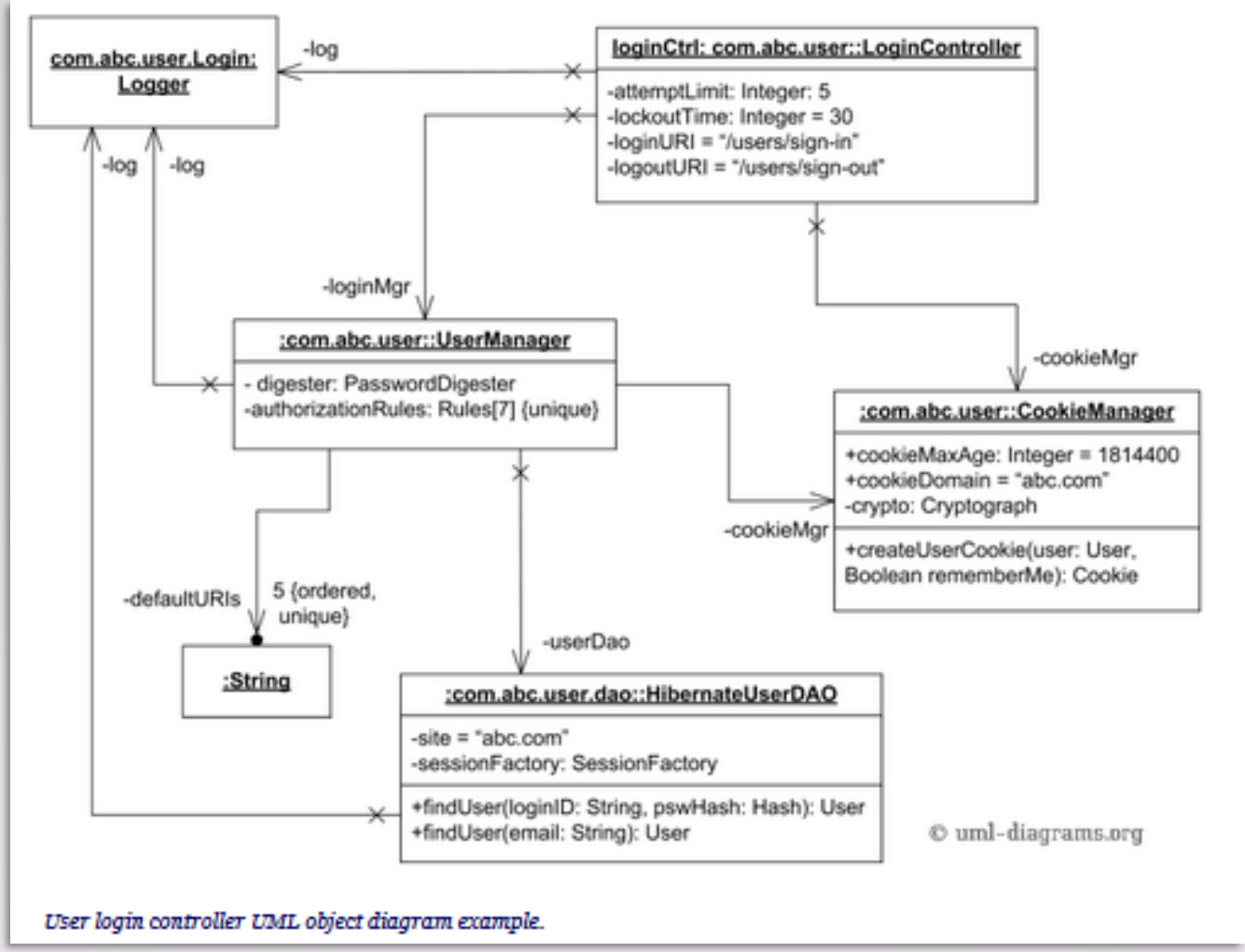
Sumber : https://www.uml-diagrams.org/uml-25-diagrams.html

Class Diagram



Sumber : https://www.uml-diagrams.org/uml-25-diagrams.html

Object Diagram



User login controller UML object diagram example.

Sumber : https://www.uml-diagrams.org/uml-25-diagrams.html

Sequence Diagram

Sumber : https://www.uml-diagrams.org/uml-25-diagrams.html