Pengembangan Aplikasi Perangkat Lunak

OOAD

Sequence Diagram

By: Augury El Rayeb (AER)

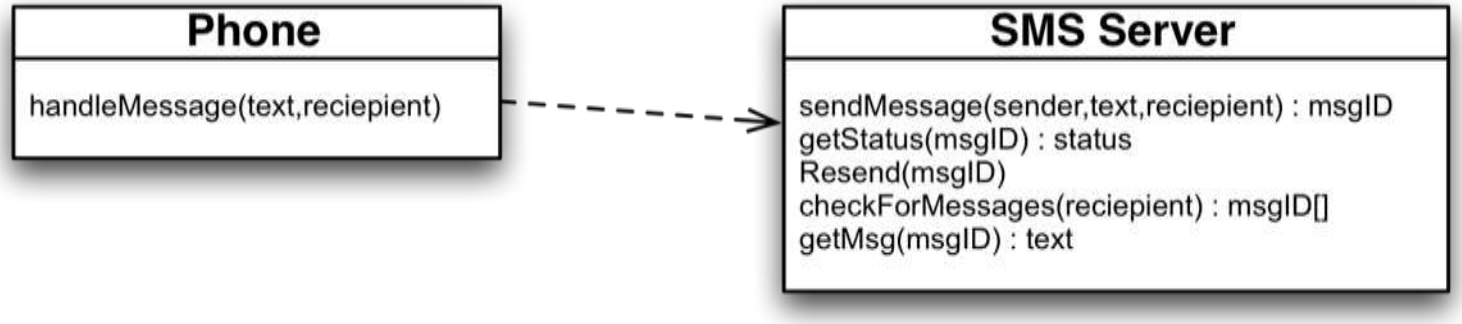
Tujuan Pertemuan

- Mahasiswa memahami fungsi sequence diagram
- Mahasiswa memahami fungsi elemen-elemen sequence diagram
- Mahasiswa mampu membuat sequence diagram untuk suatu urutan eksekusi proses

System Development Process

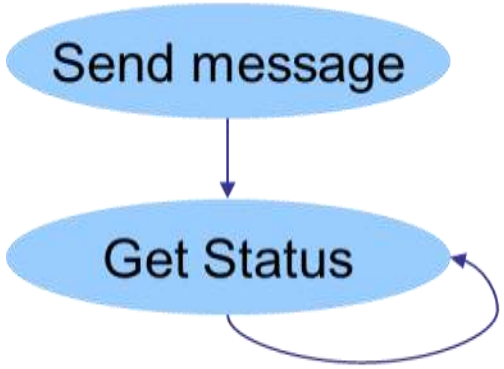
Phase	Actions	Outcome
Initiation	Raising a business need	Business documents
Analysis	Interviewing stakeholders, exploring the system environment	Organized documentation
Specification	Analyze the engineering aspect of the system, building system concepts	Logical System Model
Implementation	Program, build, unit-testing, integrate, documentation	Testable system
Testing & Integration	Integrate all components, verification, validation, installation, guidance	Testing results, Working sys
Maintenance	Bug fixes, modifications, adaptation	System versions

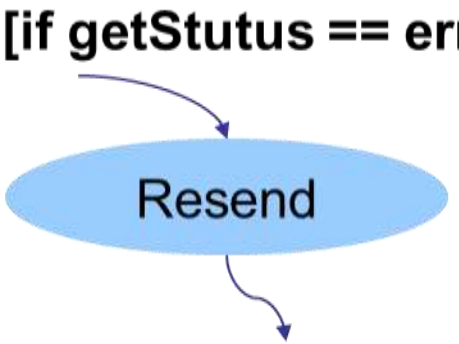
Why to Model Behavior?

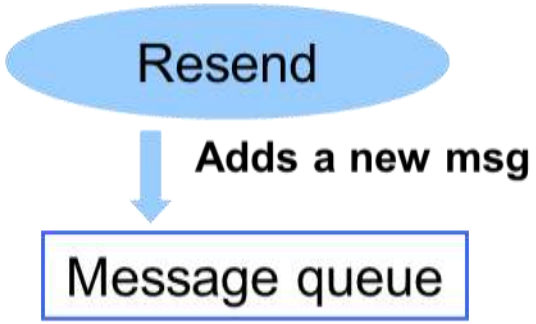


- Bagaimana kita menggunakan interface SMS Server? Bagaimana urutan eksekusi operasinya?
  - sendMessage, getStatus, Resend?
  - getStatus, sendMessage, checkForMessages?
- When do we use resend?

Behavioral Modeling







Urutan Aksi

- Bagaimana operasinya di urutkan
- Alur Eksekusi:
  - Sequential
  - Parallel
  - Loops

Prasyarat

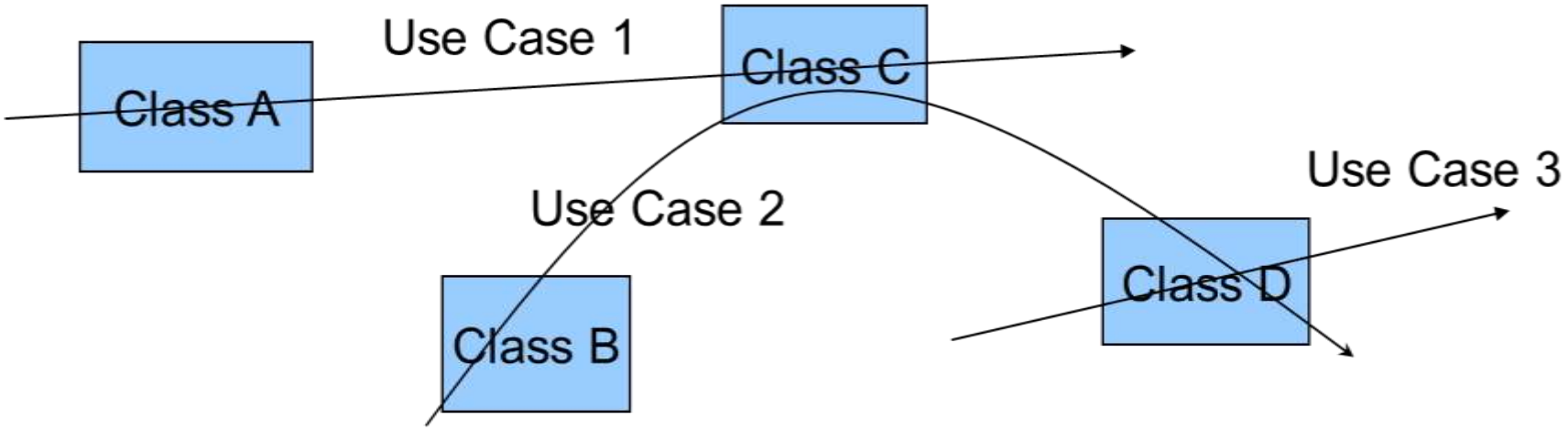
- Kapan Operasi di eksekusi
- Bagaimana hasil dari operasi mempengaruhi eksekusi

Efek

- Apa output dari operasi
- Bagaimana Operasi merubah state dari sistem

Building a Sequence Diagrams

Sequence diagrams capture **use-case** behavior menggunakan dasar **classes**.



Sequence = Objects + messages

Element Sequence Diagram

Sequence diagram terdiri dari:

- **object**

  Direpresentasikan dalam persegi panjang (dengan nama digaris bawahi)

- **Message**

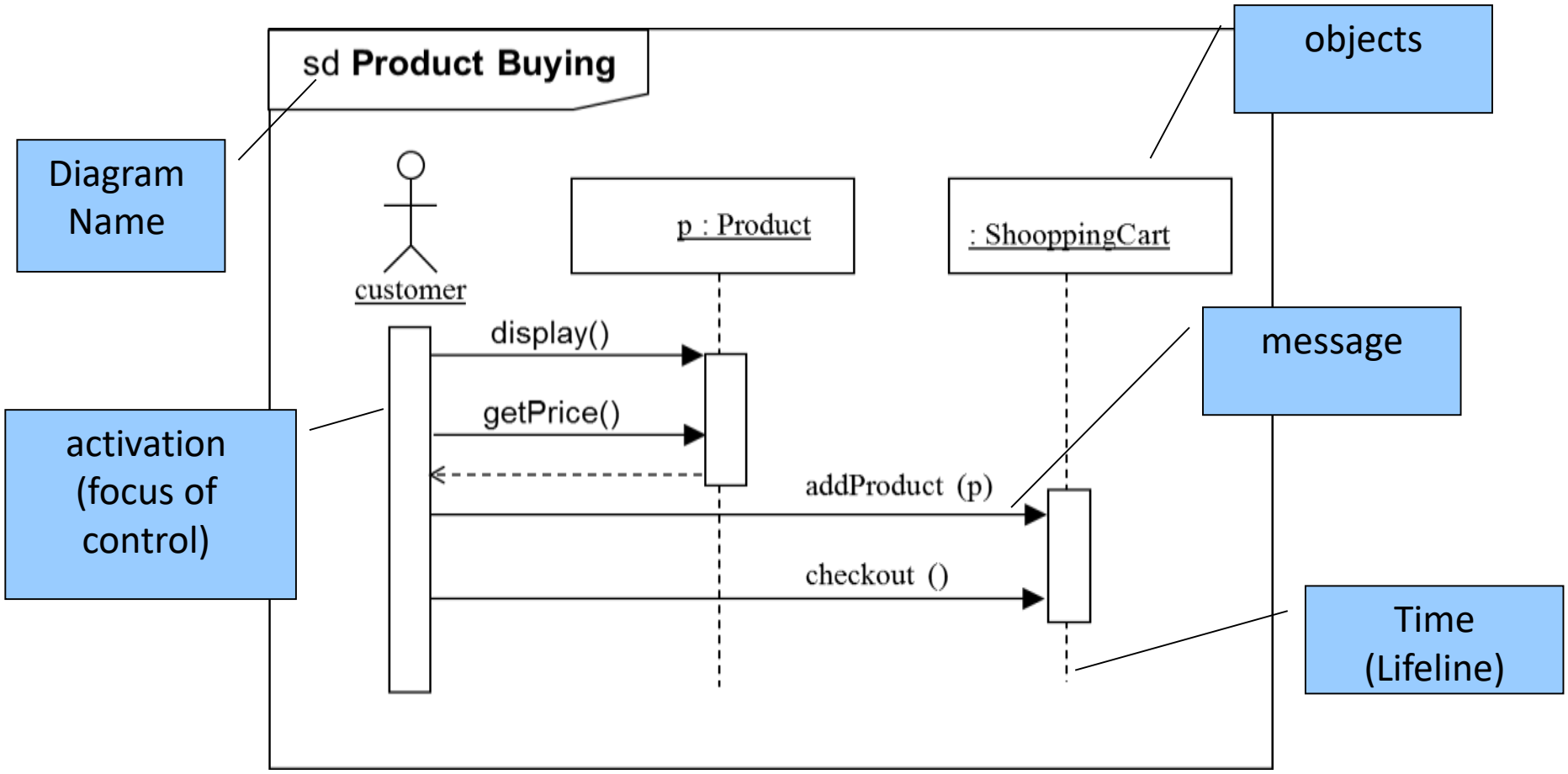
  direpresentasikan sebagai garis panah padat

- **Time**

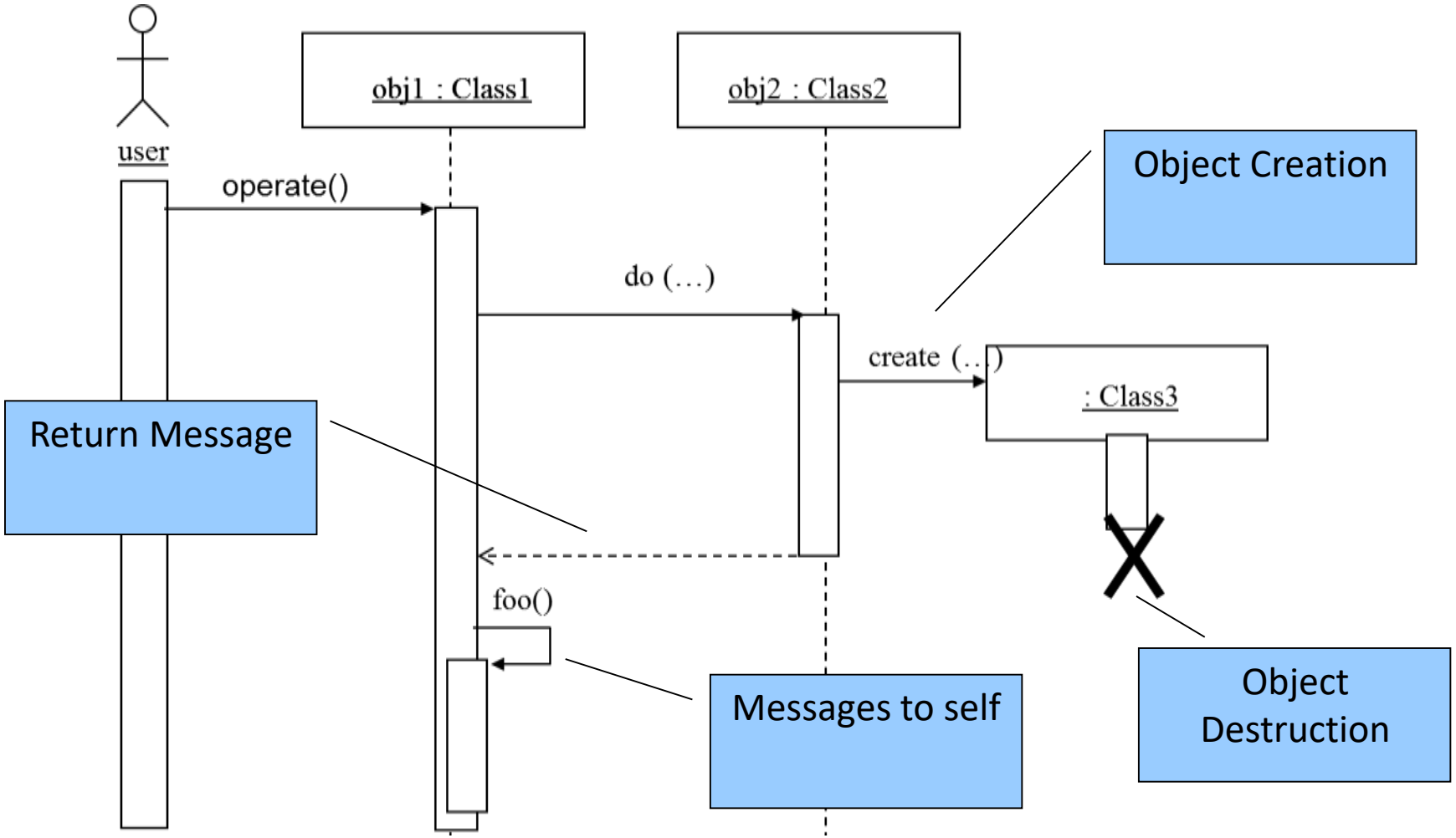
  direpresentasikan sebagai garis progress (vertikal).

Sequence Diagrams

- A simple sequence diagram:



Object Control



Notasi Object

Format Penamaan	Notasi	
Object yang tidak ditentukan class-nya		
Object yang ditentukan class-nya		
Object tanpa nama dari suatu class		

Message

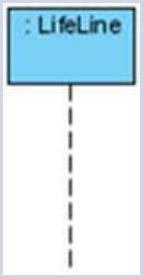
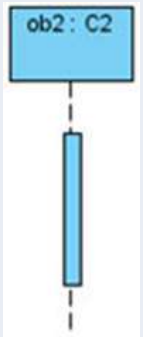
- Message merupakan sarana umum untuk komunikasi antar object.
- suatu object dapat mengirim message ke object lain untuk:
  - Untuk memanggil operasi,
  - Memberi sinyal,
  - Membuat suatu objek atau
  - Menghancurkan objek.

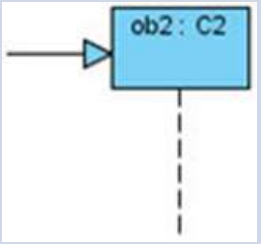
- Dalam sequence diagram, message diwakili oleh garis panah.

Notasi Messages

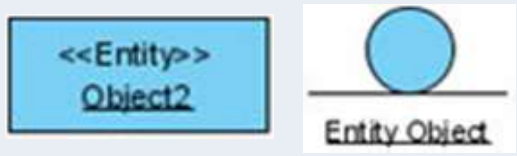
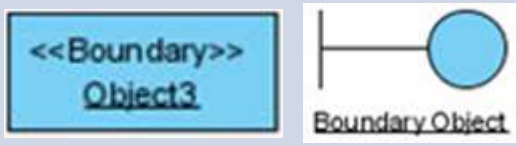
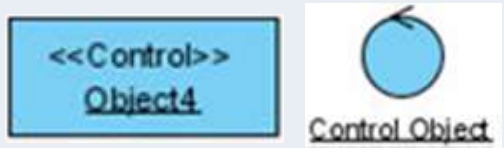
Message	Description	Notasi
Pemanggilan prosedur atau flow dari kontrol	Pengirim message menunggu selesainya panggilan prosedur dari penerima pesan.	
Komunikasi Asynchronous	Pengirim mengirimkan message dan segera dilanjutkan dengan langkah berikutnya pada suatu eksekusi.	
Return message	Message kembali dari panggilan prosedur	
Message dengan travel delay	Message akan akan memerlukan waktu yang cukup untuk tiba pada object penerima. (hanya digunakan pada sequence diagrams).	

Notasi - Lanjutan

Bentuk2 lain	Deskripsi	Notasi
Lifeline (Time)		
Execution Occurrence		

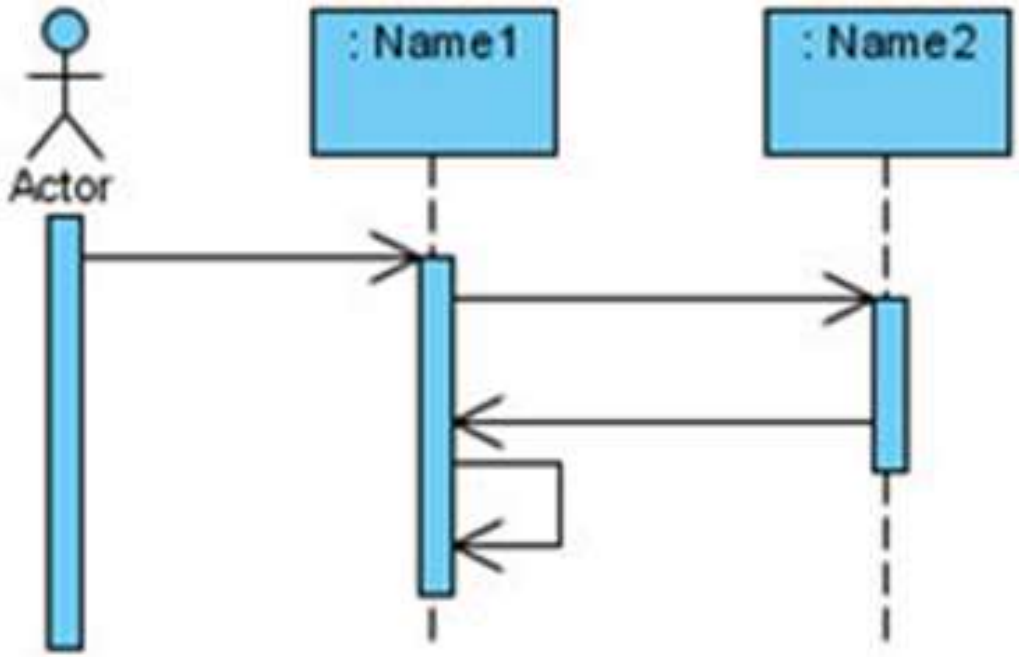
Bentuk2 lain	Deskripsi	Notasi
Creation		
Destruction	destruction akan ditandai dengan simbol silang di bawah garis lifline	

Object Stereotype

Object Category	Description	Graphical Notations
Actor Object	Suatu entitas eksternal (external entity) yang berinteraksi dengan sistem.	
Entity Object	Suatu object yg merupakan model data pada sistem. (Object ini Sering direpresentasikan pada problem domain)	
Boundary Object	Suatu Object yang menangani komunikasi antara objectr actor dan system.	
Control Object	Suatu object sebagai **model flow suatu kendali** dan fungsionalitas yang bukan milik suatu object atau object batas	

Object Stereotype - lanjutan

- Commonly Used Stereotypes for Objects



Skenario Representasi Fokus waktu

- Skenario pemodelan adalah tentang menggambarkan bagaimana objek dalam sistem berinteraksi satu sama lain dalam suatu skenario.
- Skenario adalah urutan peristiwa yang terjadi selama satu jalur eksekusi tertentu dalam ***use-case*** sistem.
- Setiap peristiwa melibatkan interaksi object melalui pesan antar object.
- Sequence diagram mewakili waktudalam arah vertikal. Waktu dimulai dari bagian atas dan berkembang ke arah bawah.

  Sebuah pesan yang lebih atas merupakan pesan yg terjadi lebih dahulu dibanding pesan yang dibawahnya.

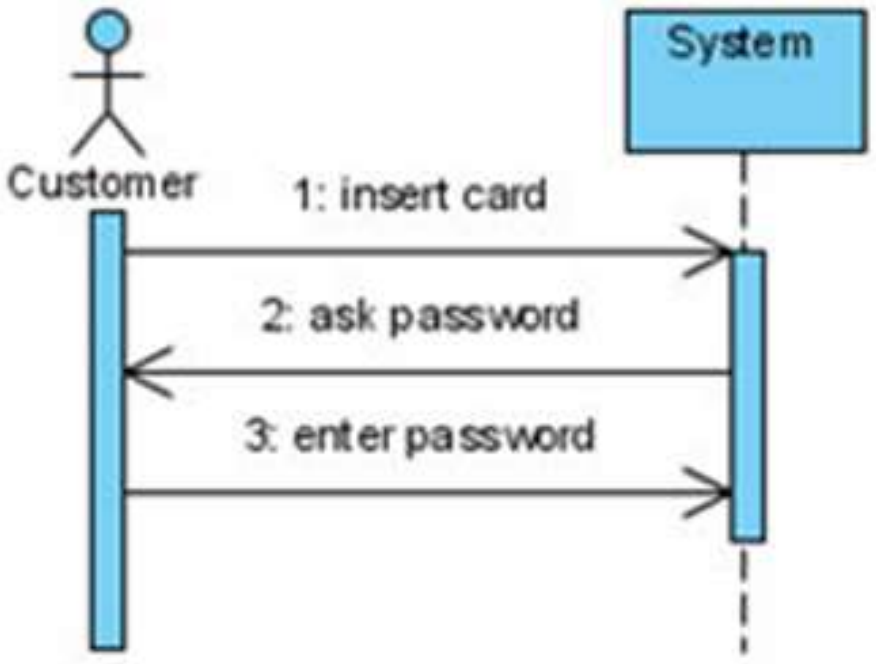
Penggambaran Sequence Diagram

- Suatu sequence diagram memiliki 2 dimensi:
  - Dimensi vertikal (tahapan waktu)
  - Dimensi horisontal (tahapan interaksi)
- Masing-masing mewakili berlalunya waktu dan objek yang terlibat dalam interaksi.
- Ikon Obyek ditempatkan secara horizontal di bagian atas sequence diagram, dan message melintas / lewat di antara object-object tersebut

  (*lihat **contoh 1**, pada slide selanjutnya*)

Contoh 1:

- Sequence diagram untuk prosedur login dari suatu sistem Automatic Teller Machine (ATM).



Merubah message menjadi fungsi prototype

- Sequence diagram kemudian disempurnakan secara iteratif sampai akhirnya semua message diubah menjadi fungsi protoype,

  Seperti;

  placeOrder (tanggal, perusahaan, contactPerson)

- Perubahan message menjadi fungsi protoype menyediakan banyak informasi yang lebih berguna untuk implementasi.

(*lihat **contoh 2**, pada slide selanjutnya*)

Contoh 2: Message → Fungsi Prototipe

- Misal; ada object customer yang melakukan order atas nama perusahaannya. Customer melakukan operasi placeOrder dengan memberikan informasi yg diperlukan (yaitu: *date, company, contactPerson*).

Penggambaran lifeline (time)

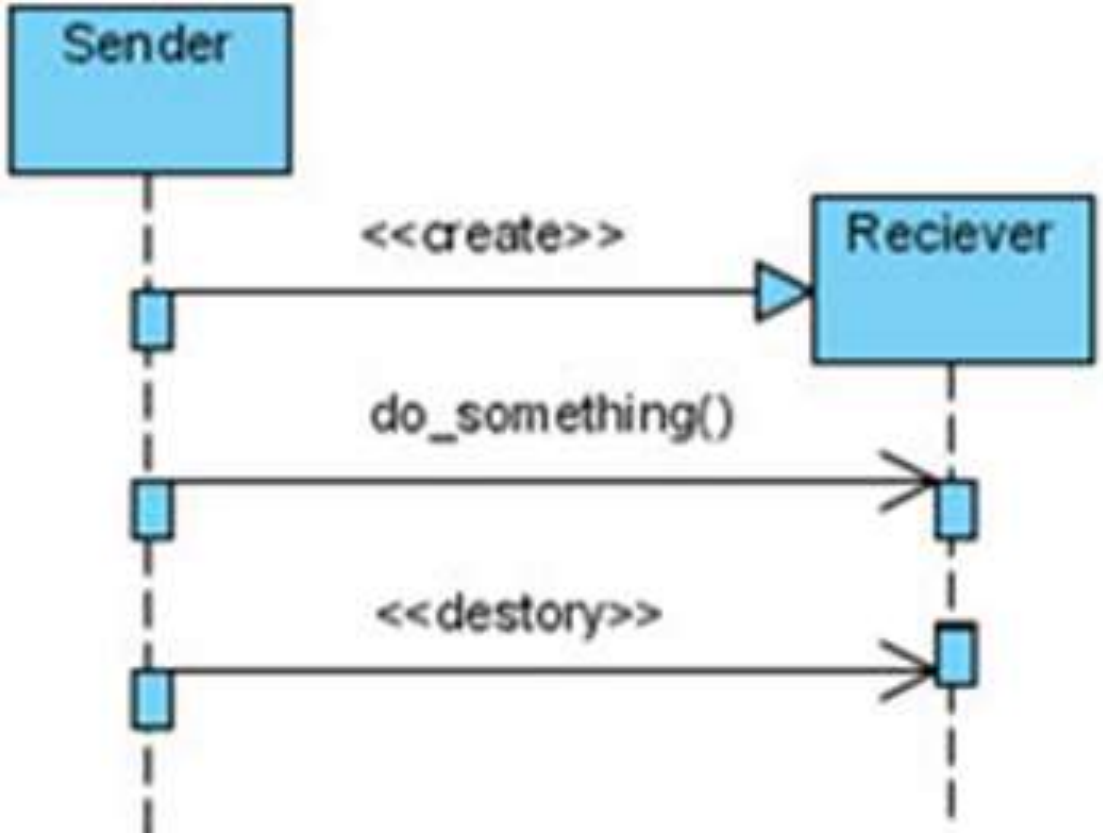
- *Lifeline* adalah garis putus-putus vertikal yang menunjukkan keberadaan object dari waktu ke waktu.
- Jika *lifeline* meluas ke bagian bawah diagram, object akan terus ada selama seluruh sesi interaksi.
- Jika object diposisikan di bagian atas diagram, ini menunjukkan bahwa object tersebut benar-benar ada sebelum interaksi.

Object Creation and Deletion

- Dengan mengirimkan message **<<create>>**, suatu object dapat membuat object baru secara dinamis.
- Dengan mengirimkan message **<<destroy>>**, suatu object dapat menghapus suatu object lain secara dinamis.
- Sebuah cross (X) ditempatkan di akhir *lifeline* suatu object, untuk menunjukkan bahwa keberadaan object telah berakhir pada saat itu.

(*lihat **contoh 3**, pada slide selanjutnya*)

Contoh 3: Object Creation and Deletion

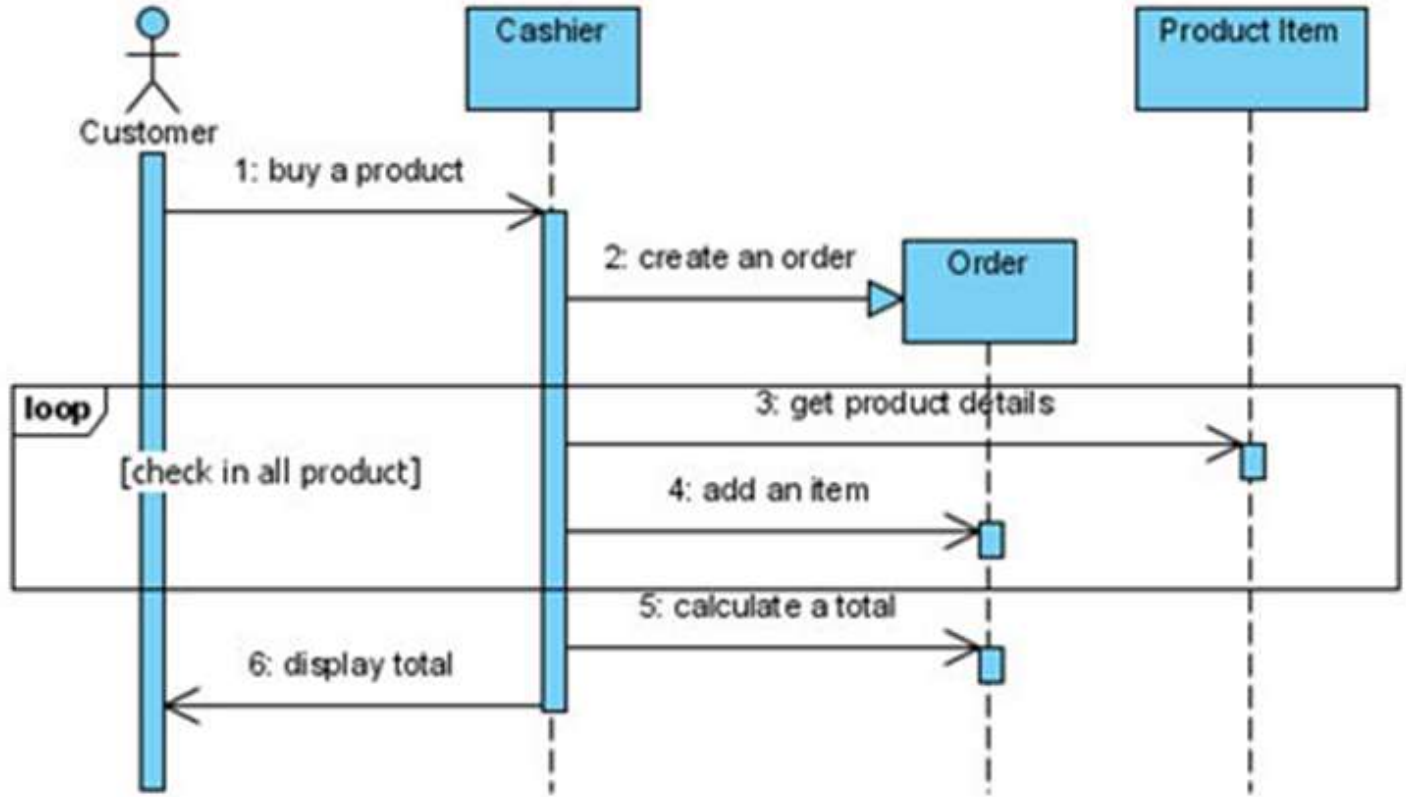


Object Activation

- *Lifeline* suatu object memberikan gambaran tentang durasi di mana object itu ada, tetapi tidak memberikan informasi kapan suatu object aktif atau tidak aktif melakukan suatu tugas.
- Sebuah persegi panjang kurus digunakan untuk mewakili waktu ketika suatu objek aktif (*lihat **contoh 4***)
- Bagian atas persegi panjang menandakan saat aktivasi dimulai dan bagian bawah persegi panjang waktu selesai

Contoh 4: Object Activation

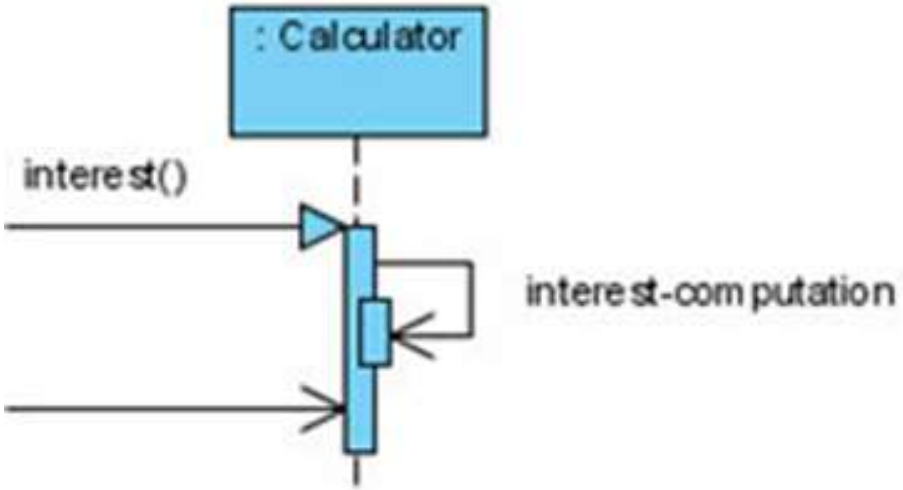
- Object penerima aktif saat menerima message dari object pengirim



Recursive Message

- Terkadang sebuah objek memiliki operasi yang memanggil dirinya sendiri (ini disebut recursive)
- Misalkan salah satu objek dalam sistem kalkulator, untuk menghitung bunga majemuk mencakup periode waktu tertentu, operasi object perhitungan bunga harus memanggil dirinya beberapa kali.

  Diagram urutan untuk ini adalah sebagai berikut:

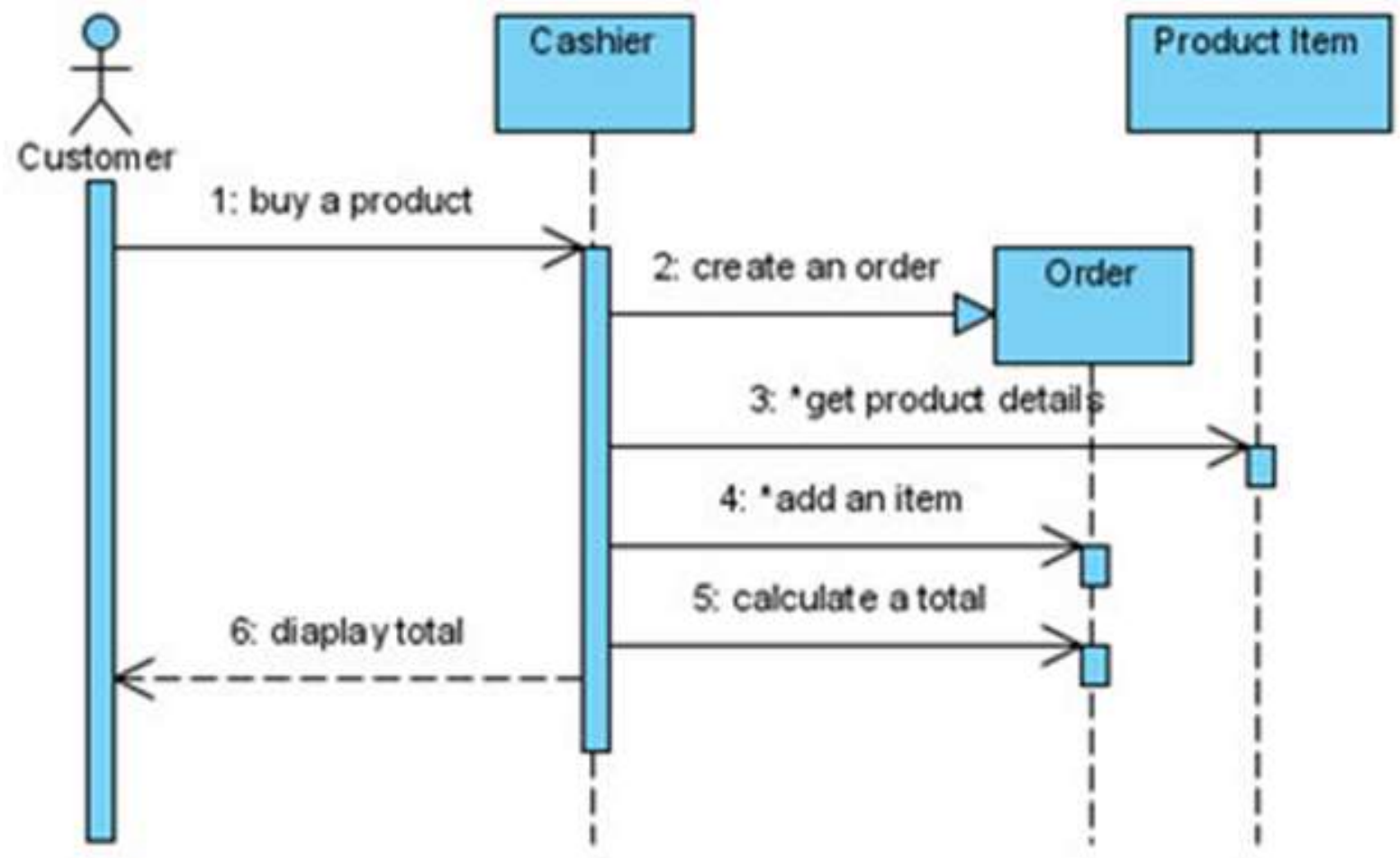


Simple and Collective Iteration

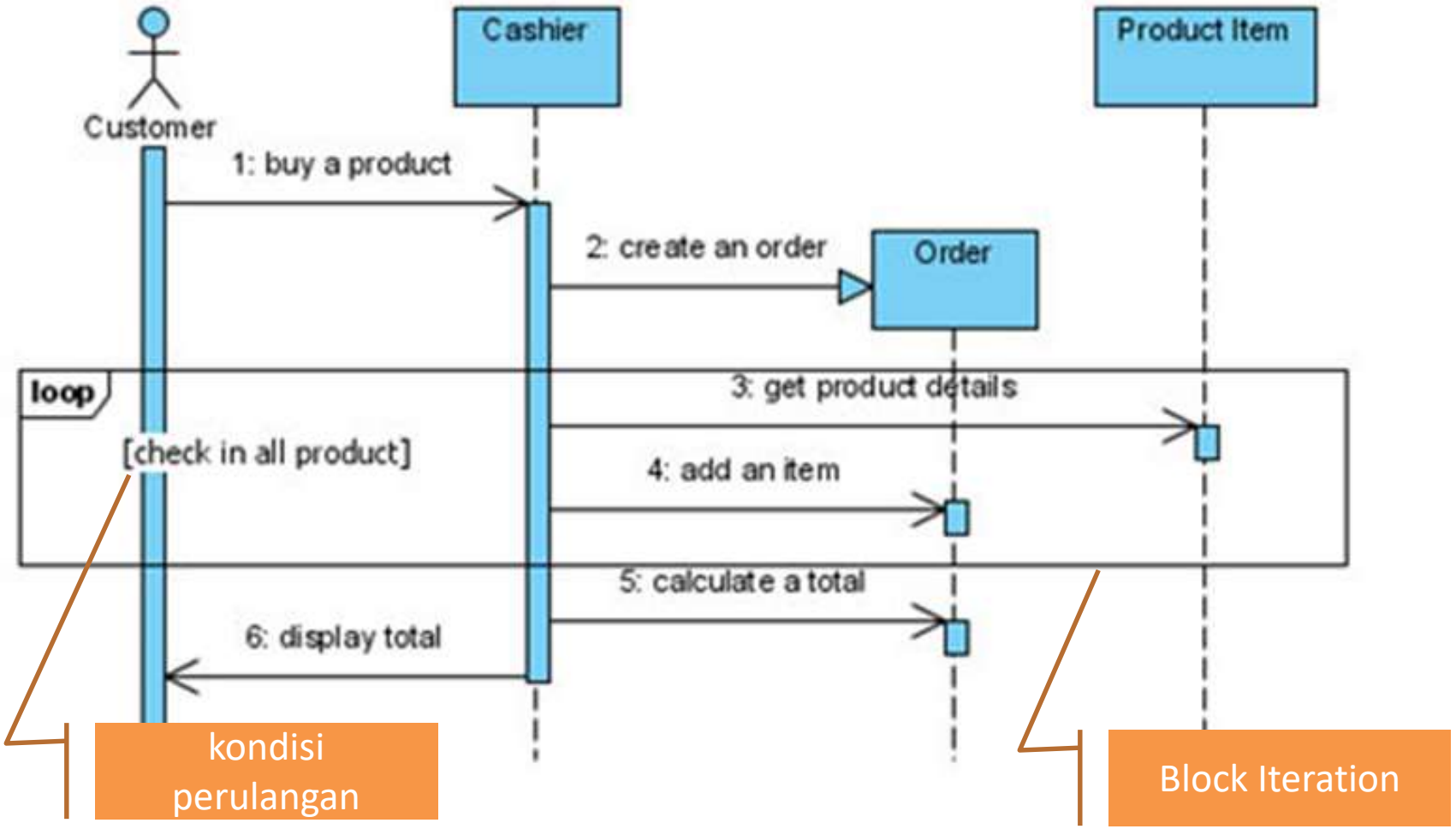
Terkadang suatu tugas dilakukan secara berulang beberapa kali

- Tanda * diberikan sebagai prefix nama message / tugas (task) yang dilakukan secara berulang beberapa kali
  (***lihat contoh 5***)
- Untuk menggambarkan sekumpulan messages yang dilakukan berulang-ulang digunakan kotak yang membungkus message yang berulang tersebut.
  Dan kondisi perulangannya ditulis di dalam simbol **[ ]**
- Jika digunakan kondisi perulangan dengan simbol **[ ]** maka tidak diperlukan lagi prefix * yang menyatakan perulangan
  (***lihat contoh 6***)

Contoh 5: Simple Iteration



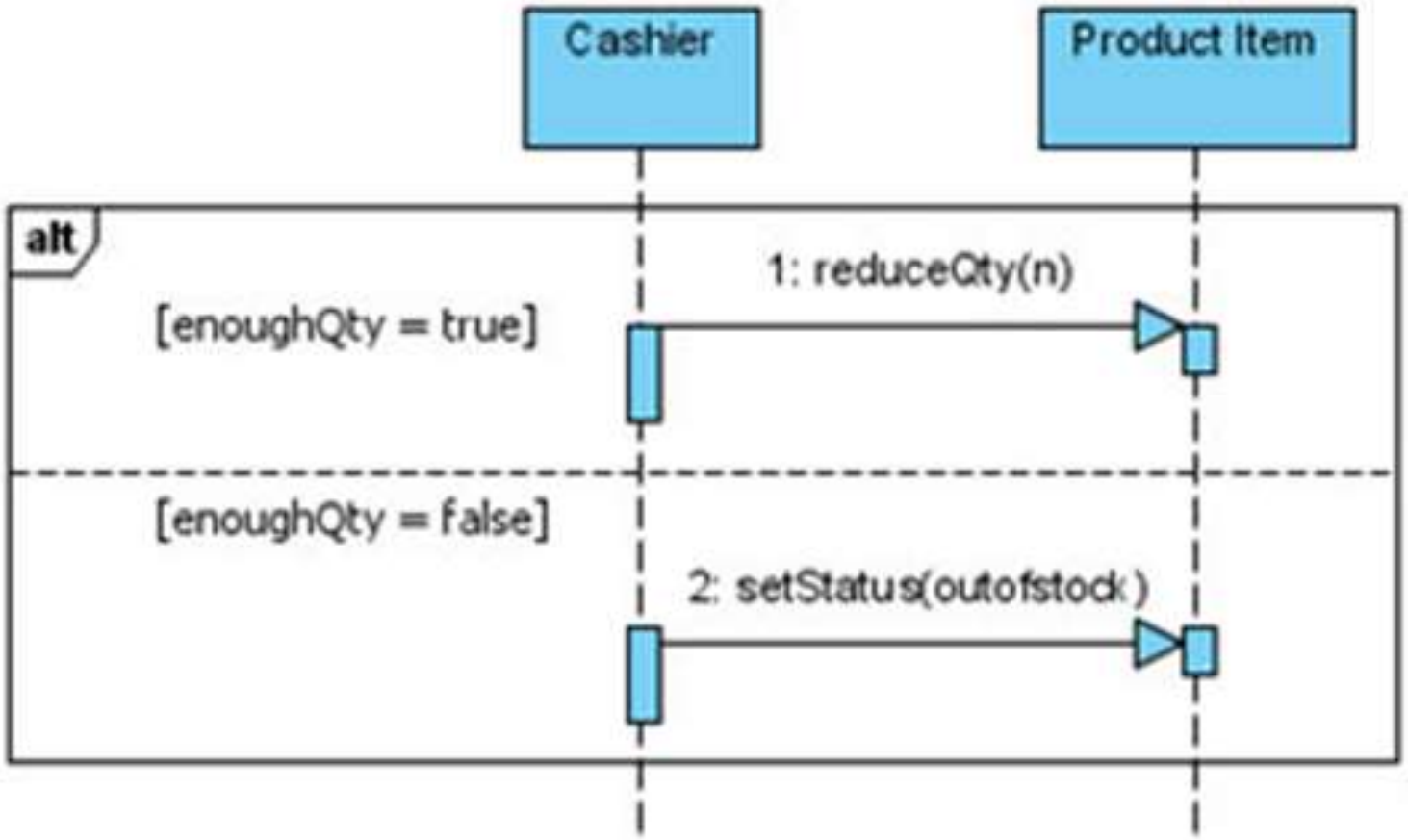
Contoh 6: Block Iteration



Branching

- Branch (percabangan) digunakan untuk merepresentasikan aksi percabangan kondisional atau bersamaan, dan ini ditunjukkan oleh beberapa panah meninggalkan titik yang sama dari *lifeline* object.
- Setiap message dapat diberi label dengan suatu kondisi. Jika kondisi dari pesan yang saling eksklusif, cabang adalah bersyarat, jika tidak maka konkurensi

Contoh 6: Conditional Case in Sequence Diagram

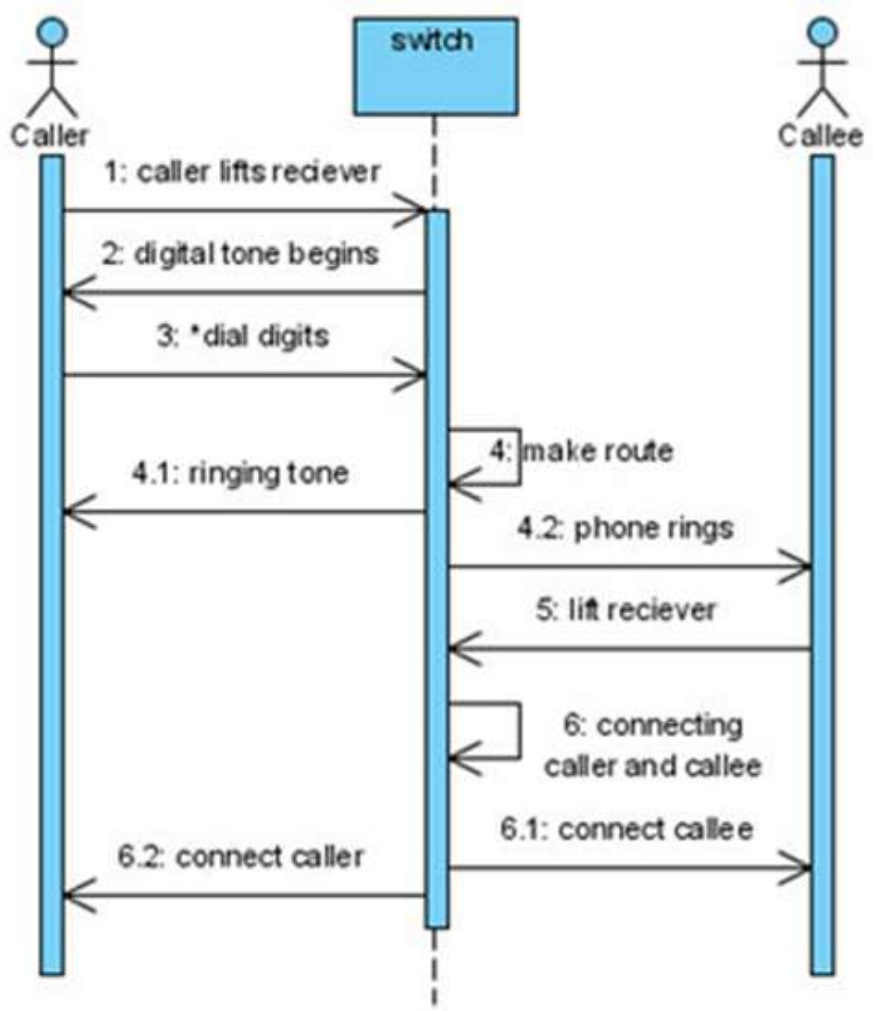


Contoh Branching in Connecting Caller and Callee

- caller lifts receiver
- dial tone begins
- caller dials digits one at a time
- switch makes routing
- ringing tone on callee's receiver begins
- phone rings on callee's receiver begins
- callee lifts receiver
- switch makes connection between caller and callee
- switch connects callee
- switch connects caller

(*lihat Contoh 7*)

Contoh 7: Branching in Connecting Caller and Callee



Thank's

- See Ya Next Week