



Sequence Diagram

Sumber: Gunadarma.ac.id,
IlmuKomputer.com

Team Teaching PPLBO

Outline

- Sequence Diagram
- Tujuan
- Komponen
- Sequence Diagram dengan Boundary-Control-Entity
- Sequence Fragment
- Contoh
- Best Practice
- Studi Kasus


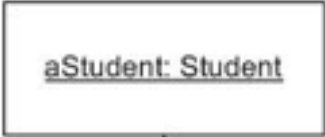

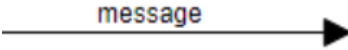
Sequences Diagram

- Interaction Diagram
 - Menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display/form) berupa message yang digambarkan terhadap waktu.
- Terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).
- Berasosiasi dengan use case diagram dan memperlihatkan tahapan detail dalam suatu use case sehingga Sequence Diagram (sebaiknya) sejumlah Use Case yang ada.

Tujuan

- Digunakan untuk memperlihatkan interaksi antar obyek dalam perintah yang berurut.
- Mendefinisikan urutan kejadian yang dapat menghasilkan output yang diinginkan
- Mirip dengan activity diagram
 - Menggambarkan alur kejadian sebuah aktivitas
 - Lebih detail dalam menggambarkan aliran data, termasuk data atau behaviour yang dikirimkan atau diterima
 - Kurang mampu menjelaskan detail dari sebuah algoritma (loop, branching)
 - Lebih teknis, umumnya dipakai sebagai alat komunikasi antara System Analyst dan Developer

Komponen




Simbol	Keterangan
	Aktor yang sedang berinteraksi dengan sistem
	Objek yang terlibat dalam use case
	Lifeline (garis putus-putus) merepresentasikan daur hidup setiap objek sedangkan activation control/focus of control (persegi panjang) mengindikasikan durasi waktu dimana objek sedang aktif (ada interaksi dengan objek lain)
	Menggambarkan pengiriman pesan antar objek



Sequence Diagram dengan Boundary-Control-Entity

Sequence Diagram ini memakai metode programming MVC (Model-View-Controller) atau dalam istilah lain Model=Entity, View=Boundary, dan Controller=Control.

Sequence Diagram dengan Boundary-Control-Entity

Simbol	Keterangan
	Boundary Class: class yang berinteraksi langsung dengan actor atau sistem lain, misalnya form input, user interface, printer, kotak uang di mesin ATM, dsb
	Control Class: class yang berhubungan dengan pemrosesan, perhitungan, kalkulasi, komputasi, query, dsb
	Entity Class: class yang berhubungan dengan data/penyimpanan data

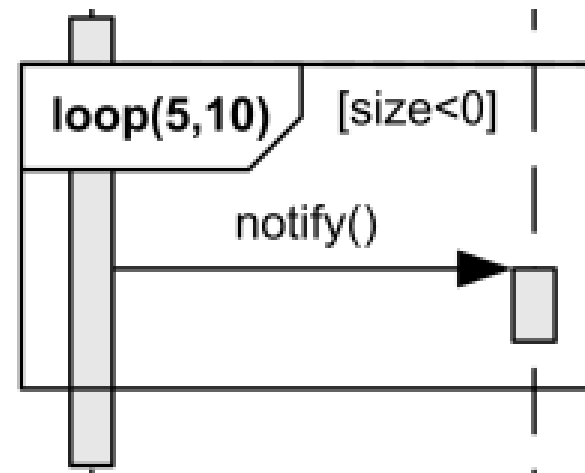
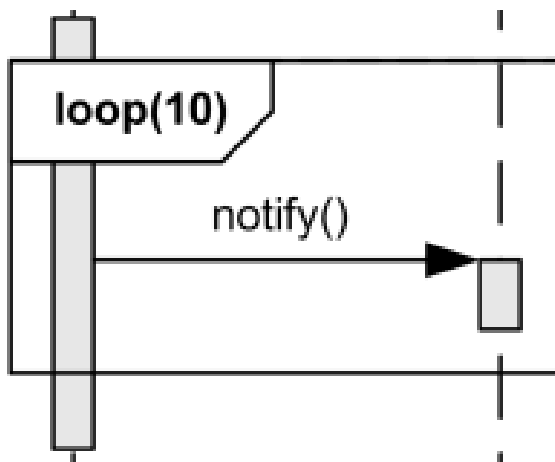
Sequence Fragment

- Digunakan untuk mempresentasikan interaksi yang kompleks, misalnya alternative flows dan loop dengan cara yang lebih terstruktur
- Ditampilkan dalam sebuah kotak (frame) yang melingkupi sebagian interaksi pada sequence diagram
- Fragment operator (di pojok kiri atas) menunjukkan tipe fragment

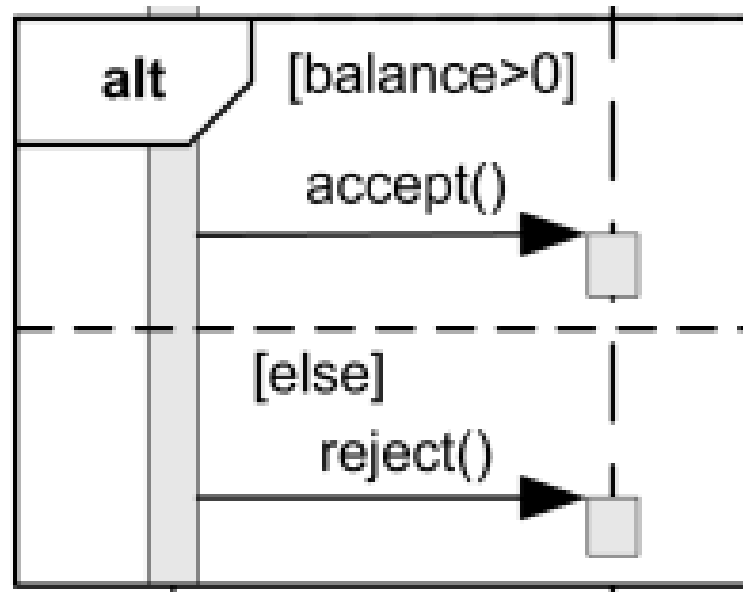
Tipe Sequence Fragment

Operator	Keterangan
alt	Alternative: alternative dari multiple fragment, hanya kondisi benar yang akan dieksekusi
opt	Optional: fragment akan dieksekusi hanya jika kondisi dipenuhi
par	Parallel: setiap fragments dieksekusi secara paralel
loop	Loop: fragment bisa dieksekusi berulang kali
critical	Critical: fragment yang melingkupi interaksi yang sangat penting
neg	Negative: fragment yang menunjukkan invalid interaction yang muncul ketika system failed
ref	Reference: menunjuk pada suatu interaksi yang direpresentasikan oleh diagram lain
sd	Sequence diagram: melingkupi sequence diagram secara keseluruhan

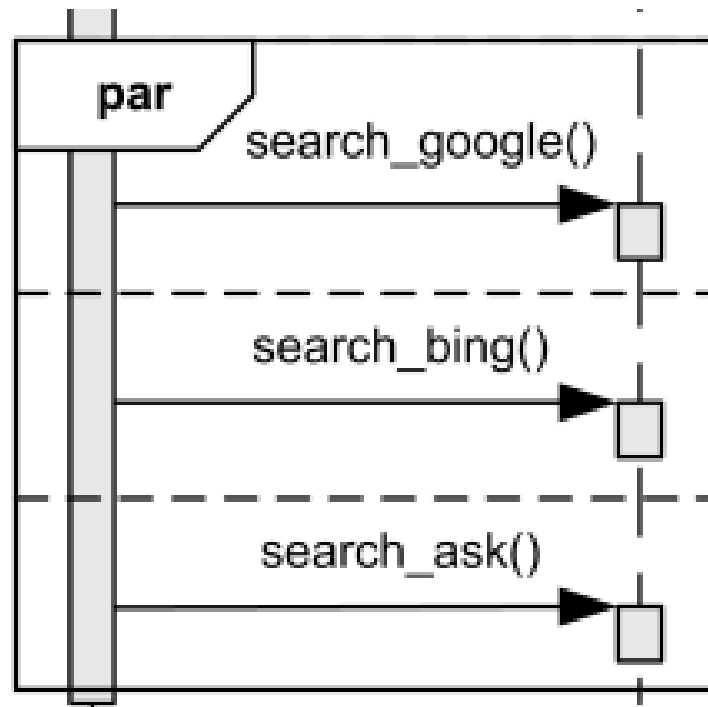
Contoh Loop Fragment



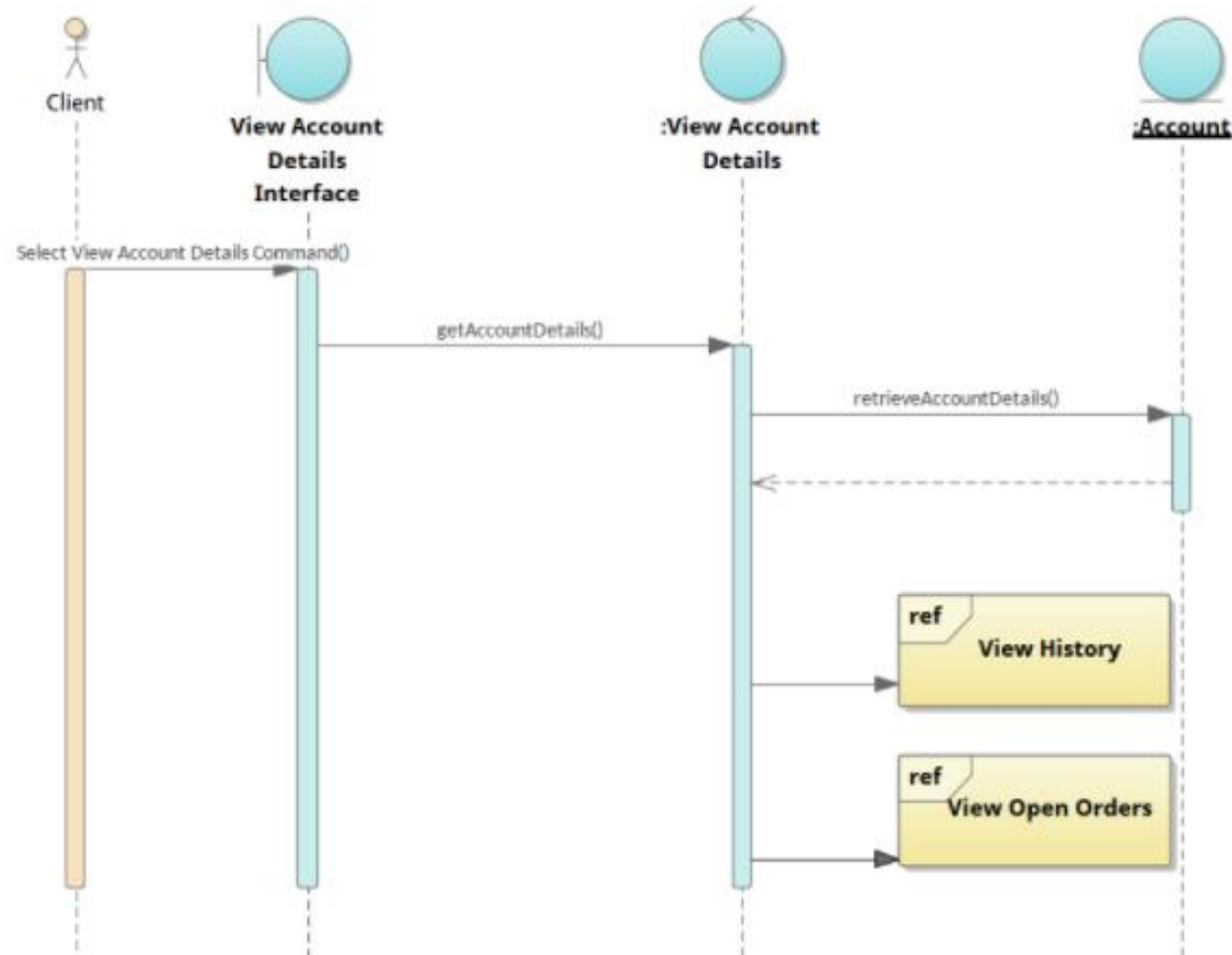
Contoh Alternative Fragment



Contoh Parallel Fragment

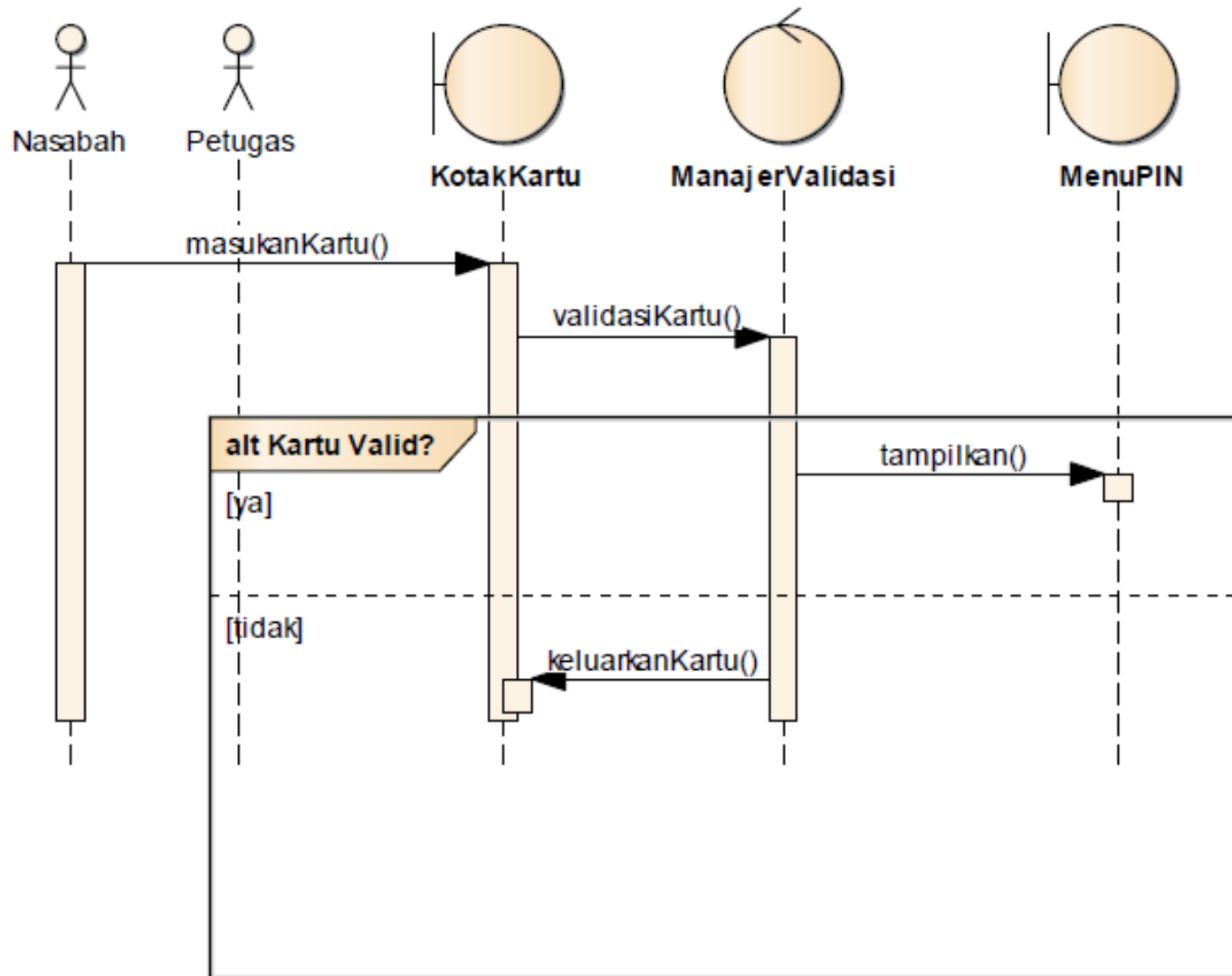


Contoh Reference Fragment

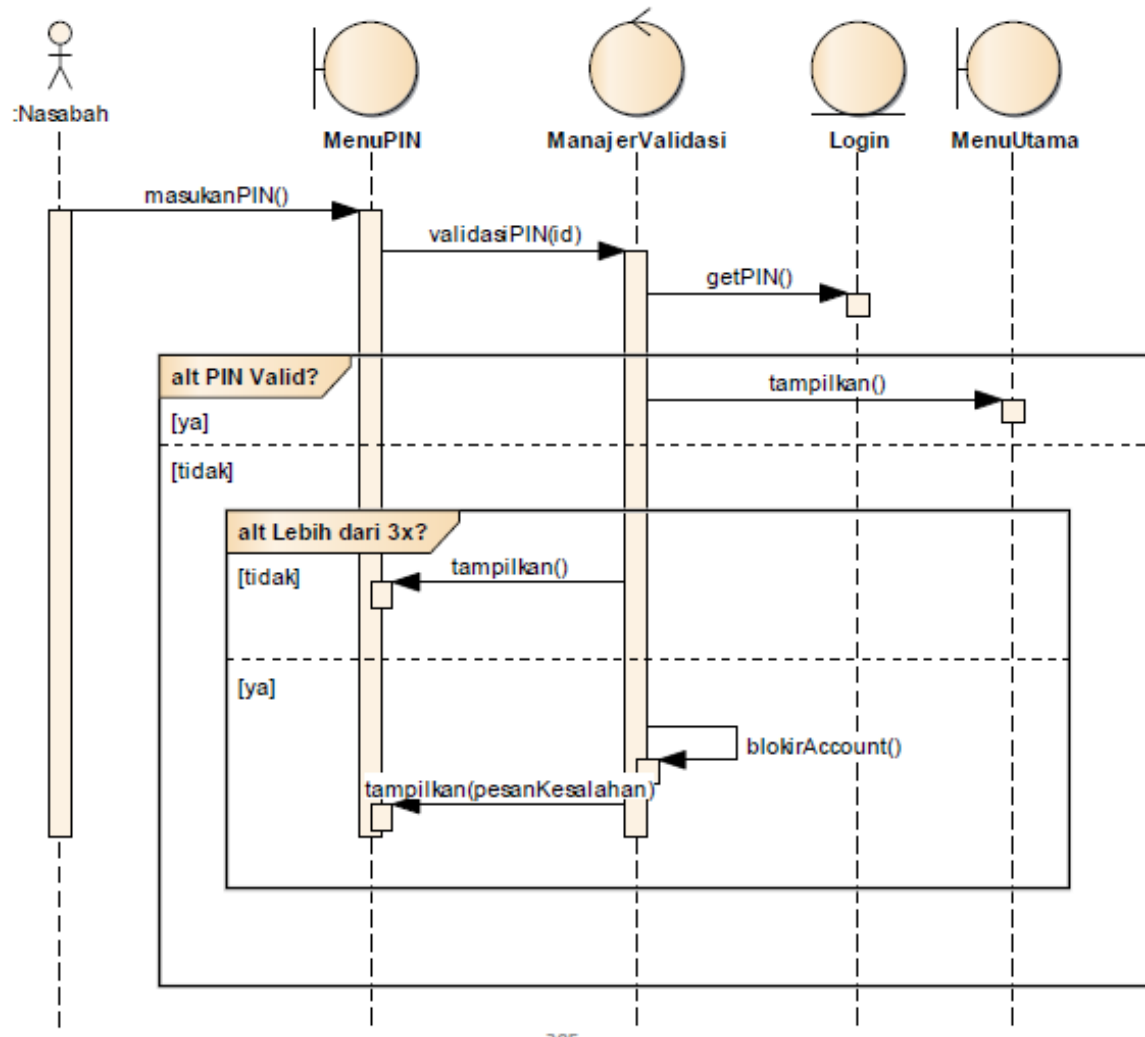


Contoh Sequence Diagram pada Sistem ATM

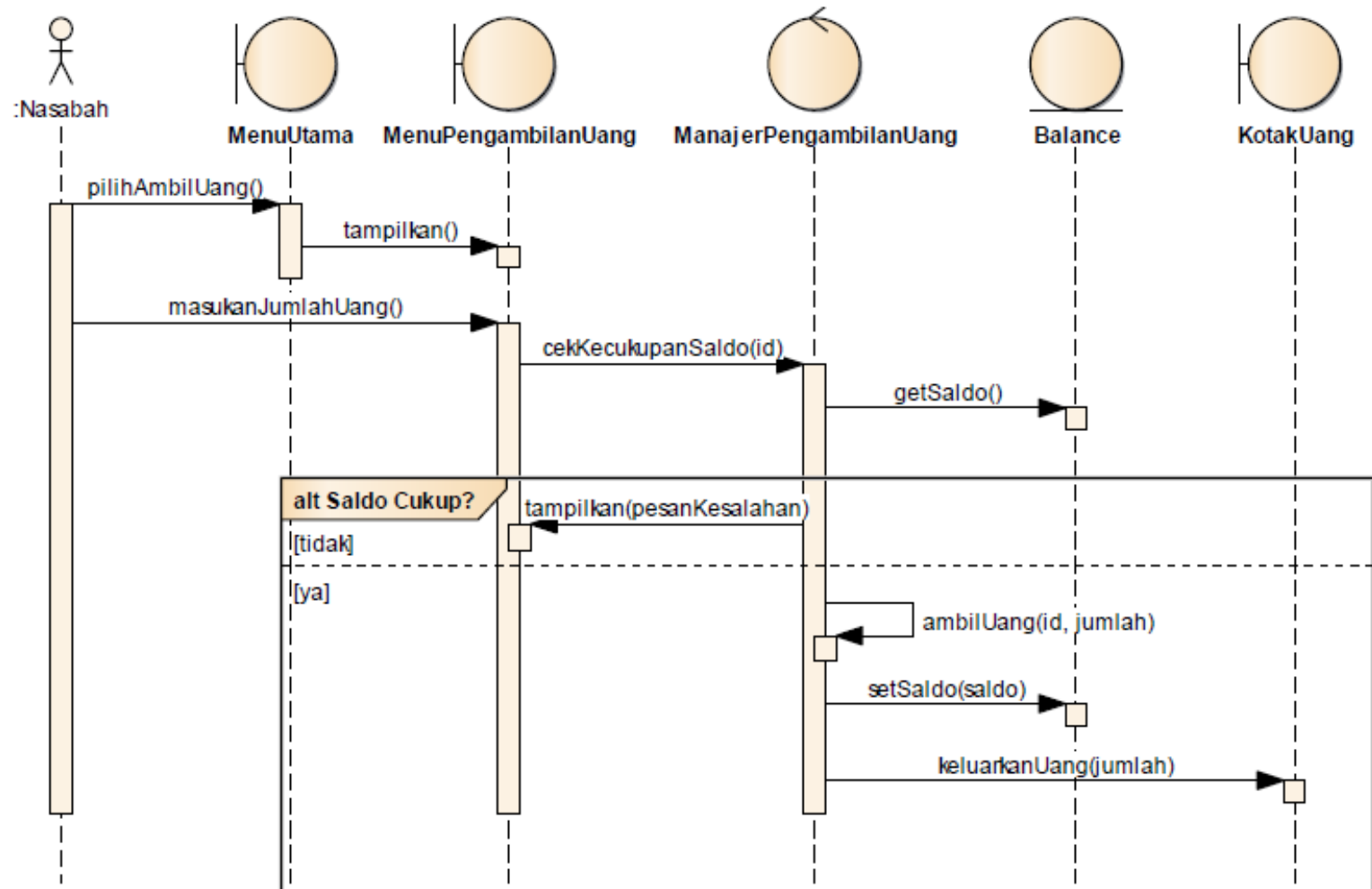
Sequence Diagram Memasukkan Kartu



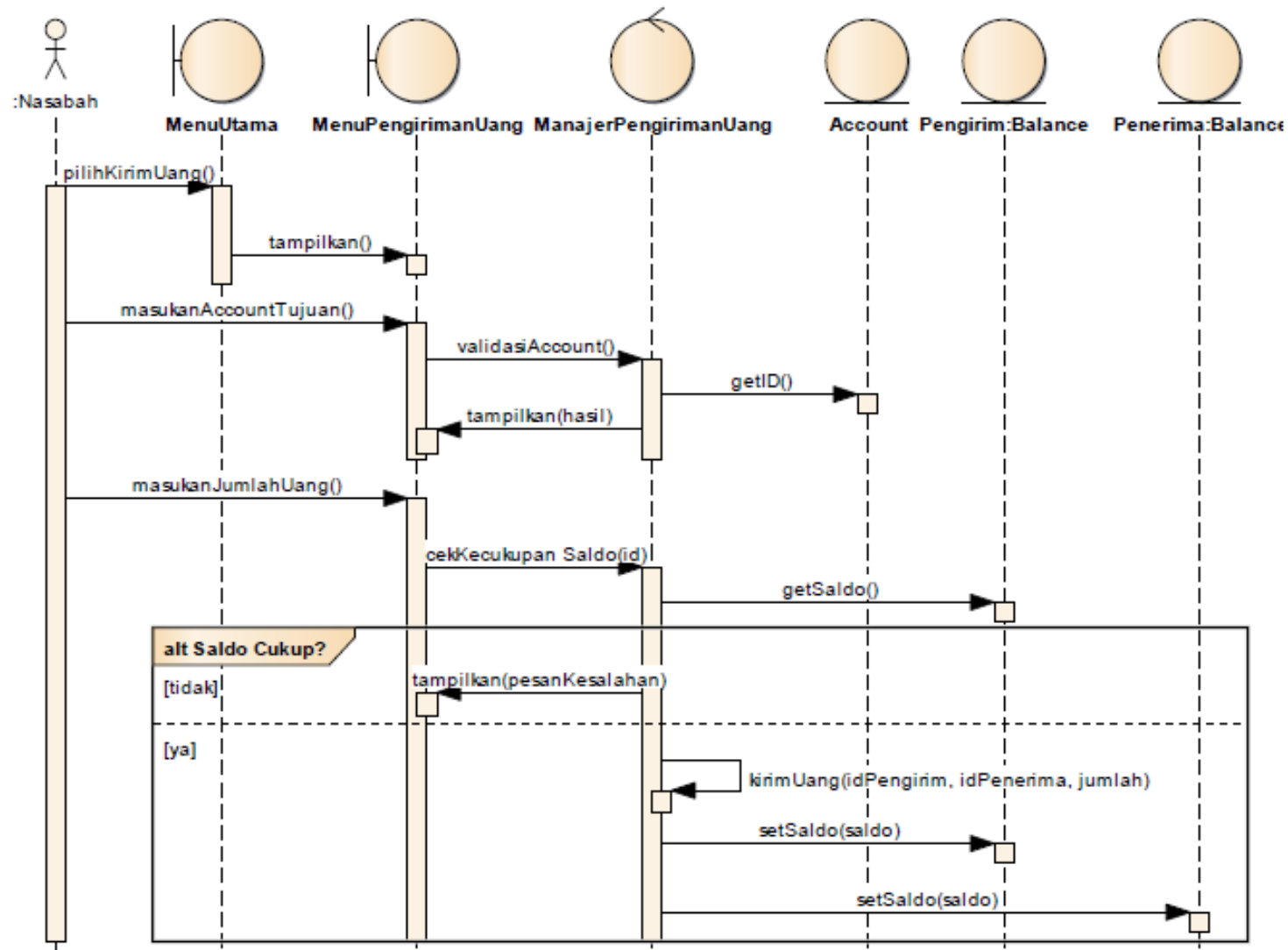
Sequence Diagram Memasukkan PIN



Sequence Diagram Mengambil Uang



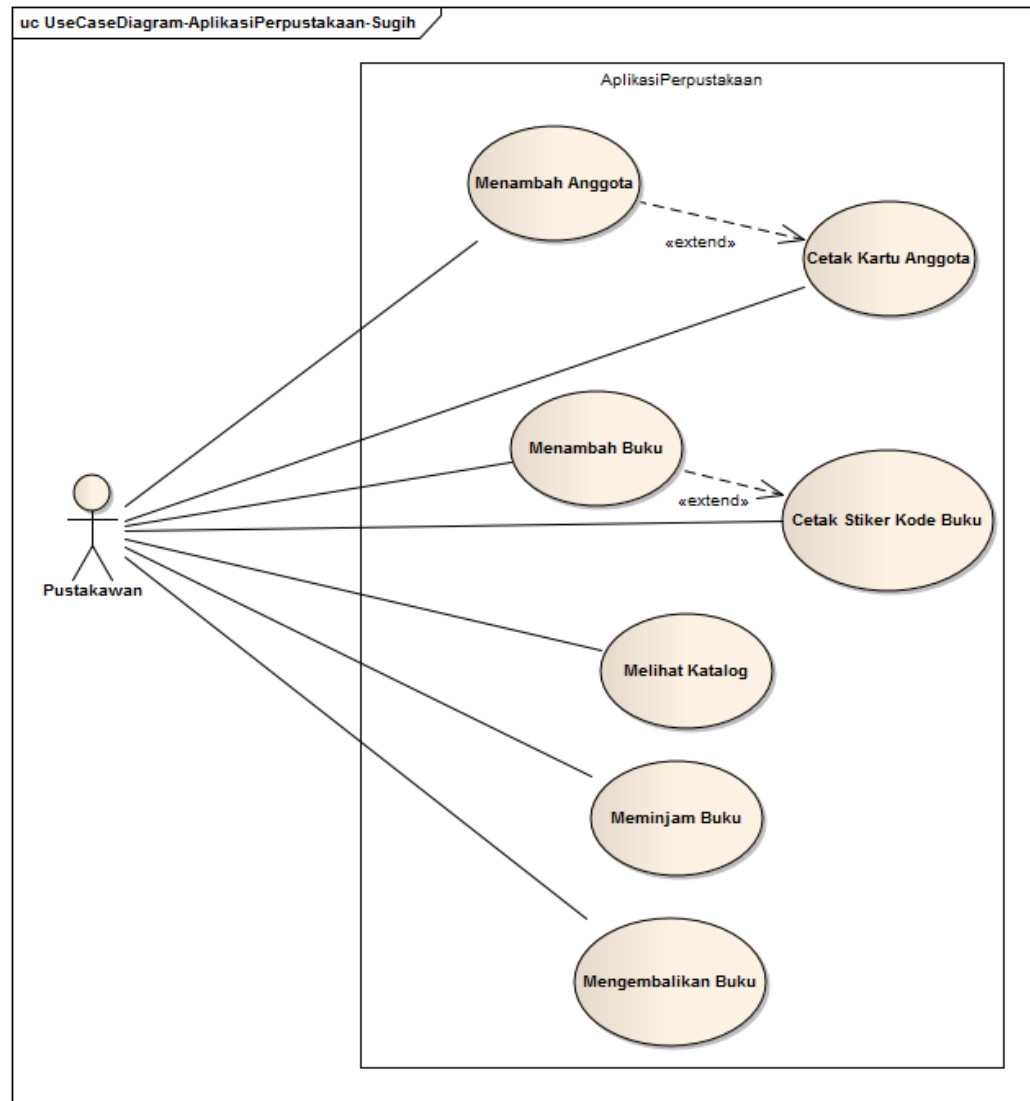
Sequence Diagram Mengirim Uang



Best Practice

- Penamaan object dan class pada sequence diagram menggunakan kata benda (noun) sedangkan message (method) menggunakan kata kerja
- Nama object, class, dan message sebaiknya tidak menggunakan spasi karena akan menjadi nama class dan method di kode program
- Actor (manusia) hanya akan mengirim message ke Boundary Class, tidak ke Control atau Entity Class. Sementara Actor (System) akan mengirim message ke Control
- Boundary class akan menjadi user interface sedangkan entity class akan menjadi Data Model

Studi Kasus Perpustakaan





Gambar di atas menggambarkan Aplikasi mempunyai satu Actor/user yaitu Pustakawan dan 7 use case. Hal ini menjelaskan bahwa dalam aplikasi, pustakawan bisa Menambah Anggota, Mencetak Kartu Anggota, Menambah Buku, Mencetak Stiker Kode Buku, Melihat Katalog, Meminjam Buku, dan Mengembalikan Buku.

Mungkin ada kebingungan, mengapa yang meminjam dan mengembalikan buku adalah Pustakawan, bukan anggota perpustakaan.

Kalau kita lihat [Business Process atau Activity Diagram](#) , terlihat bahwa yang berinteraksi langsung dengan aplikasi adalah Pustakawan, bukan anggota. Anggota meminjam dan mengembalikan buku kepada Pustakawan, selanjutnya Pustakawan lah yang menginput ke aplikasi.

Activity Diagram

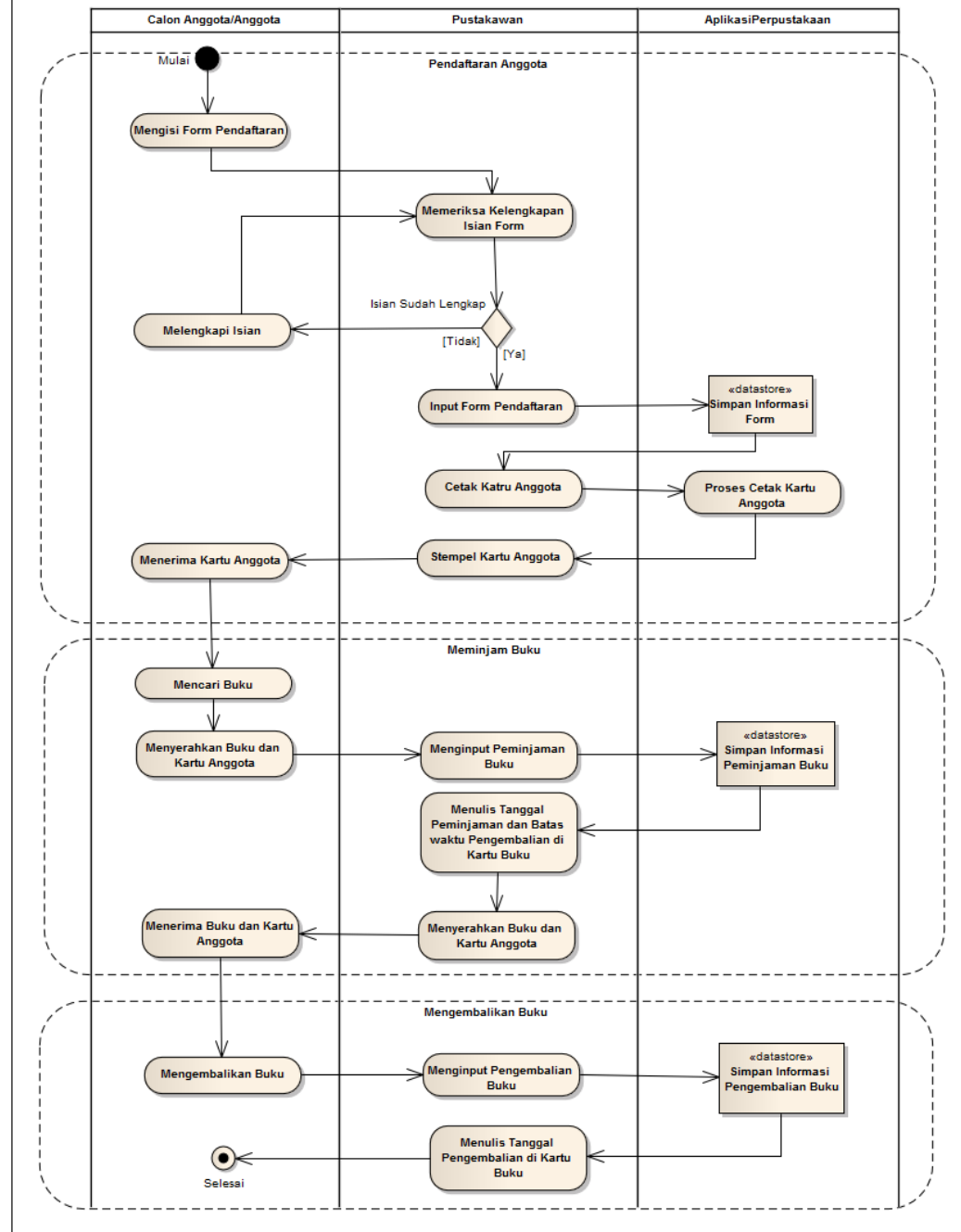




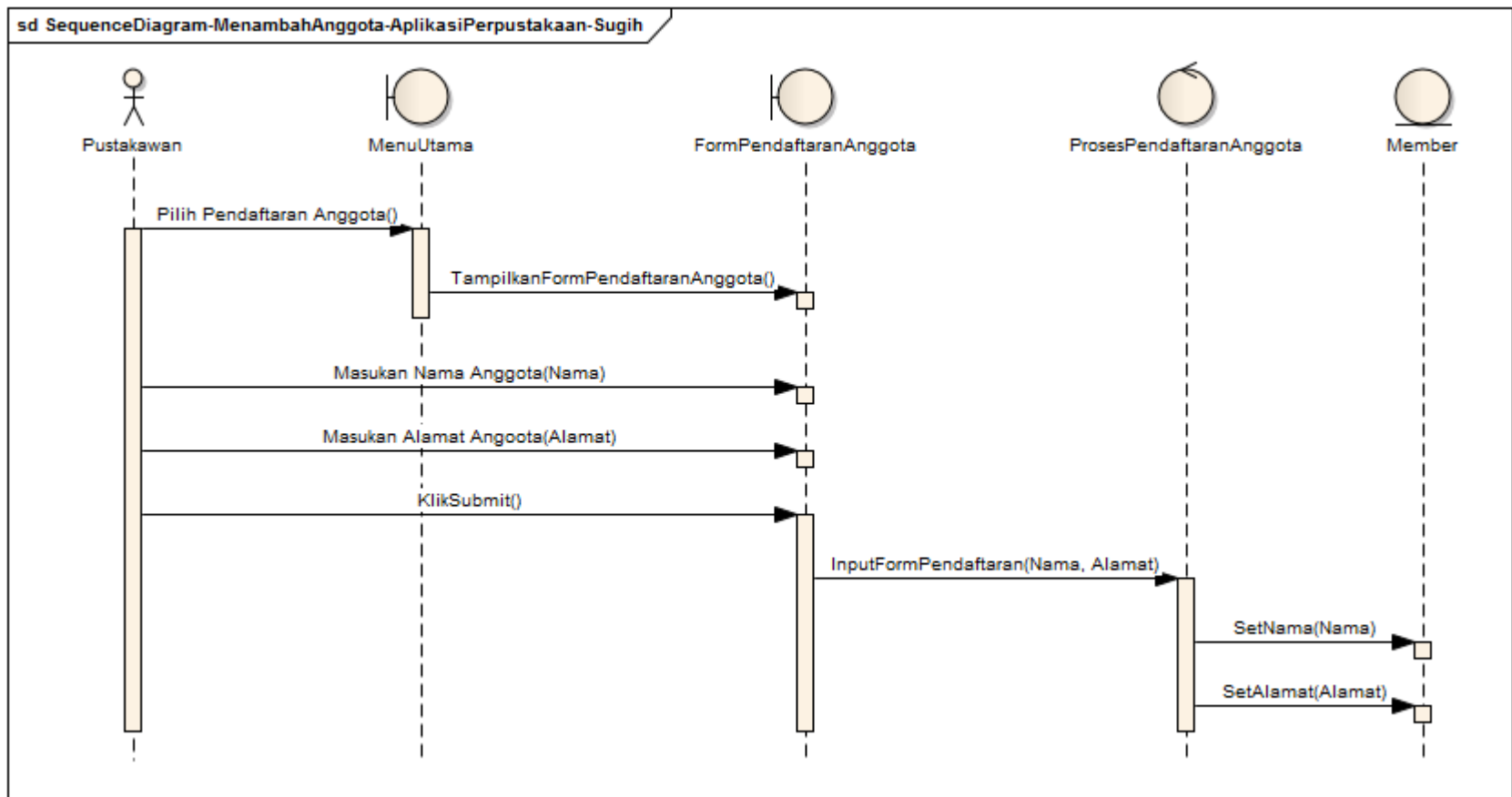
Diagram di atas menggambarkan 3 Activity utama di dalam perpustakaan,yaitu:

- Menambah anggota/member perpustakaan.
- Anggota meminjam buku.
- Anggota mengembalikan buku.

Walaupun mungkin masih banyak activity-activity lain yang terkait dengan perpustakaan tetapi bukan merupakan business process yang utama dari perpustakaan.

Menambah anggota

Ada dua class Boundary yaitu MenuUtama dan FormPendaftaran Anggota, satu class Control yaitu ProsesPendaftaranAnggota, dan satu class Entity yaitu Member.



Mencetak Kartu Anggota

