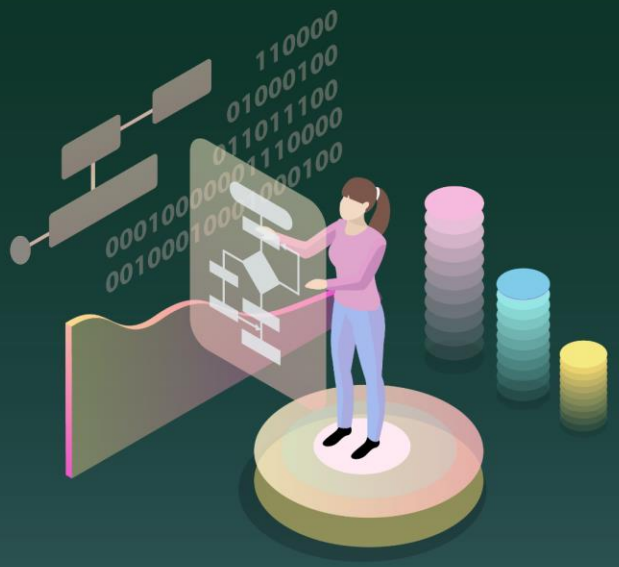
# Unified Modeling Language (UML)

Romi Satria Wahono
*romi@romisatriawahono.net*
*http://romisatriawahono.net*
*08118228331*



# Romi Satria Wahono

- SMA Taruna Nusantara Magelang (1993)
- B.Eng, M.Eng and Ph.D in Software Engineering
  Saitama University Japan (1994-2004)
  Universiti Teknikal Malaysia Melaka (2014)
- Core Competency in Enterprise Architecture, Software Engineering and Machine Learning
- LIPI Researcher (2004-2007)
- Founder and CEO:
  - PT Brainmatics Cipta Informatika (2005)
  - PT IlmuKomputerCom Braindevs Sistema (2014)
- Professional Member of IEEE, ACM and PMI
- IT and Research Award Winners from WSIS (United Nations), Kemdikbud, Ristekdikti, LIPI, etc
- SCOPUS/ISI Indexed Q1 Journal Reviewer: Information and Software Technology, Journal of Systems and Software, Software: Practice and Experience, Empirical Software Engineering, etc
- Industrial IT Certifications: TOGAF, ITIL, CCAI, CCNA, etc
- Enterprise Architecture Consultant: KPK, RistekDikti, INSW, BPPT, Kemsos Kemenkeu (Itjend, DJBC, DJPK), Telkom, FIF, PLN, PJB, Pertamina EP, etc

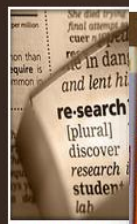
RomiSatriaWahono.Net

Lecture Notes in Software Engineering, Computing Research and Technopreneurship

HOME ABOUT ME IN THE NEWS LECTURES PUBLICATIONS RESEARCH


BAGAIMANA MELAKUKAN PENELITIAN YANG BAIK?

Pada artikel ini, saya mencoba merangkumkan tahapan melakukan penelitian yang



TAHAPAN MEL
PENELITIAN U
MAHASISWA

Jujur, secara um
kecewa dengan
mahasiswa tingk
masuk lewat em

RomiSatriaWahono.Net

Lecture Notes in Software Engineering, Computing Research and Technopreneurship

HOME ABOUT ME IN THE NEWS LECTURES PUBLICATIONS RESEARCH

Home » Lectures

universitas di Indonesia. Seluruh materi kuliah bisa diunduh dan
nan mata kuliah memuat course description, standard competency,
ook yang digunakan.

Research Methodology (updated January 2015)
Data Mining (updated January 2015)
Theory of Computation (updated March 2015)

ava Fundamentals (updated October 2013)
ava Enterprise Edition

ystems Analysis and Design (updated January 2015)
Business Process Model and Notation (updated January 2015)
oftware Engineering
oftware Testing
oftware Quality Assurance
roject Management

OGAF 9.1 Fundamental
OGAF 9.1 Foundation
OGAF 9.1 Certified

YouTube ID

Upload

189 subscribers 4,535 views Video Manager View as public


Romi Satria Wahono

Home Videos Playlists Channels Discussion About

Uploads

Date added (newest - oldest)


Menjadi Programmer Technopreneur
116 views • 3 weeks ago


Kuliah 10 Menit tentang Enterprise Architecture
816 views • 3 weeks ago


Kuliah 20 Menit tentang Metodologi Penelitian
1,756 views • 4 weeks ago


Kuliah 10 Menit tentang Data Mining
1,898 views • 1 month ago

BRAINDEVS

1 Hour Online Training

# Software Engineering Research Trends

**Romi Satria Wahono, Ph.D**
Researcher & Technopreneur. Founder & CEO PT Braindevs & PT Brainmatics.
TOGAF & ITIL Certified. Enterprise Architect di KPK, Pertamina EP, BPPT, PLN, PJB, FIF, INSW, RistekDikti, Kemenkeu. SCOPUS Q1 Journal Reviewer:
Information & Software Technology, Journal of System & Software, etc.

GRATIS

**Jum'at, 8 Mei 2020**
**09.00 - 10.00 WIB**
Terbatas 500 Peserta, Registrasi Sebelum
Kamis, 7 Mei 2020, Pukul 17:00 Melalui
*sert1hour.brainmatics.com*


#TrainingDirumahAja

Contact Us:
Irma 08118228881
Annisa 08118228882
Vina 08118228883
Lina 08118228884
Rachma 08118228885

Menara Bidakara 1 Suite 0205
Jl. Gatot Subroto Kav 71-73 Jakarta 12870
info@brainmatics.com http://brainmatics.com
BrainmaticsID

# Course Outline

## 1. System Planning

1.1 Siklus dan Metodologi Pengembangan Software
1.2 System Request dan Feasibility Analysis

## 2. Systems Analysis

2.1 Requirement Gathering
2.2 Identifikasi Proses Bisnis dengan Use Case Diagram
2.3 Pemodelan Proses Bisnis dengan AD atau BPMN
2.4 Realisasi Proses Bisnis dengan Sequence Diagram

## 3. Systems Design

3.1 Perancangan Class Diagram
3.3 Perancangan User Interface Design
3.4 Perancangan Data Model
3.5 Perancangan Deployment Diagram

# 1. Systems Planning

# 1.1 Siklus dan Metodologi Pengembangan Software

# Siklus Pengembangan Software:

## Alur, Peran, dan Tahapan (*Deliverable*) *(Tilley, 2012) (Dennis, 2016) (Valacich, 2017)*

1. **User/Product Owner** membawa permintaan kebutuhan (perubahan) software (System Request) ke System Analyst
2. **System Analyst** membuat analisis kelayakan (Feasibility Analysis) dari System Request tersebut

**Planning**
*(System Proposal)*

3. Setelah dinyatakan layak, System Analyst melakukan analysis dan design, dan hasilnya adalah System Specification
   - **Business Analyst** membantu System Analyst memahami proses bisnis dari software yang akan dibangun

**Analysis and Design**
*(System Specification)*

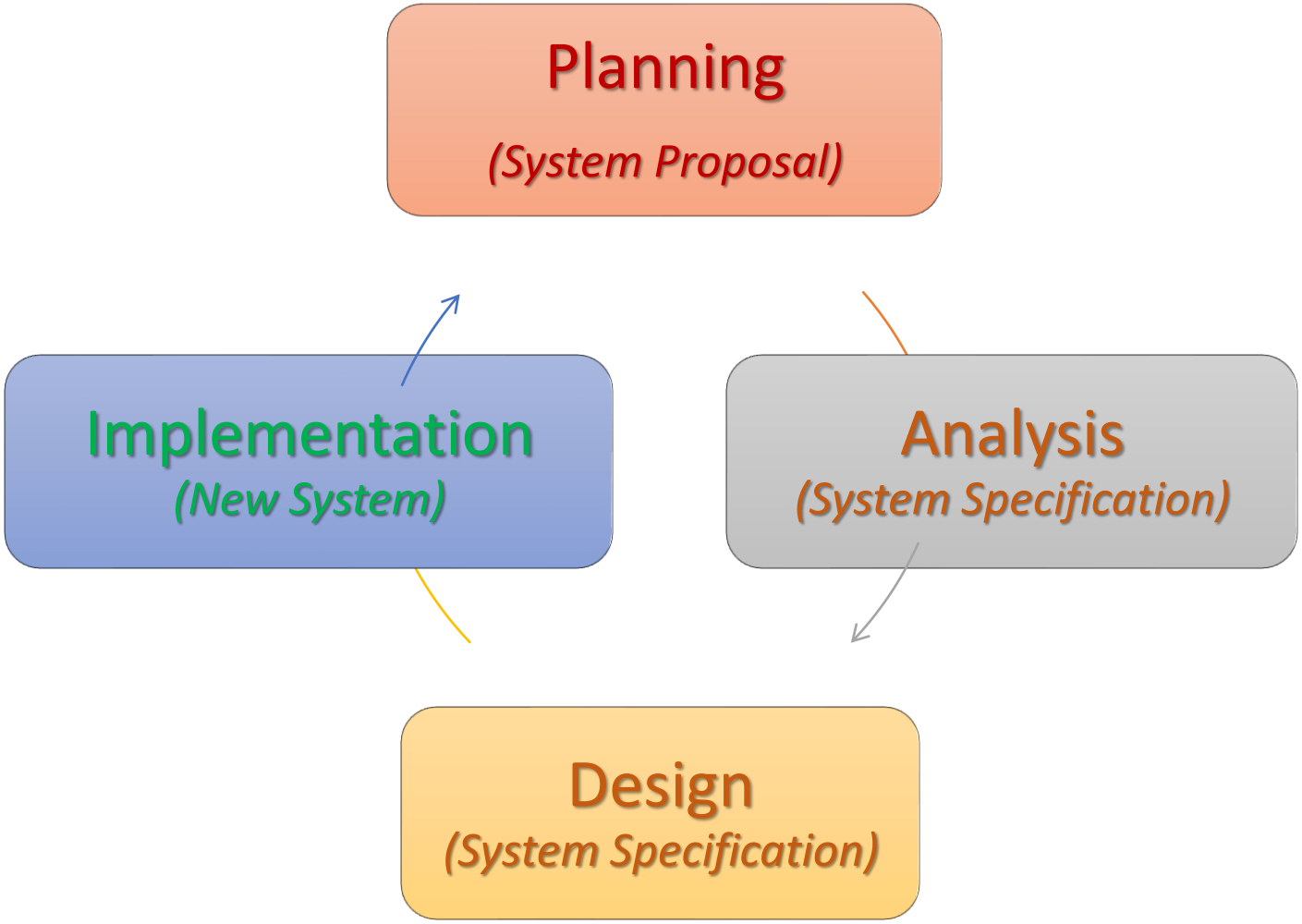
4. System Specification diserahkan oleh System Anayst ke **Programmer** untuk dilakukan Konstruksi (*Coding*)
5. Hasil Konstruksi berupa Kode Program diserahkan ke **Software Tester** untuk dilakukan Pengujian (*Unit, Integration, System, User Acceptance Testing*)
6. Instalasi (*delivery*) software dan manajemen perubahan
   - Software = Kode Program + Dokumentasi (Pengembangan dan Penggunaan)

**Implementation**
*(Software)*

7. Siklus kembali ke 1 apabila ada permintaan perubahan (Permintaan Perubahan Software)

**Maintenance**
*(Updated Software)*

# Siklus Pengembangan Software




*(Tilley, 2012)* *(Dennis, 2016)* *(Valacich, 2017)*

# Metodologi Pengembangan Software (Model Process)

- A formalized approach to implementing the Software Development Life Cycle *(Dennis, 2012)*

- A simplified representation of a software process *(Sommerville, 2015)*

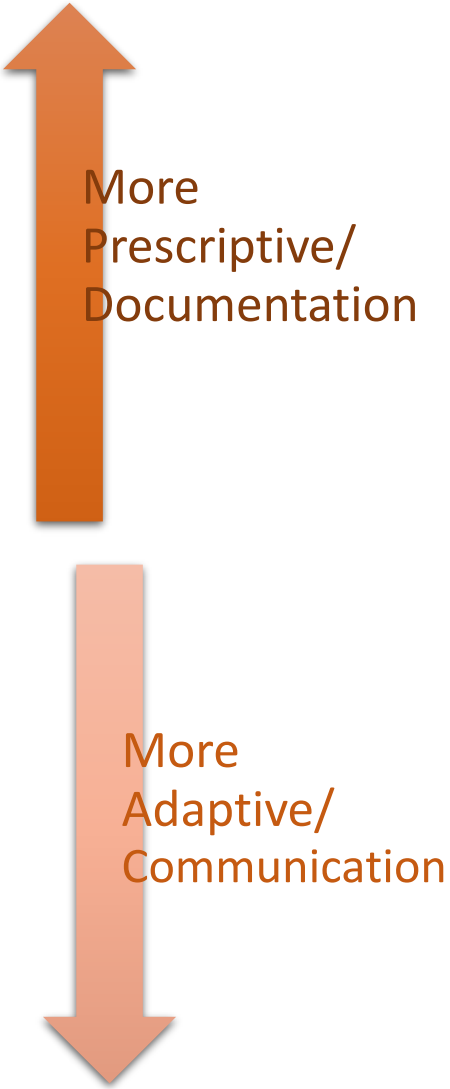
- A distinct set of activities, actions, tasks, milestones, and work products required to engineer high quality software *(Pressman, 2015)*

# Metodologi Pengembangan Software

1. **Structured Design**
   (Prescriptive) (1967- )
   - Waterfall method
   - Parallel development

2. **Rapid Application Development**
   (Iterative) (1985-)
   - Phased Development
   - Prototyping

3. **Agile Development**
   (Adaptive) (1995-)
   - Extreme Programming (XP)
   - Scrum




*Compiled from (Dennis, Wixom and Tegarden, 2016)*

# Planning

**System Request: Sistem Penjualan Musik Online**

**Project Sponsor:** Margaret Mooney, Vice President of Marketing

**Business Needs:** Project ini dibangun untuk:

1. Mendapatkan pelanggan baru lewat Internet

**Studi Kelayakan Sistem Penjualan Musik Online**

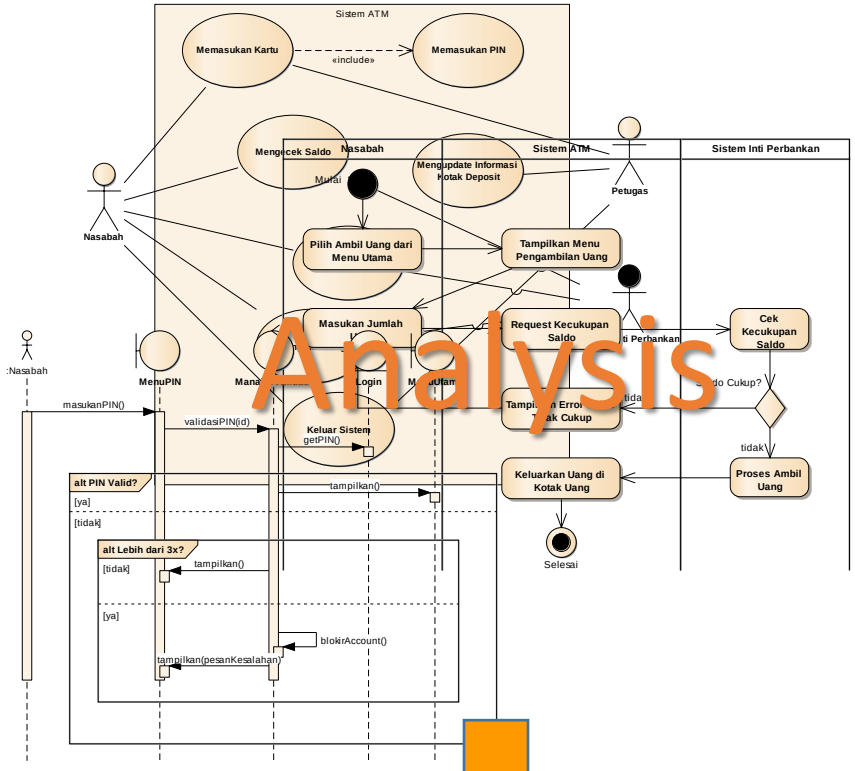
Margaret Mooney dan Alec Adams membuat studi kelayakan untuk pengembangan Sistem Penjualan Musik Online

**Kelayakan Teknis**

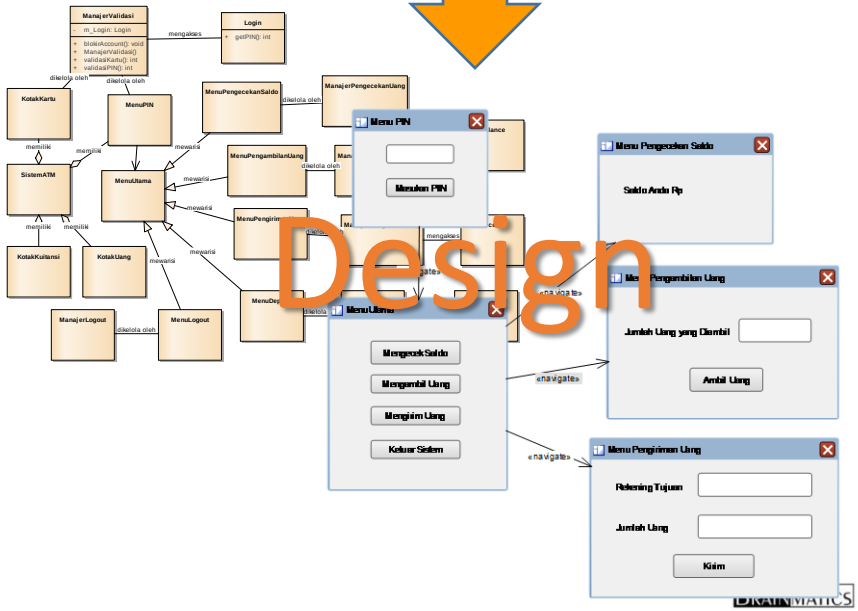
Sistem penjualan musik online layak secara teknis, meskipun memiliki beberapa risiko.

| | 2016 | 2017 | 2018 |
|---|---|---|---|
| Peningkatan penjualan dari pelanggan baru | 0 | 400,000,000 | 500,000,000 |
| Peningkatan penjualan dari pelanggan lama | 0 | 600,000,000 | 700,000,000 |
| Pengurangan biaya operasional dan telepon | 0 | 100,000,000 | 100,000,000 |
| **Total Benefits:** | **0** | **1,100,000,000** | **1,300,000,000** |
| **PV of Benefits:** | | | |
| **PV of All Benefits:** | | | |
| Honor Tim (Planning, Analysis, Design dan Implementation) | | | |
| Honor Konsultan Infrastruktur Internet | | | |
| **Total Development Costs:** | **450,000,000** | **0** | **0** |
| Honor Pengelola Web | 60,000,000 | 70,000,000 | 80,000,000 |
| Biaya Lisensi Software | 50,000,000 | 60,000,000 | 70,000,000 |
| Hardware upgrades | 100,000,000 | 100,000,000 | 100,000,000 |
| Biaya Komunikasi | 20,000,000 | 30,000,000 | 40,000,000 |
| Biaya Marketing | 100,000,000 | 200,000,000 | 300,000,000 |
| **Total Operational Costs:** | **330,000,000** | **460,000,000** | **590,000,000** |
| **Total Costs:** | **780,000,000** | **460,000,000** | **590,000,000** |
| **PV of Costs:** | **735,849,057** | **409,398,362** | **495,375,377** |
| **PV of all Costs:** | **735,849,057** | **1,145,247,419** | **1,640,622,796** |
| **Total Project Costs Less Benefits:** | **-780,000,000** | **640,000,000** | **710,000,000** |
| **Yearly NPV:** | **-735,849,057** | **569,507,722** | **596,129,691** |
| **Cumulative NPV:** | **-735,849,057** | **-166,251,335** | **429,878,356** |
| **Return on Investment (ROI) di Tahun 3: 26.2%** | **429,878,356/1,640,622,796** | | **0.262021445** |
| **Break-even Point (BEP): 2.28 tahun** | **2 + (596,129,691-429,878,356)/ 596,129,691** | | **2.278884507** |

# Analysis




# Design




# Implementation

**System Request: Sistem Penjualan Musik Online**

**Project Sponsor:** Margaret Mooney, Vice President of Marketing

**Business Needs:** Project ini dibangun untuk:

1. Mendapatkan pelanggan baru lewat Internet

**Studi Kelayakan Sistem Penjualan Musik Online**

Margaret Mooney dan Alec Adams membuat studi kelayakan untuk pengembangan Sistem Penjualan Musik Online

**Kelayakan Teknis**

Sistem penjualan musik online layak secara teknis, meskipun memiliki beberapa risiko.

| | 2016 | 2017 | 2018 |
|---|---|---|---|
| Peningkatan penjualan dari pelanggan baru | 0 | 400,000,000 | 500,000,000 |
| Peningkatan penjualan dari pelanggan lama | 0 | 600,000,000 | 700,000,000 |
| Pengurangan biaya operasional dan telepon | 0 | 100,000,000 | 100,000,000 |
| **Total Benefits:** | **0** | **1,100,000,000** | **1,300,000,000** |
| **PV of Benefits:** | | | |
| **PV of All Benefits:** | | | |
| Honor Tim (Planning, Analysis, Design and Implementation) | | | |
| Honor Konsultan Infrastruktur Internet | | | |
| **Total Development Costs:** | **450,000,000** | **0** | **0** |
| Honor Pengelola Web | 60,000,000 | 70,000,000 | 80,000,000 |
| Biaya Lisensi Software | 50,000,000 | 60,000,000 | 70,000,000 |
| Hardware upgrades | 100,000,000 | 100,000,000 | 100,000,000 |
| Biaya Komunikasi | 20,000,000 | 30,000,000 | 40,000,000 |
| Biaya Marketing | 100,000,000 | 200,000,000 | 300,000,000 |
| **Total Operational Costs:** | **330,000,000** | **460,000,000** | **590,000,000** |
| **Total Costs:** | **780,000,000** | **460,000,000** | **590,000,000** |
| **PV of Costs:** | **735,849,057** | **409,398,362** | **495,375,377** |
| **PV of all Costs:** | **735,849,057** | **1,145,247,419** | **1,640,622,796** |
| **Total Project Costs Less Benefits:** | **-780,000,000** | **640,000,000** | **710,000,000** |
| **Yearly NPV:** | **-735,849,057** | **569,507,722** | **596,129,691** |
| **Cumulative NPV:** | **-735,849,057** | **-166,251,435** | **429,878,356** |
| **Return on Investment (ROI) di Tahun 3: 26.2%** | **429,878,356/1,640,622,796** | | **0.262021445** |
| **Break-even Point (BEP): 2.28 tahun** | **2 + (596,129,691-429,878,356)/ 596,129,691** | | **2.278884507** |

Planning

Analysis

Design

Implementation

Versi 1

Versi 2

**System Request: Sistem Penjualan Musik Online**

| | |
|---|---|
| **Project Sponsor:** | Margaret Mooney, Vice President of Marketing |
| **Business Needs:** | Project ini dibangun untuk: |
| | 1. Mendapatkan pelanggan baru lewat Internet |
| | 2. M |
| **Business Requirement** | |
| Sistem yang menduku | |
| 1. Fitur Pencarian Pro | |
| 2. Fitur Pencarian Tok | |
| 3. Fitur Pemesanan Pr | |
| 4. Fi | |

## Studi Kelayakan Sistem Penjualan Musik Online

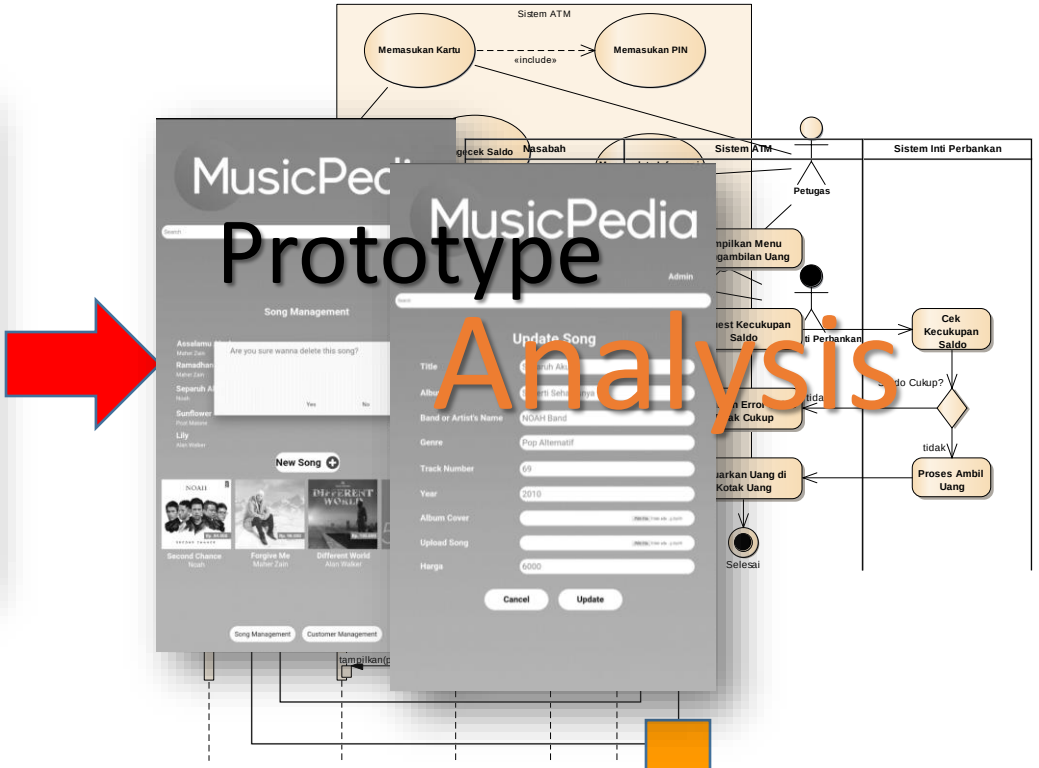
Margaret Mooney dan Alec Adams membuat studi kelayakan untuk pengembangan Sistem Penjualan Musik Online

### Kelayakan Teknis

Sistem penjualan musik online layak secara teknis, meskipun memiliki beberapa risiko.

| | 2016 | 2017 | 2018 |
|---|---|---|---|
| Peningkatan penjualan dari pelanggan baru | 0 | 400,000,000 | 500,000,000 |
| Peningkatan penjualan dari pelanggan lama | 0 | 600,000,000 | 700,000,000 |
| Pengurangan biaya operasional dan telepon | 0 | 100,000,000 | 100,000,000 |
| **Total Benefits:** | **0** | **1,100,000,000** | **1,300,000,000** |
| **PV of Benefits:** | | | |
| **PV of All Benefits:** | | | |
| Honor Tim (Planning, Analysis, Design dan Implementation) | | | |
| Honor Konsultan Infrastruktur Internet | | | |
| **Total Development Costs:** | **450,000,000** | **0** | **0** |
| Honor Pengelola Web | 60,000,000 | 70,000,000 | 80,000,000 |
| Biaya Lisensi Software | 50,000,000 | 60,000,000 | 70,000,000 |
| Hardware upgrades | 100,000,000 | 100,000,000 | 100,000,000 |
| Biaya Komunikasi | 20,000,000 | 30,000,000 | 40,000,000 |
| Biaya Marketing | 100,000,000 | 200,000,000 | 300,000,000 |
| **Total Operational Costs:** | **330,000,000** | **460,000,000** | **590,000,000** |
| **Total Costs:** | **780,000,000** | **460,000,000** | **590,000,000** |
| **PV of Costs:** | **735,849,057** | **409,398,362** | **495,375,377** |
| **PV of all Costs:** | **735,849,057** | **1,145,247,419** | **1,640,622,796** |
| **Total Project Costs Less Benefits:** | **-780,000,000** | **640,000,000** | **710,000,000** |
| **Yearly NPV:** | **-735,849,057** | **569,507,722** | **596,129,691** |
| **Cumulative NPV:** | **-735,849,057** | **-166,251,435** | **429,878,356** |
| **Return on Investment (ROI) di Tahun 3: 26.2%** | **429,878,356/1,640,622,796** | | **0.262021445** |
| **Break-even Point (BEP): 2.28 tahun** | **2 + (596,129,691-429,878,356)/ 596,129,691** | | **2.278884507** |

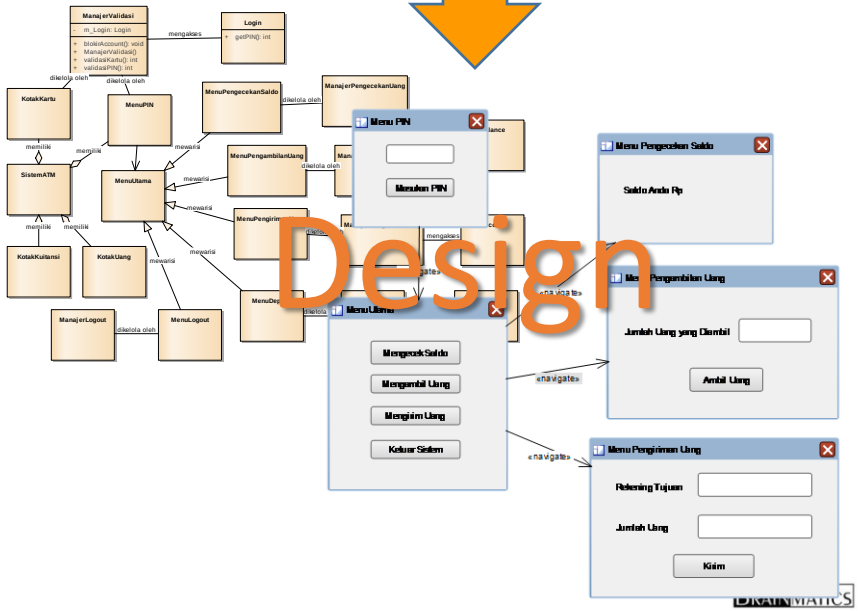
Planning




Prototype

Analysis


Implementation




Design

**System Request: Sistem Penjualan Musik Online**

**Project Sponsor:** Margaret Mooney, Vice President of Marketing

**Business Needs:** Project ini dibangun untuk:

1. Mendapatkan pelanggan baru lewat Internet

**Studi Kelayakan Sistem Penjualan Musik Online**

Margaret Mooney dan Alec Adams membuat studi kelayakan untuk pengembangan Sistem Penjualan Musik Online

**Kelayakan Teknis**

Sistem penjualan musik online layak secara teknis, meskipun memiliki beberapa risiko.

| | 2016 | 2017 | 2018 |
|---|---|---|---|
| Peningkatan penjualan dari pelanggan baru | 0 | 400,000,000 | 500,000,000 |
| Peningkatan penjualan dari pelanggan lama | 0 | 600,000,000 | 700,000,000 |
| Pengurangan biaya operasional dan telepon | 0 | 100,000,000 | 100,000,000 |
| **Total Benefits:** | **0** | **1,100,000,000** | **1,300,000,000** |
| **PV of Benefits:** | | | |
| **PV of All Benefits:** | | | |
| Honor Tim (Planning, Analysis, Design and Implementation) | | | |
| Honor Konsultan Infrastruktur Internet | | | |
| **Total Development Costs:** | **450,000,000** | **0** | **0** |
| Honor Pengelola Web | 60,000,000 | 70,000,000 | 80,000,000 |
| Biaya Lisensi Software | 50,000,000 | 60,000,000 | 70,000,000 |
| Hardware upgrades | 100,000,000 | 100,000,000 | 100,000,000 |
| Biaya Komunikasi | 20,000,000 | 30,000,000 | 40,000,000 |
| Biaya Marketing | 100,000,000 | 200,000,000 | 300,000,000 |
| **Total Operational Costs:** | **330,000,000** | **460,000,000** | **590,000,000** |
| **Total Costs:** | **780,000,000** | **460,000,000** | **590,000,000** |
| **PV of Costs:** | **735,849,057** | **409,398,362** | **495,375,377** |
| **PV of all Costs:** | **735,849,057** | **1,145,247,419** | **1,640,622,796** |
| **Total Project Costs Less Benefits:** | **-780,000,000** | **640,000,000** | **710,000,000** |
| **Yearly NPV:** | **-735,849,057** | **569,507,722** | **596,129,691** |
| **Cumulative NPV:** | **-735,849,057** | **-166,251,435** | **429,878,356** |
| **Return on Investment (ROI) di Tahun 3: 26.2%** | **429,878,356/1,640,622,796** | | **0.262021445** |
| **Break-even Point (BEP): 2.28 tahun** | **2 + (596,129,691-429,878,356)/ 596,129,691** | | **2.278884507** |

Planning

Sprint 1

Analysis

Sprint 2

Hasil Sprint 1

Hasil Sprint 2

Implementation

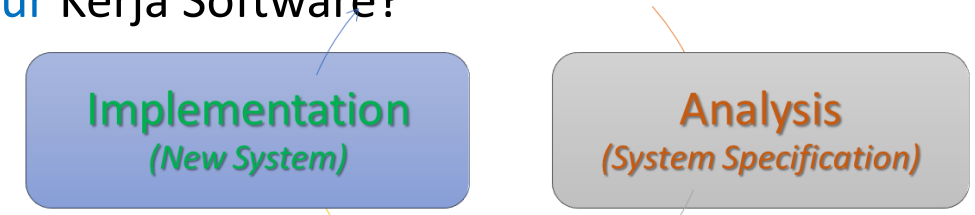
Design

# 1.2 System Request dan Feasibility Analysis

# Siklus Pengembangan Software

1. **Planning**: Mengapa Software harus Dikembangkan?
   - System request
   - Feasibility Analysis (project size estimation)

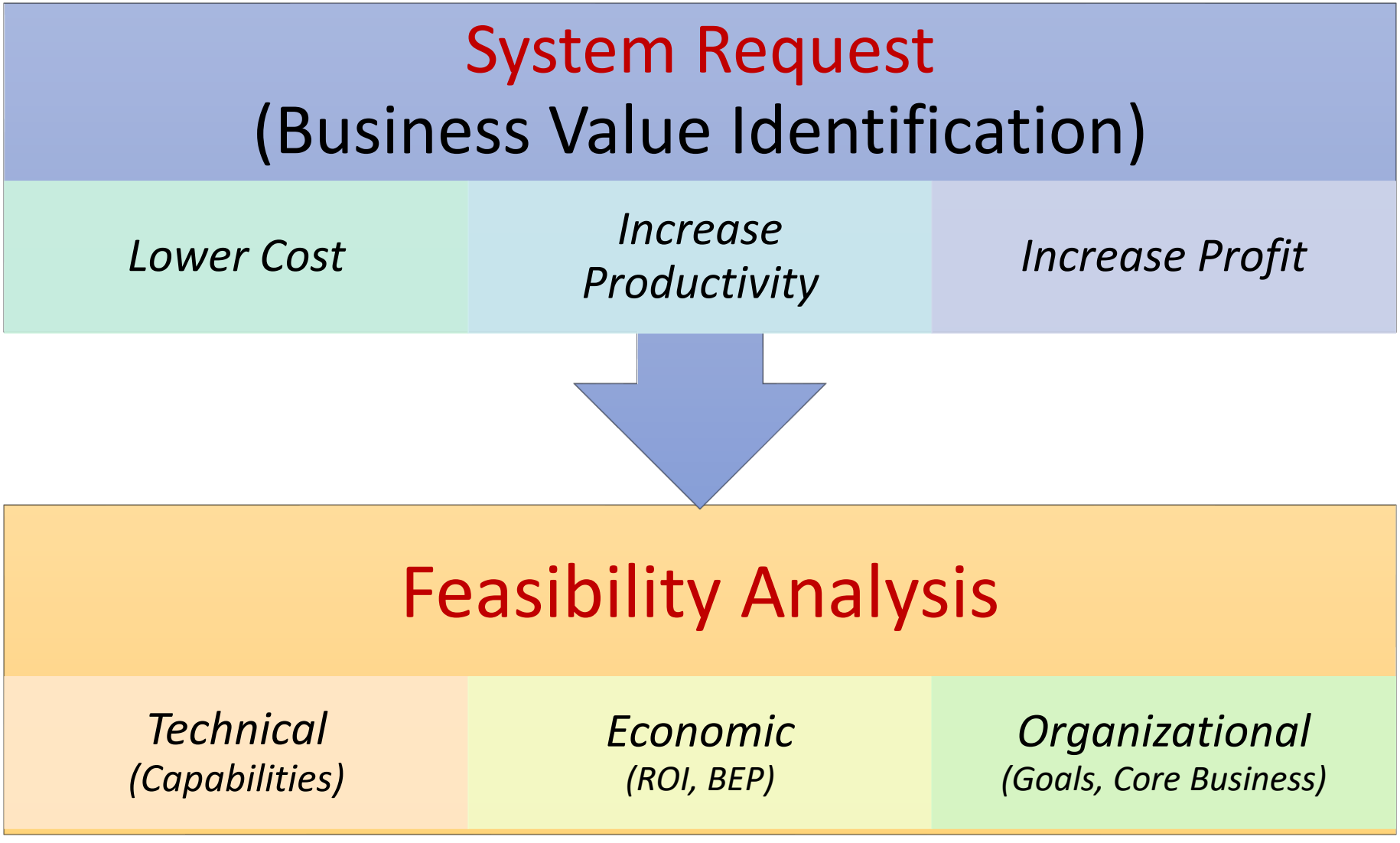

2. **Analysis**: Siapa Pengguna dan Alur Kerja Software?
   - Requirement gathering
   - Business process modeling




3. **Design**: Bagaimana Komposisi dari Software?
   - Program design
   - User interface design
   - Data design


4. **Implementation**: Konstruksi dan Penyerahan Software
   - System construction
   - Testing
   - Documentation
   - Installation

# Planning



# System Request

| Elemen | Deskripsi | Contoh |
|---|---|---|
| **Business Need** | • The business-related reason for initiating the software development project<br>• Reason prompting the project, and why the project should be funded? | • Meningkatkan penjualan<br>• Mengurangi biaya operasional<br>• Meningkatkan produktifitas pegawai<br>• Meningkatkan kualitas layanan<br>• Mengurangi kebocoran/kecurangan<br>• Mengurangi cacat produksi<br>• Meningkatkan efisiensi kerja |
| **Business Value** | • The benefits that the software will create for the organization<br>• Tangible value (a quantiable value) and intangible value (intuitive believe) | • Peningkatan penjualan 3%<br>• Pengurangan biaya operasional 10%<br>• Peningkatan produktifitas pegawai 10% (dihitung rasio pekerjaan dan gaji)<br>• Pengurangan cacat produksi 20%<br>• Peningkatan efisiensi kerja 20% |
| **Business Requirements** | • The business capabilities that software will provide<br>• Can be replaced by Use Case Diagram | • Fitur registrasi, login, dan logout<br>• Fitur pengelolaan data pengguna<br>• Fitur pengiriman notifikasi otomatis<br>• Fitur cetak laporan bulanan dan tahunan |

# Software Berkualitas?

Software quality is *(IEEE, 1991)*:

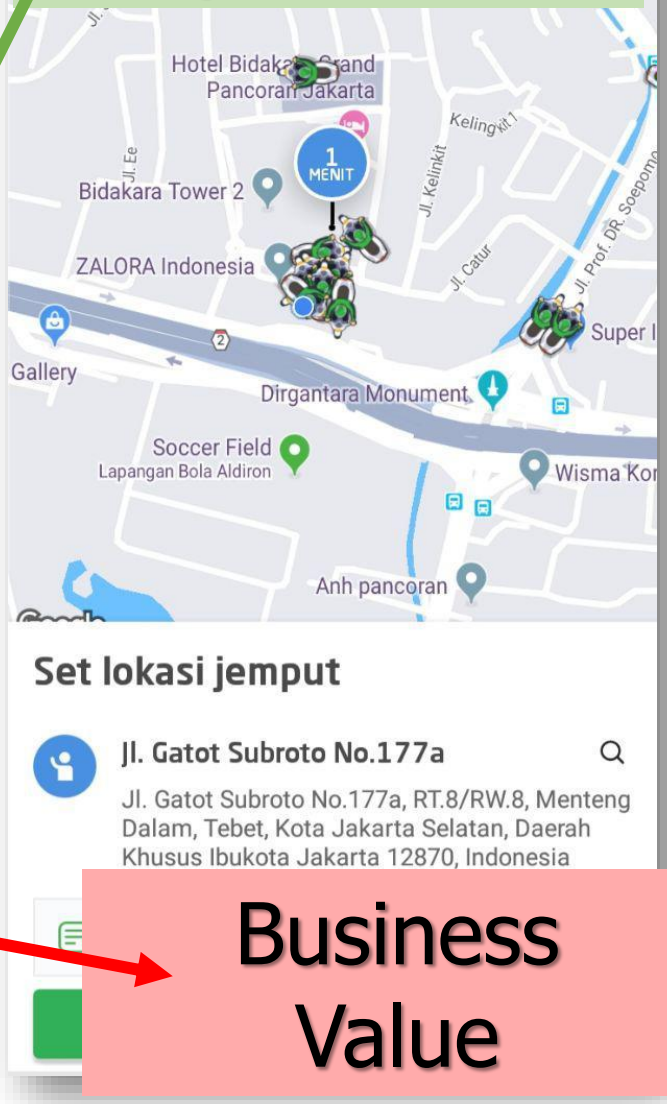
1. The degree to which a system, component, or

**Sesuai Kebutuhan**

2. The degree to which a system, component, or process meets customer

**Ada Keuntungan**

**Business Requirements**




**Business Value**

# System Request: Sistem Penjualan Musik Online

| | |
|---|---|
| **Project Sponsor:** | Margaret Mooney, Vice President of Marketing |
| **Business Needs:** | Project ini dibangun untuk:<br>1. Mendapatkan pelanggan baru lewat Internet<br>2. Meningkatkan efisiensi penanganan masalah pelanggan melalui internet |

**Business Requirements:**

Sistem yang mendukung penjualan musik secara online. Fitur-fitur yang harus ada:

1. Fitur Pencarian Produk
2. Fitur Pencarian Toko yang Menyediakan Stok Produk
3. Fitur Pemesanan Produk Melalui Toko yang Menyediakan
4. Fitur Pembayaran dengan Berbagai Pilihan Pembayaran

**Business Value:**

*Intangible Value:*

- Meningkatkan kenyamanan dan kepuasan pelanggan
- Meningkatkan brand recognition tentang perusahaan di dunia Internet

*Tangible Value:*

1. Meningkatkan penjualan dari pelanggan baru lewat Internet:
   - Rp 400 juta peningkatan penjualan dari pelanggan baru dan Rp 600 juta dari pelanggan lama
2. Mengurangi biaya operasional untuk menangani komplain dari pelanggan
   - Rp 100 juta pengurangan tahunan biaya telepon untuk menangani pelanggan

# Studi Kelayakan Sistem Penjualan Musik Online

Margaret Mooney dan Alec Adams membuat studi kelayakan untuk pengembangan Sistem Penjualan Musik Online

## Kelayakan Teknis

Sistem penjualan musik online layak secara teknis, meskipun memiliki beberapa risiko.

Risiko Berhubungan dengan **Kefamilieran dengan Aplikasi**: Resiko Tinggi

- Divisi Marketing tidak memiliki pengalaman menggunakan sistem penjualan online
- Divisi IT memiliki pemahaman yang baik tentang sistem penjualan offline, akan tetapi tidak berpengalaman mengembangkan sistem penjualan musik online

Risiko Berhubungan dengan **Kefamilieran dengan Teknologi**: Resiko Sedang

- Divisi IT tidak menguasai masalah infrastruktur dan ISP, tetapi akan menyewa konsultan
- Divisi IT cukup familier dengan framework dan IDE yang akan digunakan
- Divisi Marketing tidak memiliki pengalaman menggunakan teknologi Web

Risiko berhubungan dengan **Ukuran Project**: Risiko Rendah

- Perusahaan memiliki total 30 orang pengembang
- Project dikerjakan oleh 5 orang pengembang dengan estimasi waktu 6 bulan

**Kompatibilitas** dengan sistem dan infrastruktur yang ada: Risiko Rendah

- Sistem pemesanan yang ada sekarang menggunakan *open standard*, jadi sangat kompatibel dengan sistem penjualan berbasis web yang akan dibangun

## Kelayakan Ekonomi

Cost benefit analysis telah dilakukan. Sistem Penjualan musik online memiliki peluang yang baik untuk bisa meningkatkan pendapatan perusahaan.

- Return on Investment (ROI) setelah 3 tahun: **31%**
- Break-even point (BEP): **2.25 tahun**
- Total keuntungan setelah 3 tahun: **Rp. 503.559.986,-**

## Kelayakan Organisasi

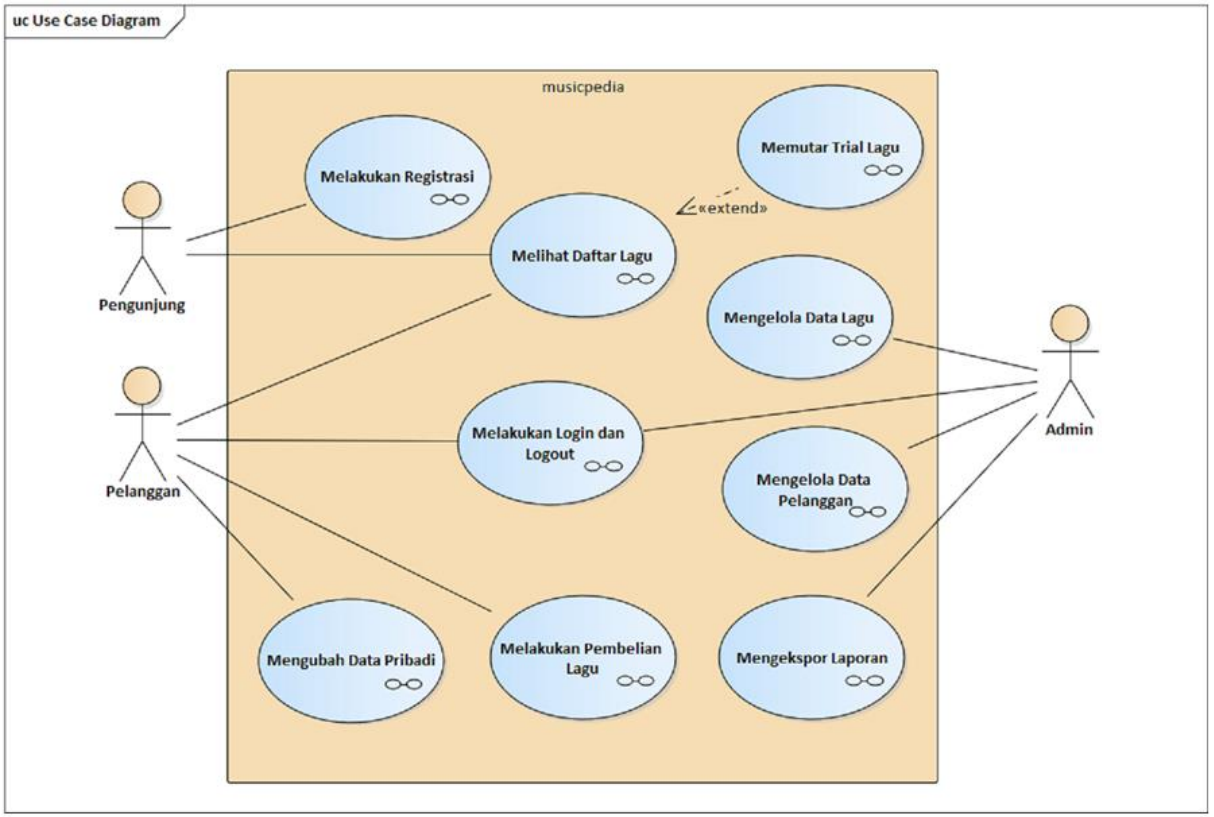
- Secara organisasi, resikonya rendah. Tujuan dari pengembangan sistem penjualan musik online adalah meningkatkan penjualan perusahaan. Dan ini selaras dengan KPI marketing yang ke arah peningkatan kuantitas penjualan
- Project champion dari pengembangan sistem penjualan musik online ini adalah Margaret Mooney, Vice President of Marketing

| | 2019 | 2020 | 2021 |
|---|---|---|---|
| Peningkatan penjualan dari pelanggan baru | 0 | 400,000,000 | 500,000,000 |
| Peningkatan penjualan dari pelanggan lama | 0 | 600,000,000 | 700,000,000 |
| Pengurangan biaya operasional dan telepon | 0 | 100,000,000 | 100,000,000 |
| **Total Benefits:** | **0** | **1,100,000,000** | **1,300,000,000** |
| **PV of Benefits:** | **0** | **978,996,084** | **1,091,505,068** |
| **PV of All Benefits:** | **0** | **978,996,084** | **2,070,501,152** |
| Honor Tim (Analysis, Design and Implementation) | 360,000,000 | 0 | 0 |
| Honor Konsultan | 90,000,000 | 0 | 0 |
| **Total Development Costs:** | **450,000,000** | **0** | **0** |
| Honor Pengelola Web | 60,000,000 | 70,000,000 | 80,000,000 |
| Biaya Lisensi Software | 50,000,000 | 60,000,000 | 70,000,000 |
| Hardware upgrades | 100,000,000 | 100,000,000 | 100,000,000 |
| Biaya Komunikasi | 20,000,000 | 30,000,000 | 40,000,000 |
| Biaya Marketing | 100,000,000 | 200,000,000 | 300,000,000 |
| **Total Operational Costs:** | **330,000,000** | **460,000,000** | **590,000,000** |
| **Total Costs:** | **780,000,000** | **460,000,000** | **590,000,000** |
| **PV of Costs:** | **735,849,057** | **409,398,362** | **495,375,377** |
| **PV of all Costs:** | **735,849,057** | **1,145,247,419** | **1,640,622,796** |
| **Total Project Costs Less Benefits:** | **-780,000,000** | **640,000,000** | **710,000,000** |
| **Yearly NPV:** | **-735,849,057** | **569,597,722** | **669,811,321** |
| **Cumulative NPV:** | **-735,849,057** | **-166,251,335** | **503,559,986** |
| **Return on Investment (ROI) di Tahun 3: 30.70%** | **-100.00%** | **-0.145166304** | **0.306932213** |
| **Break-even Point (BEP): 2.25 tahun** | | | **2.248206218** |

# Software Request

| musicpedia | | | | |
|---|---|---|---|---|
| **Date** | 26 Oktober 2018 | | | |
| **Description** | Musicpedia adalah aplikasi layanan download musik & audio dimana saja dan kapan saja, menawarkan akses lengkap ke jutaan lagu dari semua artis papan atas di industri musik barat maupun musik lokal, mendengarkan musik baru dan top musik dunia dalam bentuk audio mp3 secara offline | | | |
| **Project Sponsor** | Wahyu Utomo, VP Business Development, PT Musika Indonesia | | | |
| **Business Need** | 1. Tidak Setuju | 2. Ragu-Ragu | 3. Setuju | 4. Sangat Setuju |
| Aplikasi yang dikembangkan mampu meningkatkan pendapatan perusahaan? | ☐ | ☐ | ☐ | ☑ |
| Aplikasi yang dikembangkan mampu mengurangi biaya operasional perusahaan? | ☐ | ☐ | ☑ | ☐ |
| Aplikasi yang dikembangkan mampu meningkatkan produktifitas kerja pegawai? | ☐ | ☑ | ☐ | ☐ |
| Aplikasi yang dikembangkan mampu meningkatkan nilai tambah perusahaan yang bersifat intangible? | ☐ | ☐ | ☑ | ☐ |
| **Business Value** | Intangible Value:<br>a. Meningkatkan brand recognition perusahaan di dunia internet<br>b. Meningkatkan produktivitas kerja pegawai dan mengurangi kuantitas pegawai<br><br>Tangible Value:<br>a. Mengurangi biaya operasional perusahaan:<br>- Sewa ruangan: Rp120.000.000,-<br>- Biaya komunikasi: Rp6.000.000,-<br>b. Meningkatkan penjualan musik: Rp400.000.000,- | | | |

## Business Requirements



# *Technical Feasibility*

| musicpedia | | | | |
|---|---|---|---|---|
| **Date: 26 Oktober 2018** | | | | |
| Penjelasan isian | 1. Sangat Kurang | 2. Kurang | 3. Baik | 4. Sangat Baik |

| Kefamiliaran dengan Aplikasi | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Pengguna familiar terhadap pengoperasian aplikasi ini. | ☐ | ☐ | ☑ | ☐ |
| Pengembang familiar terhadap pengembangan aplikasi ini. | ☐ | ☐ | ☐ | ☑ |
| | | | | |
| **Kefamiliaran dengan Teknologi** | **1** | **2** | **3** | **4** |
| Pengguna familiar dengan teknologi pendukung aplikasi. | ☐ | ☑ | ☐ | ☐ |
| Pengembang familiar mengembangkan aplikasi dengan platform, bahasa pemrograman dan tool IDE yang dipilih. | ☐ | ☐ | ☐ | ☑ |
| | | | | |
| **Ukuran Proyek** | | | | |
| Jumlah pengembang yang dibutuhkan. | 7 Man/Month | | | |
| Waktu yang dibutuhkan dalam mengembangkan aplikasi ini. | 6 Month | | | |
| | | | | |
| **Kompatibilitas** | **1** | **2** | **3** | **4** |
| Kebutuhan pengguna terhadap kompatibilitas aplikasi untuk terintegrasi dengan aplikasi lain. | ☐ | ☐ | ☑ | ☐ |
| Kompatibilitas aplikasi terhadap teknologi yang ada pada organisasi. | ☐ | ☐ | ☐ | ☑ |
| | | | | |

| Secara analisi kelayakan teknis, apakah aplikasi layak dikembangkan sesuai kriteria di atas? | ☑ Layak | ☐ Tidak Layak |
|---|---|---|

# *Technical Feasibility*

**Use Case Points**

| Tahap 1 - Menghitung Person Hours (PH) | | | | |
|---|---|---|---|---|
| Use Case Points (UCP) | | Person Hours Multiplier (PHM) | Person Hours (PH) | |
| 51 | | 20 | 1020 | |
| 51 | | 28 | 1428 | |
| **Tahap 2 - Menghitung Person Month (PM)** | | | | |
| PHM | Person Hours (PH) | Lama Bekerja Perhari | Jumlah Bekerja Sebulan | Person Months (PM) |
| 20 | 1020 | 8 | 22 | 5.80 |
| | 1020 | 10 | 26 | 3.92 |
| 28 | 1428 | 8 | 22 | 8.11 |
| | 1428 | 10 | 26 | 5.49 |
| **Tahap 3 - Menghitung Time (Month)** | | | | |
| PHM | Formula Penghitung Waktu | | Jumlah Bekerja Sebulan | Waktu dalam Bulan (M) |
| 20 | 3 * PM^(1/3) | | 22 | 5.39 |
| | | | 26 | 4.73 |
| 28 | | | 22 | 6.03 |
| | | | 26 | 5.29 |

# Economic Feasibility

**Cost-Benefit Analysis**

| Tahun | 2019 | 2020 | 2021 | 2022 |
|---|---|---|---|---|
| Peningkatan Pendapatan Penjualan Lagu | | 400,000,000 | 400,000,000 | 400,000,000 |
| Pengurangan Biaya Sewa Ruangan | | 120,000,000 | 120,000,000 | 120,000,000 |
| Pengurangan Biaya Komunikasi | | 6,000,000 | 6,000,000 | 6,000,000 |
| **Total Benefits** | **0** | **526,000,000** | **526,000,000** | **526,000,000** |
| **PV of Benefits** | **0** | **468,138,127** | **441,639,743** | **416,641,267** |
| **PV of All Benefits** | **0** | **468,138,127** | **909,777,870** | **884,779,394** |
| Honor Tim (Analysis, Design and Implementation) | 250,000,000 | 120,000,000 | 120,000,000 | 120,000,000 |
| | | | | |
| **Total Development Costs** | **250,000,000** | **120,000,000** | **120,000,000** | **120,000,000** |
| Honor Pengelola Web | 72,000,000 | 72,000,000 | 72,000,000 | 72,000,000 |
| Biaya Lisensi Software | 10,000,000 | 10,000,000 | 10,000,000 | 10,000,000 |
| Hardware upgrades | 50,000,000 | 50,000,000 | 50,000,000 | 50,000,000 |
| Biaya Komunikasi | 1,000,000 | 1,000,000 | 1,000,000 | 1,000,000 |
| Biaya Marketing | 50,000,000 | 50,000,000 | 50,000,000 | 50,000,000 |
| **Total Operational Costs** | **183,000,000** | **183,000,000** | **183,000,000** | **183,000,000** |
| **Total Costs** | **433,000,000** | **303,000,000** | **303,000,000** | **303,000,000** |
| **PV of Costs** | **408,490,566** | **269,668,921** | **153,650,329** | **144,953,140** |
| **PV of all Costs** | **408,490,566** | **678,159,487** | **831,809,816** | **976,762,957** |
| **Total Project Costs Less Benefits** | **-433,000,000** | **223,000,000** | **223,000,000** | **223,000,000** |
| **Yearly NPV** | **-408,490,566** | **198,469,206** | **187,235,100** | **176,636,887** |
| **Cumulative NPV** | **-408,490,566** | **-210,021,360** | **-22,786,260** | **153,850,627** |
| **Return on Investment (ROI)** | **-100.00%** | **-0.309693168** | **-0.027393593** | **0.15751071** |
| **Break-even Point (BEP)** | | | | **3.129000574** |

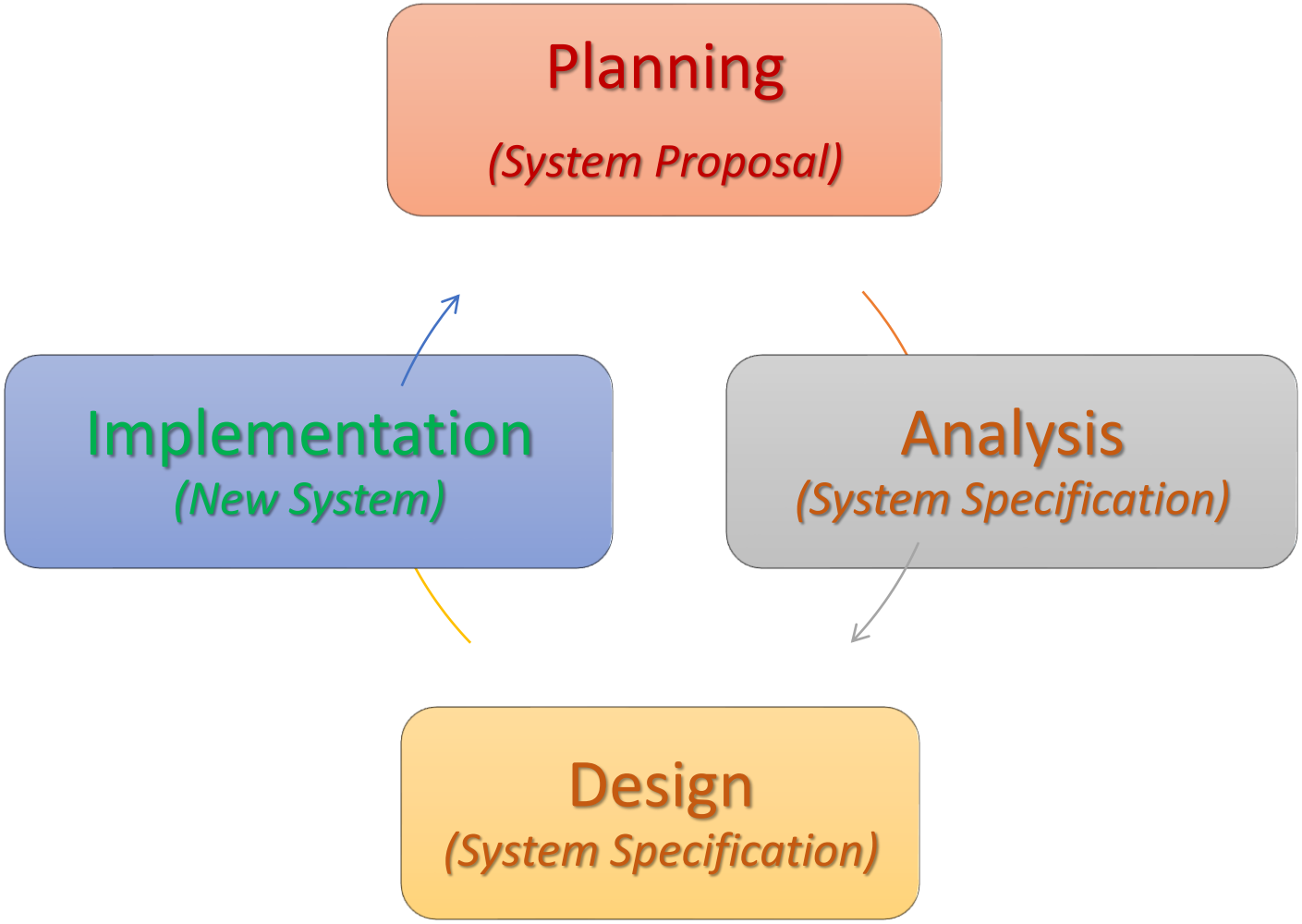
# *Organizational Feasibility*

| musicpedia | | | |
|---|---|---|---|
| **Date** | **26 Oktober 2018** | | |
| **Anggota Tim** | | | |
| User/Product Owner | Wahyu Utomo | | |
| Project Manager | Haris Dermawan | | |
| System Analyst | Risa Dhani Horasman Purba | | |
| Business Analyst | Mulyana | | |
| Programmer | Achmad Fatkarrofiqi | | |
| Tester | Januar Sapareza | | |
| **Apakah aplikasi ini mendukung visi dan misi organisasi?** | | | |
| Ya | | | |
| **Apakah aplikasi ini sesuai dengan tugas, fungsi dan KPI unit kerja anda?** | | | |
| Ya | | | |
| **Apakah aplikasi ini selaras dengan proses bisnis unit kerja anda?** | | | |
| Ya | | | |
| Secara analisis kelayakan organisasi, apakah aplikasi layak dikembangkan sesuai kriteria di atas? | | ☑ Layak | ☐ Tidak Layak |

# 2. Systems Analysis

# 2.1 Requirement Gathering

# Siklus Pengembangan Software




*(Tilley, 2012)* *(Dennis, 2016)* *(Valacich, 2017)*

# Boehm's First Law

Errors are most frequent during the requirements and design activities, and are the more expensive the later they are removed

*(Endres, 2003)* *[L2]*

# What is a Requirements

- Business Requirements
  - Statement of what the system must do
  - Focus on what the system must do, not how to do it
- There are 2 kinds of requirements
  1. Functional Requirements
  2. Nonfunctional Requirements

**D. Nonfunctional Requirements**

**1. Operational Requirements**
1.1. The system will operate in Windows and Macintosh environments
1.2. The system will be able to read and write Word documents, RTF, and HTML
1.3. The system will be able to import Gif, Jpeg, and BMP graphics files

**2. Performance Requirements**
2.1. Response times must be less than 7 seconds
2.2. The Inventory database must be updated in real time

**3. Security Requirements**
3.1. No special security requirements are anticipated

**4. Cultural and Political Requirements**
4.1. No special cultural and political requirements are anticipated

**C. Functional Requirements**

**1. Printing**
1.1. The user can select which pages to print
1.2. The user can view a preview of the pages before printing
1.3. The user can change the margins, paper size (e.g., letter, A4) and orientation on the page

**2. Spell Checking**
2.1. The user can check for spelling mistakes; the system can operate in one of two modes as selected by the users
2.1.1. Mode 1 (Manual): The user will activate the spell checker and it will move the user to the next misspelled word
2.1.2. Mode 2 (Automatic): As the user types, the spell checker will flag misspelled words so the user immediately see the misspelling
2.2. The user can add words to the dictionary
2.3. The user can mark words as not misspelled but not add them to the dictionary

# Contoh Template System Specification (SRS, BRD, FSD)

1. System Planning
   - 1.1 Project Scope
   - 1.2 Project Schedule
   - 1.3 Project Team
2. System Design
   - 2.1 Functional Requirements
     - 2.1.1 Actor
     - 2.1.2 Use Case Diagram
     - 2.1.3 Activity Diagram (BPMN)
     - 2.1.4 Sequence Diagram
     - 2.1.5 Class Diagram
     - 2.1.6 Data Model
     - 2.1.7 User Interface Design
     - 2.1.8 Deployment Diagram
     - 2.1.9 Relational Matrices
       - 2.1.9.1 Actor – Activity Diagram
       - 2.1.9.2 Actor – Sequence Diagram
   - 2.2 Nonfunctional Requirements
     - 2.2.1 Operational
     - 2.2.2 Performance
     - 2.2.3 Security
     - 2.2.4 Hardware
     - 2.2.5 Development Platform
     - 2.2.6 Deadline
3. System Implementation
   - 3.1 Testing Strategy
   - 3.2 Installation Strategy
   - 3.3 Change Management Strategy

# 1. Functional Requirement

- Kebutuhan tentang fungsi software secara menyeluruh
- Pemodelan dengan UML, ataupun penjelasan fitur-fitur dalam bentuk problem statements, adalah termasuk dalam Functional Requirement
  - Diagrams:
    - Use Case Diagrams
    - Activity Diagrams
  - Problem Statements:
    - Must search for inventory
    - Must perform these calculations
    - Must produce a specific report

# 2. Nonfunctional Requirements

1. Operational – Physical/technical environment
2. Performance – Speed and reliability
3. Security – Who can use the system
4. Cultural & Political – Company policies, legal issues

# *Nonfunctional Requirements*

## Jaringan Dokumentasi dan Informasi Hukum (JDIH)

**Date: 23 Oktober 2018**

**Operational Requirements**

1. Sistem dapat digunakan oleh semua perangkat lunak yang lainnya.
2. Sistem layanan harus memiliki tingkat ketersediaan 99%.
3. System harus go live sebelum tanggal 25 Januari 2018.

**Performance Requirements**

1. Waktu respon harus tidak lebih dari 2 detik.
2. Mampu diakses oleh lebih dari 50 pengguna secara bersamaan.
3. Server harus bersih dari virus.

**Security Requirements**

1. Akses ke dalam system tanpa otorisasi tidak dimungkinkan.
2. Data hanya dapat diubah oleh administrator sistem.
3. Seluruh data harus di-backup setiap 24jam serta hasil backup-nya disimpan di lokasi yang berbeda dengan sistem.
4. Seluruh komunikasi antara client-server harus dipastikan keamanannya.

**Deadline Requirements**

1. Pengembangan aplikasi tidak boleh melewati jangka waktu yang sudah ditentukan.

# Metode Requirement Gathering

1. Document Analysis
2. Interviews
3. Joint Application Design (JAD)
4. Questionnaires
5. Observation

# 1. Document Analysis

- Provides clues about the "formal" existing As-Is system
- Typical documents
  - Forms
  - Reports
  - Policy manuals
- Look for user additions to forms
- Look for unused form elements
- Do document analysis before interviews

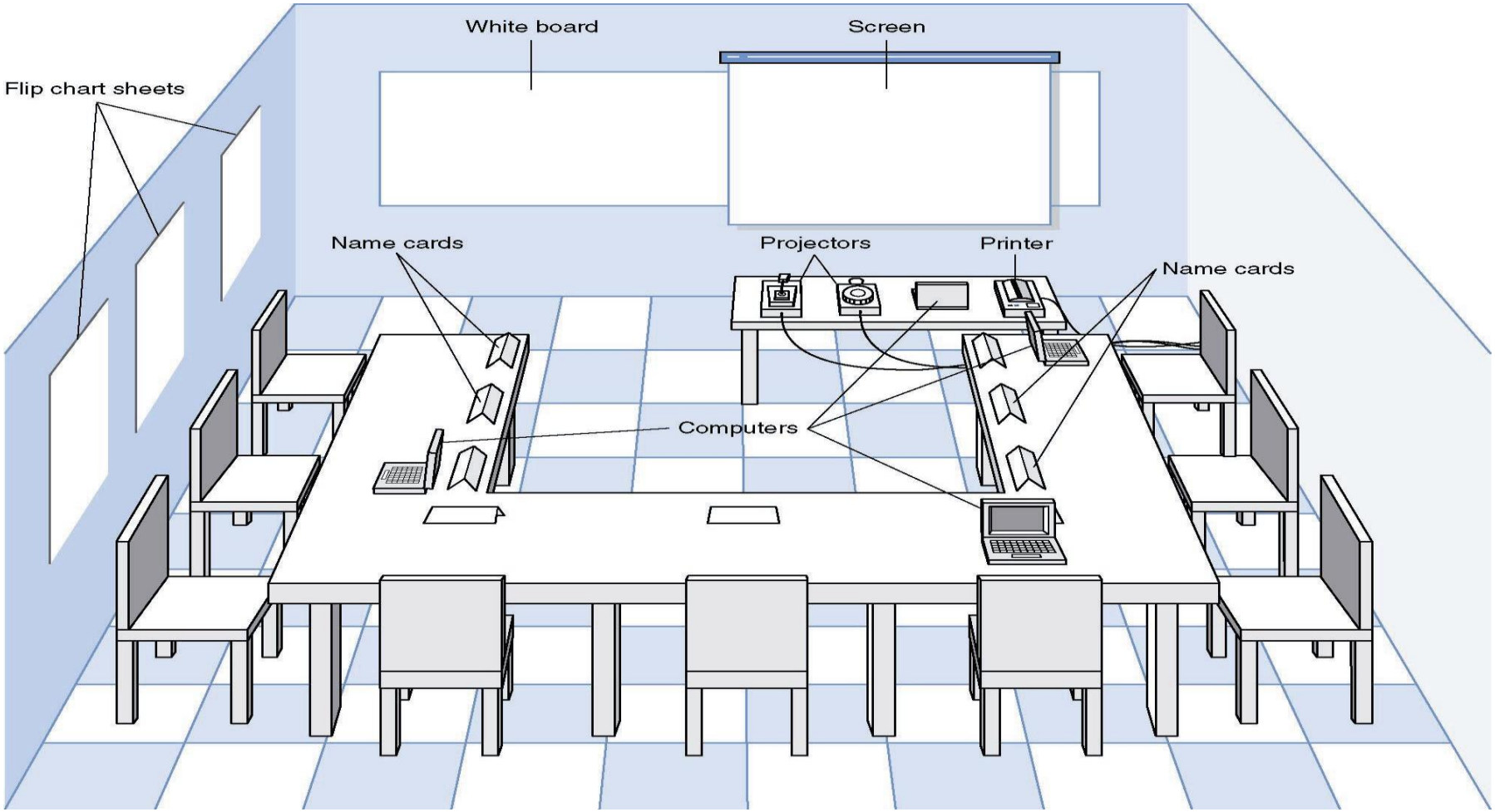
# 2. Interviews

- Most commonly used technique and very natural
- If you need to know something, you ask someone
- Five basic steps:
  1. Selecting interviewees
  2. Designing interview questions
  3. Preparing for the interview
  4. Conducting the interview
  5. Post-interview follow-up

# 3. Joint Application Design (JAD)

- Allows project managers, users, and developers to work together
- May reduce scope creep by 50%
- Avoids requirements being too specific or too vague
- Include 10 to 20 users
- Tend to last 5 to 10 days over a three-week period

# JAD Meeting Room



# 4. Questionnaire

- Selecting participants
  - Using samples of the population
- Designing the questionnaire
  - More important than interview questions
  - Prioritize questions to grab attention
  - Distinguish between:
    1. Fact-oriented questions (specific answers)
    2. Opinion questions (agree – disagree scale)
- Administering the questionnaire
  - Explain its importance & how it will be used
  - Give expected response date
  - Follow up on late returns and have supervisors follow up
  - Promise to report results
- Questionnaire follow-up
  - Send results to participants

# 5. Observation

- Users/managers often don't remember everything they do
- Validates info gathered in other ways
- Behaviors change when people are watched
- Keep low profile, don't change the process
- Careful not to ignore periodic activities
  - Weekly … Monthly … Annual

# Karakteristik Metode Requirement Gathering

| | Interviews | JAD | Questionnaires | Document Analysis | Observation |
|---|---|---|---|---|---|
| Type of Information | As-Is Improve. To-Be | As-Is Improve. To-Be | As-Is Improve. | As-Is | As-Is |
| Depth of Information | High | High | Medium | Low | Low |
| Breadth of Information | Low | Medium | High | High | Low |
| Integration of Info. | Low | High | Low | Low | Low |
| User Involvement | Medium | High | Low | Low | Low |
| Cost | Medium | Low-Medium | Low | Low | Low-Medium |

# Business Process Analysis Strategies

1. BPA (Business Process Automation)
   - Makes almost no changes to business processes, just makes them more efficient

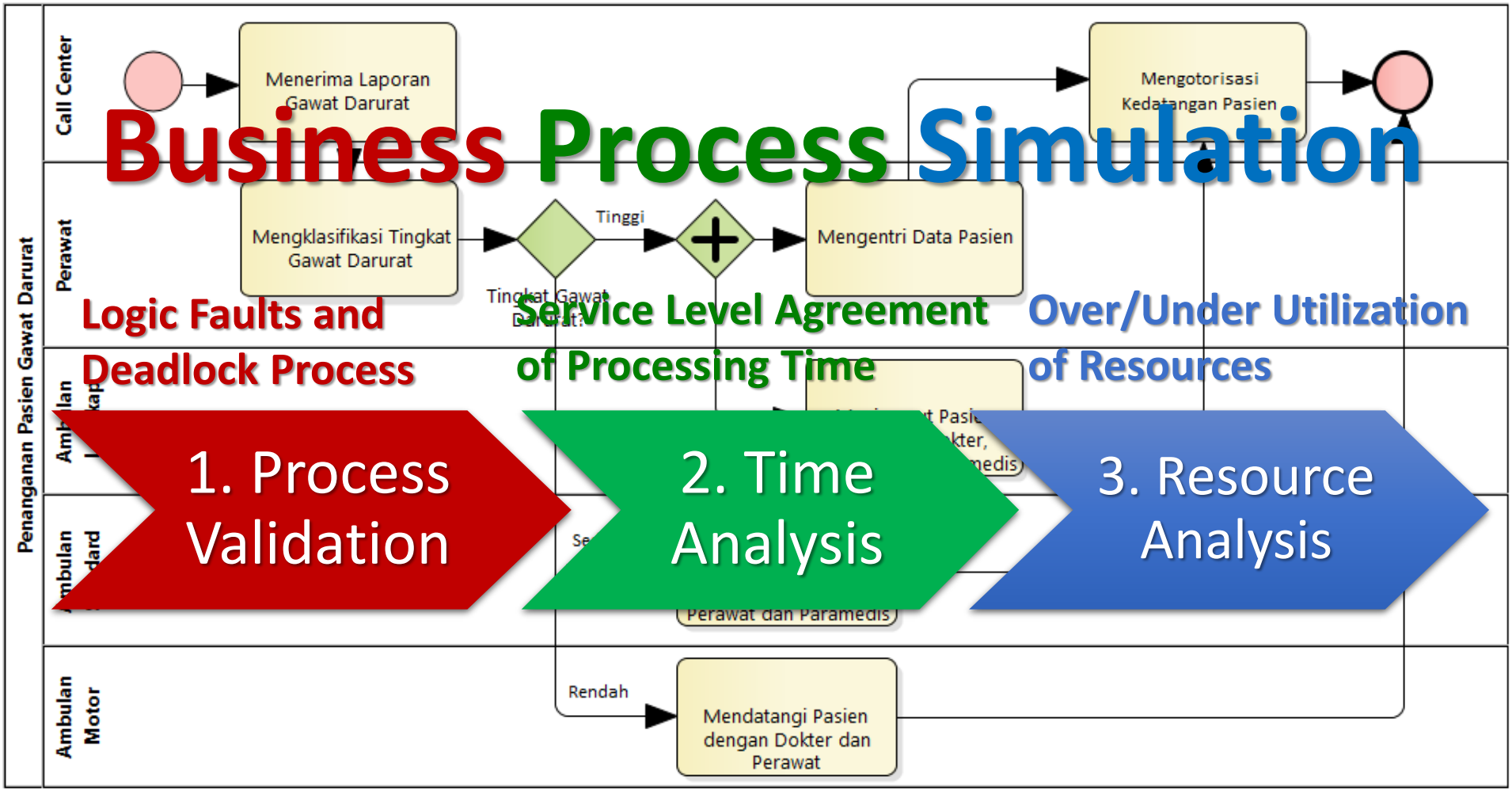
2. BPI (Business Process Improvement)
   - Change what the users do, not just how efficiently they do it

3. BPR (Business Process Reengineering)
   - Throw away everything, start with a blank page

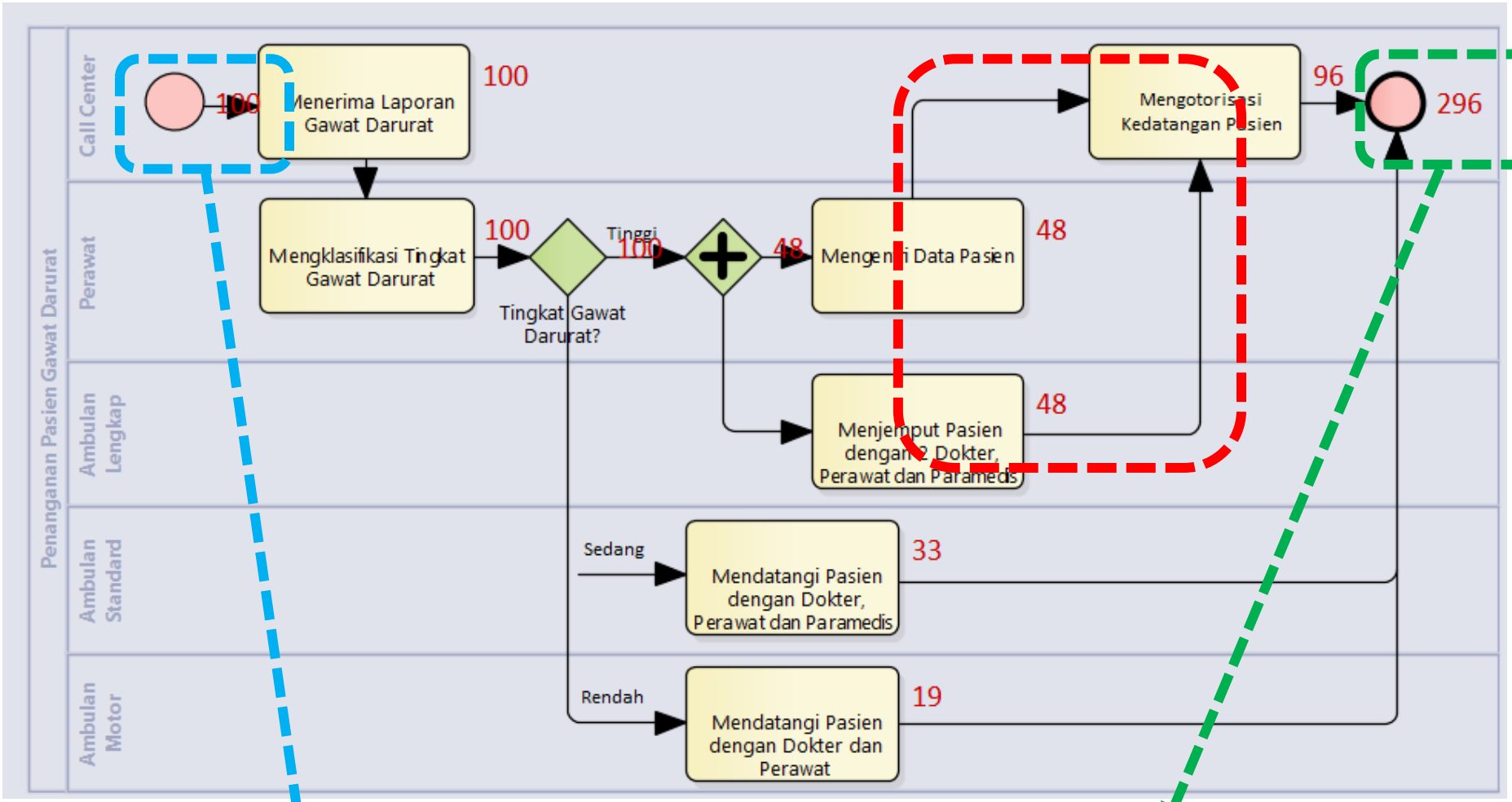
# Business Process Analysis Strategies Comparison

| | Business Process Automation | Business Process Improvement | Business Process Reeingineering |
|---|---|---|---|
| Potential Business Value | Low-Moderate | Moderate | High |
| Project Cost | Low | Low-Moderate | High |
| Breadth of Analysis | Narrow | Narrow-Moderate | Very Broad |
| Risk | Low | Low-Moderate | Very High |

# Business Process Simulation: Penanganan Pasien Gawat Darurat



# Validasi Proses Bisnis

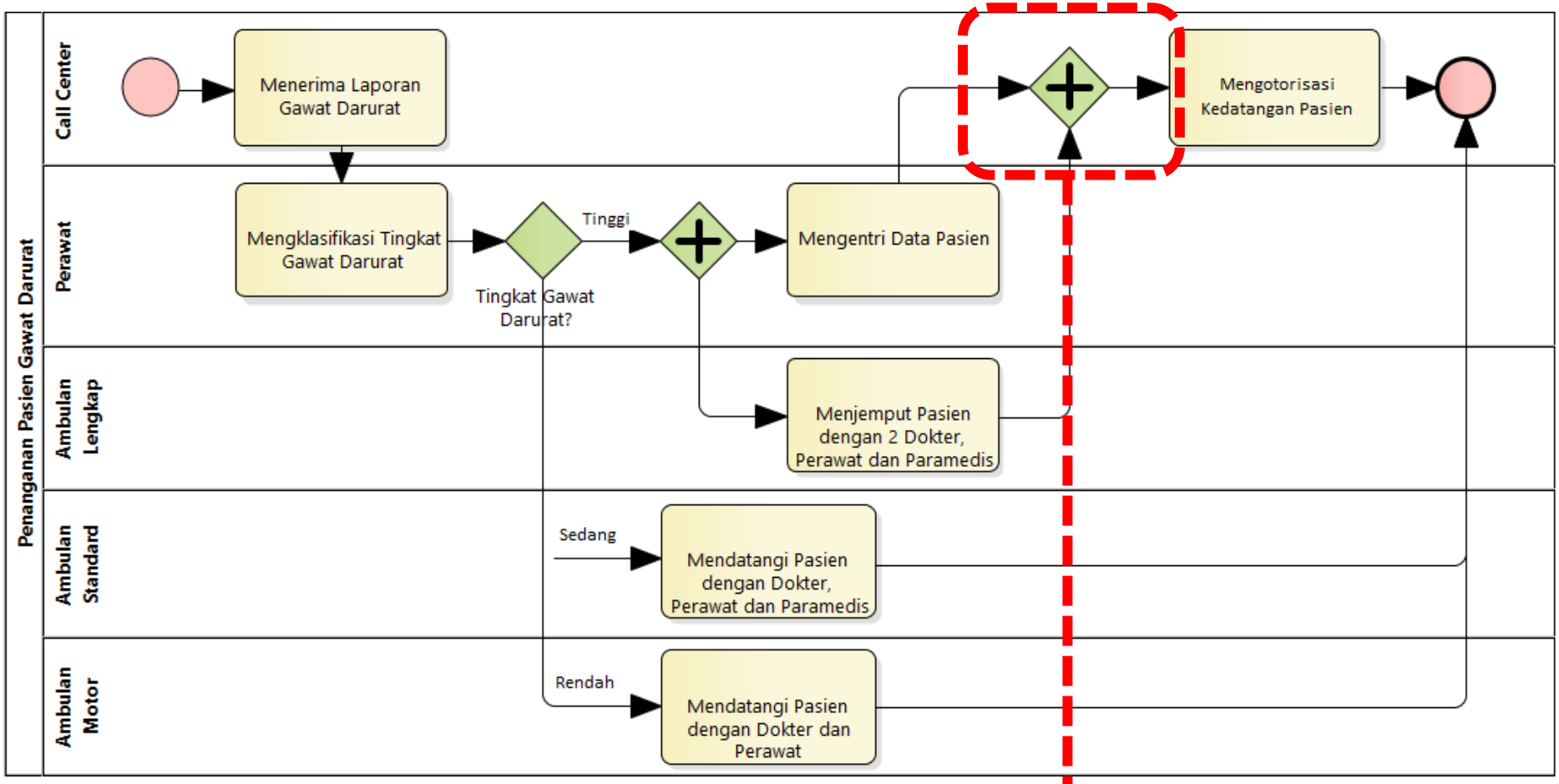



**Menangani 100 Orang Pasien**

**Seperti Menangani 296 Orang Pasien**

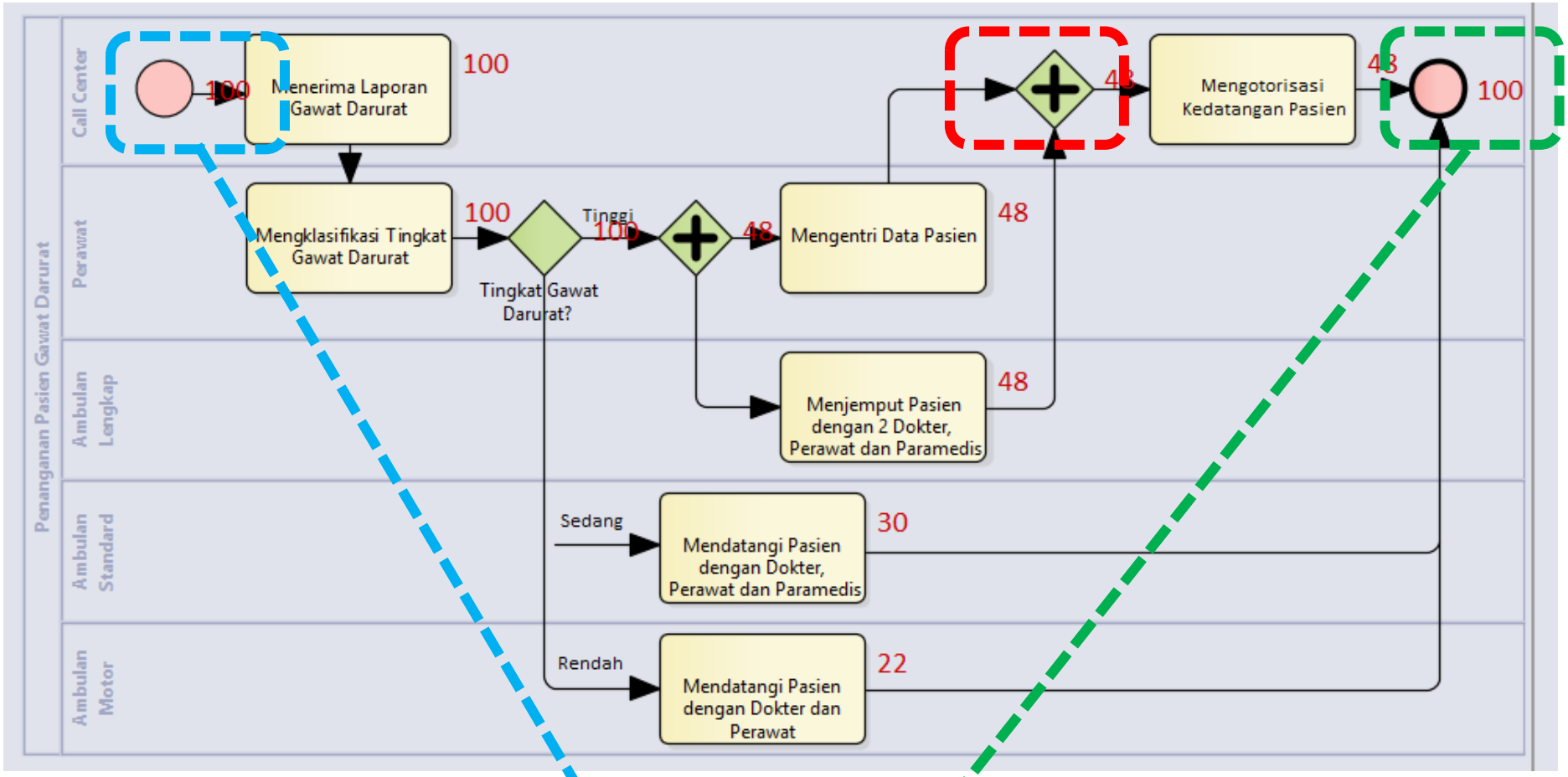
**PROSES BISNIS INI TIDAK EFISIEN!**

# Validasi Proses Bisnis




**Perlu Sinkronisasi Pekerjaan,**
**Sebelum Otorisasi Kedatangan Dilakukan**

# Validasi Proses Bisnis



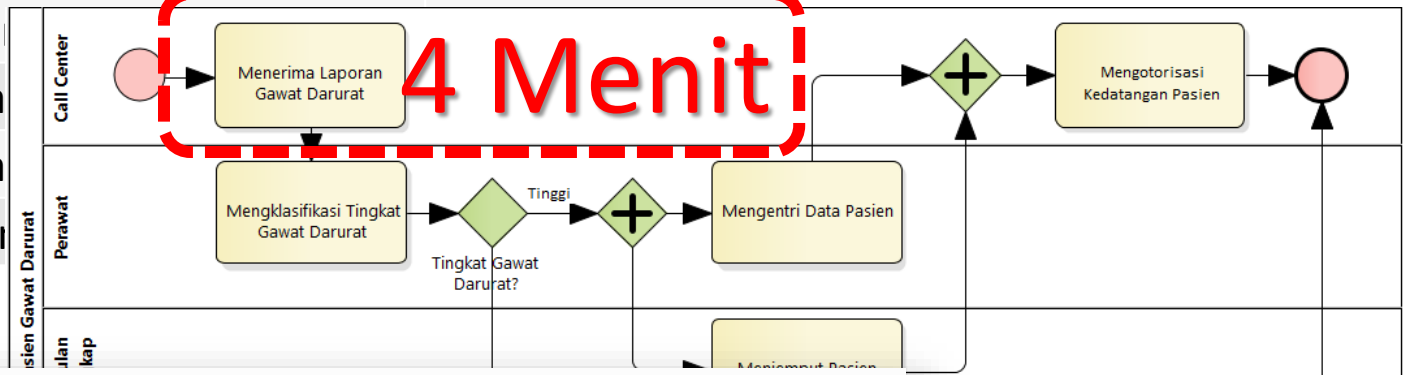

**Proses Bisnis Terbukti Valid dan**

**100 Pasien Tertangani dengan Efisien dan Efektif!**

# Analisis Waktu

| Aktifitas | Waktu (Menit) |
|---|---|
| Menerima Laporan Gawat Darurat | 4 |
| Mengklasifikasi Tingkat Gawat Darurat | 5 |
| Mengentri Data Pasien | 11 |
| Menjemput Pasien denga | |
| Mendatangi Pasien denga | |
| Mendatangi Pasien denga | |
| Mengotorisasi Kedatanga | |

4 Menit

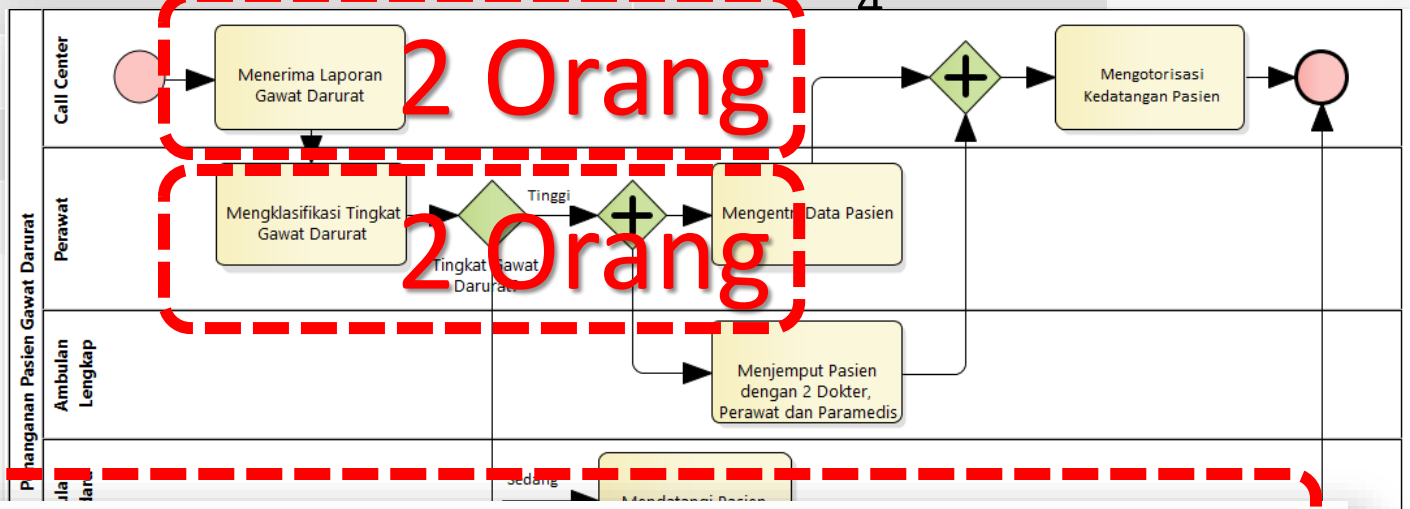



| | | |
|---|---|---|
| Penanganan Pasien Gawat Darurat | Average Time | 249.63 |
| Penanganan Pasien Gawat Darurat | Mean Of Processing Time | 17.73 |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | Number Of Processes Started | 100 |
| Penanganan Pasien Gawat Darurat | Standard Deviation Time | 131.97 |

Perlu Waktu 17.73 Menit untuk Menangani 1 Pasien

# Analisis Sumber Daya Manusia

| Sumber Daya | Kuantitas |
|---|---|
| Call center | 2 |
| Perawat | 2 |
| Ambulan Lengkap | 4 |
| Ambulan Standar | |
| Ambulan Motor | |
| Resepsionis | |



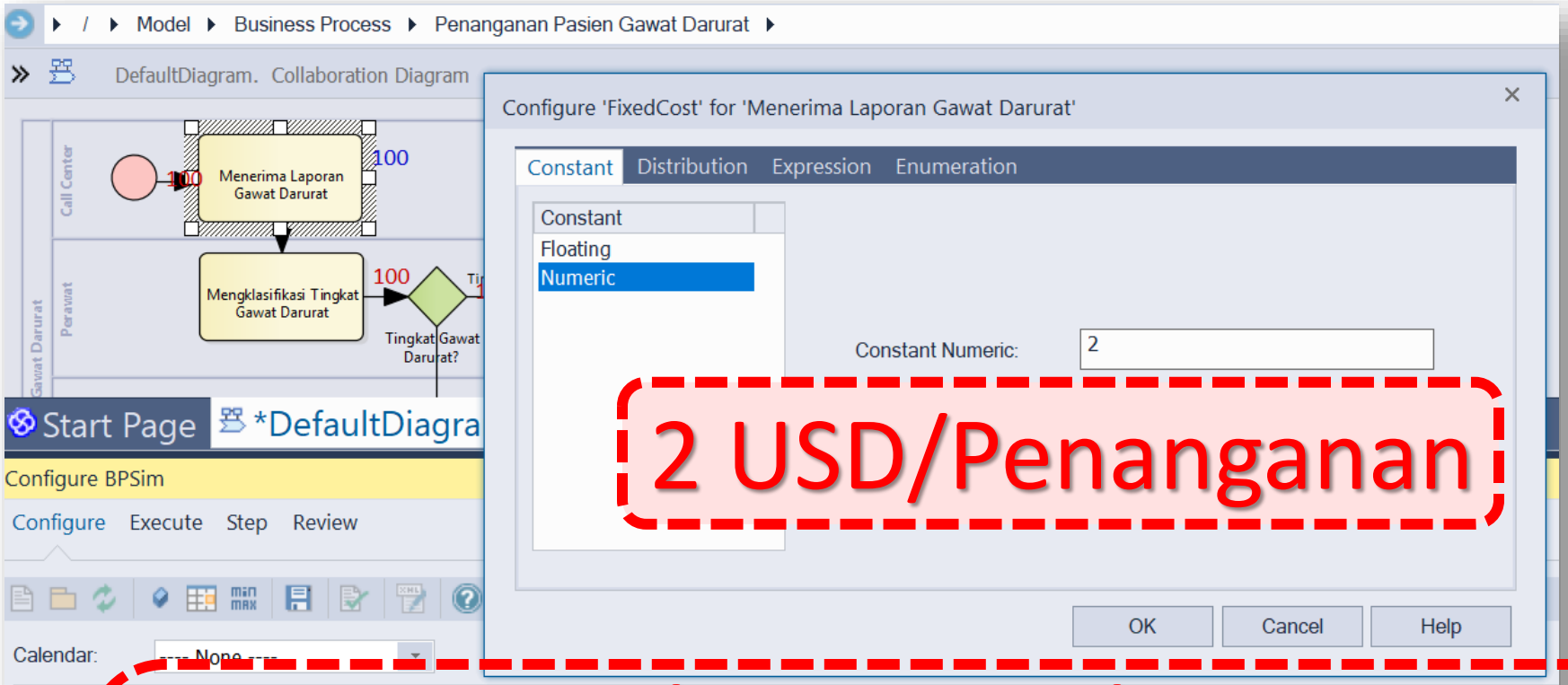

Overutilization

| | | |
|---|---|---|
| Perawat | Degree Of Utilisation | 99% |
| Paramedis | Degree Of Utilisation | 98.34% |

| | | |
|---|---|---|
| Call Center | Degree Of Utilisation | 65.56% |
| Dokter | Degree Of Utilisation | 50.28% |
| Supir | Degree Of Utilisation | 33.22% |
| Ambulan Standard | Degree Of Utilisation | 33.22% |
| Ambulan Motor | Degree Of Utilisation | 17.05% |

Underutilization

# Analisis Biaya




2 USD/Penanganan

Total Biaya Perhari

| | | |
|---|---|---|
| Menerima Laporan Gawat Darurat | Total Completion Cost | 200 |
| Mengentri Data Pasien | Total Completion Cost | 48 |
| Mengklasifikasi Tingkat Gawat Darurat | Total Completion Cost | 100 |
| Mengotorisasi Kedatangan Pasien | Total Completion Cost | 48 |

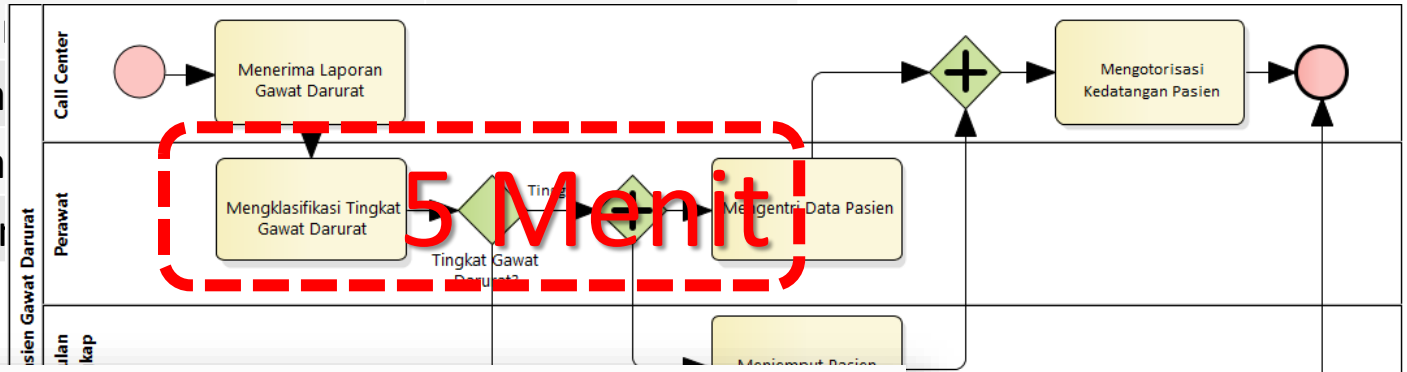
# Business Process Automation

- Dikembangkan Sistem Cerdas yang membantu perawat dengan Mengautomasi Klasifikasi Tingkat Gawat Darurat

- Pekerjaan "Mengklasifikasi Tingkat Gawat Darurat" yang sebelumnya perlu waktu 5 menit
  - Dengan sistem cerdas menjadi hanya 1 menit

# Business Process Simulation

| Aktifitas | Waktu (Menit) |
|---|---|
| Menerima Laporan Gawat Darurat | 4 |
| Mengklasifikasi Tingkat Gawat Darurat | 5 |
| Mengentri Data Pasien | 11 |
| Menjemput Pasien denga | |
| Mendatangi Pasien denga | |
| Mendatangi Pasien denga | |
| Mengotorisasi Kedatanga | |




| | | |
|---|---|---|
| Penanganan Pasien Gawat Darurat | Average Time | 249.63 |
| Penanganan Pasien Gawat Darurat | Mean Of Processing Time | 17.73 |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | Number Of Processes Started | 100 |
| Penanganan Pasien Gawat Darurat | Standard Deviation Time | 131.97 |

**Perlu Waktu 17.73 Menit untuk Menangani 1 Pasien**

# Business Process Automation

| Aktifitas | Waktu (Menit) |
|---|---|
| Menerima Laporan Gawat Darurat | 4 |
| Mengklasifikasi Tingkat Gawat Darurat | 1 |
| Mengentri Data Pasien | 11 |
| Menjemput Pasien denga | |
| Mendatangi Pasien denga | |
| Mendatangi Pasien denga | |
| Mengotorisasi Kedatanga | |


| | | |
|---|---|---|
| Penanganan Pasien Gawat Darurat | Average Time | 249.63 |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | | |
| Penanganan Pasien Gawat Darurat | Number Of Processes Completed | 52 |
| Penanganan Pasien Gawat Darurat | Number Of Processes Started | 100 |
| Penanganan Pasien Gawat Darurat | Standard Deviation Time | 131.97 |

**Hanya Perlu Waktu 8 Menit untuk Menangani 1 Pasien**

# Tantangan di Requirement Gathering

1. The "Yes, But" syndrome
   - Stems from human nature and the users' inability to experience the software as they might a physical device
2. The “Undiscovered Ruins”
   - Searching for requirements is like searching for "Undiscovered Ruins“
   - The more you find, the more you know remain
3. The "User and the Developer" syndrome
   - Reflects the profound differences between these two, making communication difficult

# The "Yes, But" Syndrome

For whatever reason, we always see two immediate, distinct, and separate reactions when the users see the system implementation for the first time

1. "Wow, this is so cool; we can really use this, what a neat job" and so on.
2. "Yes, but, hmmmm, now that I see it, what about this ... ? Wouldn't it be nice if . . . ? Whatever happened to . . . ?“

# The "Undiscovered Ruins" Syndrome

- In many ways, the search for requirements is like a search for undiscovered ruins
  - The more you find, the more you know remain
  - You never really feel that you have found them all, and perhaps you never will.

- Indeed, software development teams always struggle to determine when they are done with requirements elicitation. When have they found
  - all the requirements
  - or at least enough requirements?

# The "User and the Developer" Syndrome

- Communication gap between the user and the developer

- Users and developers are
  - typically from different worlds,
  - may even speak different languages,
  - have different backgrounds,
  - have different motivations, and
  - have different objectives

# The "User and the Developer" Syndrome

Reasons for this problem and some suggested solutions

| Problem | Solution |
|---|---|
| Users do not know what they want, or they know what they want but cannot articulate it. | Recognize and appreciate the user as domain expert; try alternative communication and elicitation techniques. |
| Users think they know what they want until developers give them what they said they wanted. | Provide alternative elicitation techniques earlier: storyboarding, role playing, throwaway prototypes, and so on. |
| Analysts think they understand user problems better than users do. | Put the analyst in the user's place. Try role playing for an hour or a day. |
| Everybody believes everybody else is politically motivated. | Yes, its part of human nature, so let's get on with the program. |

# Evolusi Paradigma Analysis dan Design

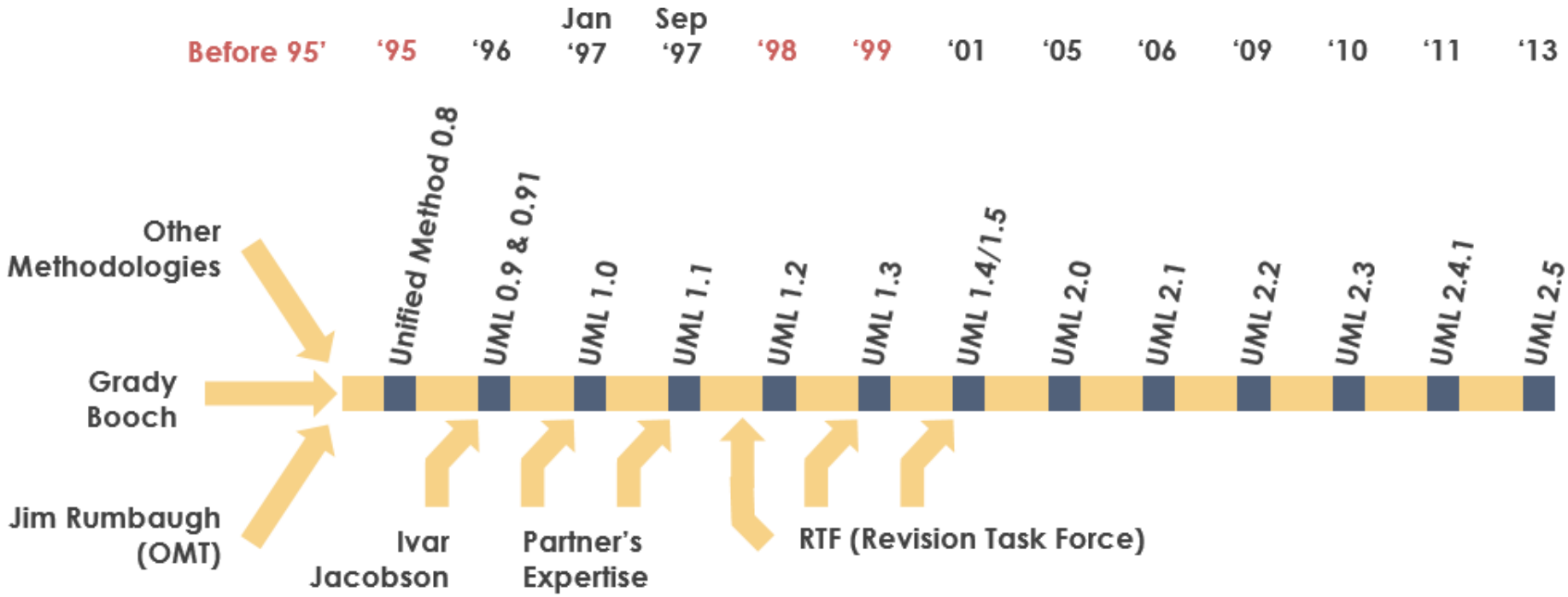
| | Paradigm | Diagrams |
|---|---|---|
| 1 | Process-oriented Paradigm | Flowchart |
| 2 | Data-oriented Paradigm | DFD |
| 3 | Object-oriented Paradigm (data + process) | UML |

# What is the UML?

- UML: Unified Modeling Language
- UML can be used for modeling all processes in the development life cycle and across different implementation technologies (technology and language independent)
- UML is the standard language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system
- UML is a communication tool – for the team, and other stakeholders

# Sejarah UML

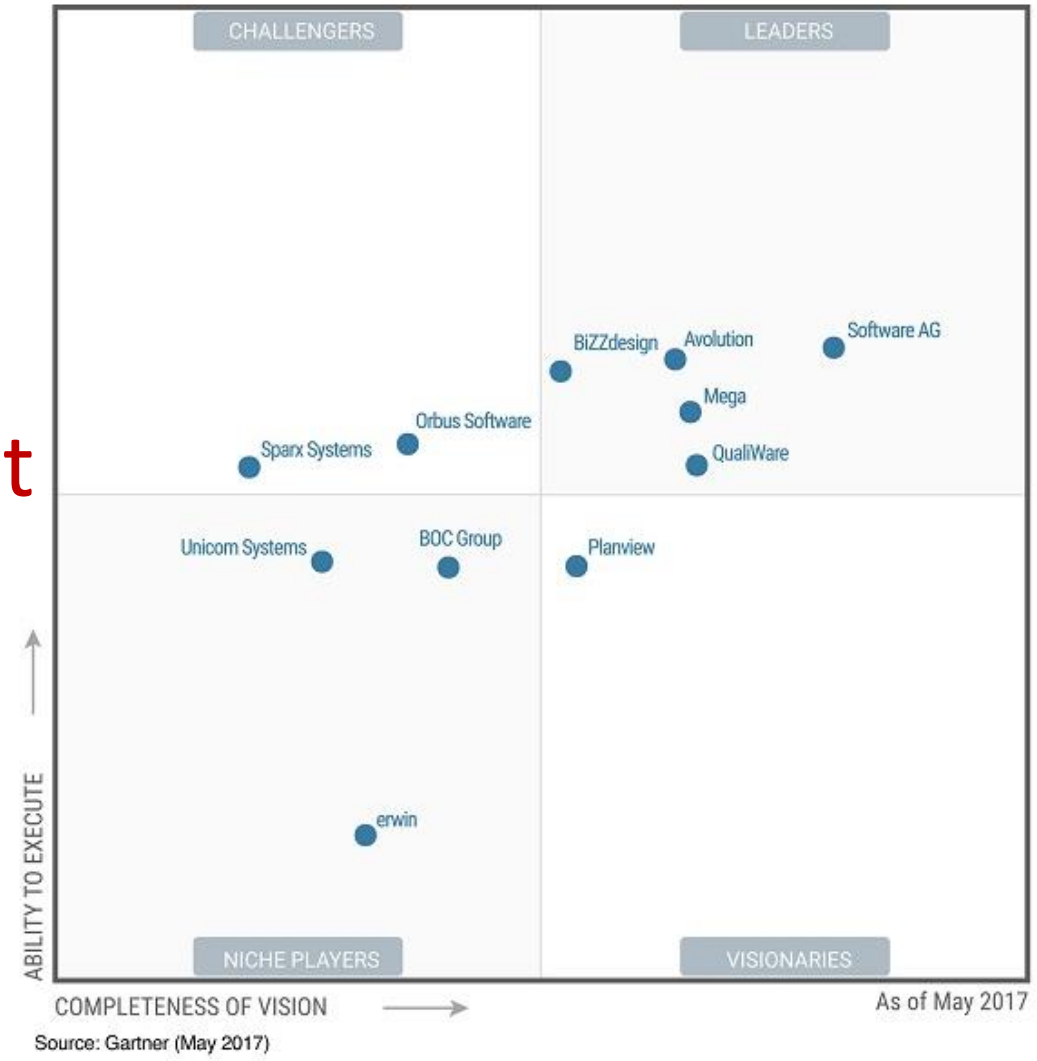
Booch, Jacobson, Rumbaugh




Before 95' - Fragmentation ▶ 95' - Unification ▶ 98' - Standardization ▶ 99' - Industrialization

# State of the Art UML Tools

- Rational Rose
- Visual Paradigm
- Sparx Systems Enterprise Architect
- Microsoft Visio
- Star UML



# UML 2.0 Diagrams

| Diagram Name | Used to | Primary Phase |
|---|---|---|
| **Structure Diagrams** | | |
| Class | Illustrate the relationships between classes modeled in the system. | Analysis, Design |
| Object | Illustrate the relationships between objects modeled in the system. Used when actual instances of the classes will better communicate the model. | Analysis, Design |
| Package | Group other UML elements together to form higher level constructs. | Analysis, Design, Implementation |
| Deployment | Show the physical architecture of the system. Can also be used to show software components being deployed onto the physical architecture. | Physical Design, Implementation |
| Component | illustrate the physical relationships among the software components. | Physical Design, Implementation |
| Composite Structure | Illustrate the internal structure of a class, i.e., the relationships among the parts of a class. | Analysis, Design |
| **Behavioral Diagrams** | | |
| Activity | Illustrate business workflows independent of classes, the flow of activities in a use case, or detailed design of a method. | Analysis, Design |
| Sequence | Model the behavior of objects within a use case. Focuses on the time-based ordering of an activity. | Analysis, Design |
| Communication | Model the behavior of objects within a use case. Focuses on the communication among a set of collaborating objects of an activity. | Analysis, Design |
| Interaction Overview | Illustrate an overview of the flow of control of a process. | Analysis, Design |
| Timing | Illustrate the interaction that takes place among a set of objects and the state changes in which they go through along a time axis. | Analysis, Design |
| Behavioral State Machine | Examine the behavior of one class. | Analysis, Design |
| Protocol State Machine | Illustrates the dependencies among the different interfaces of a class. | Analysis, Design |
| Use-Case | Capture business requirements for the system and to illustrate the interaction between the system and its environment. | Analysis |

# UML Problems

1. UML is modeling notation, it is not a development process or a methodology
   - UML driven development process?

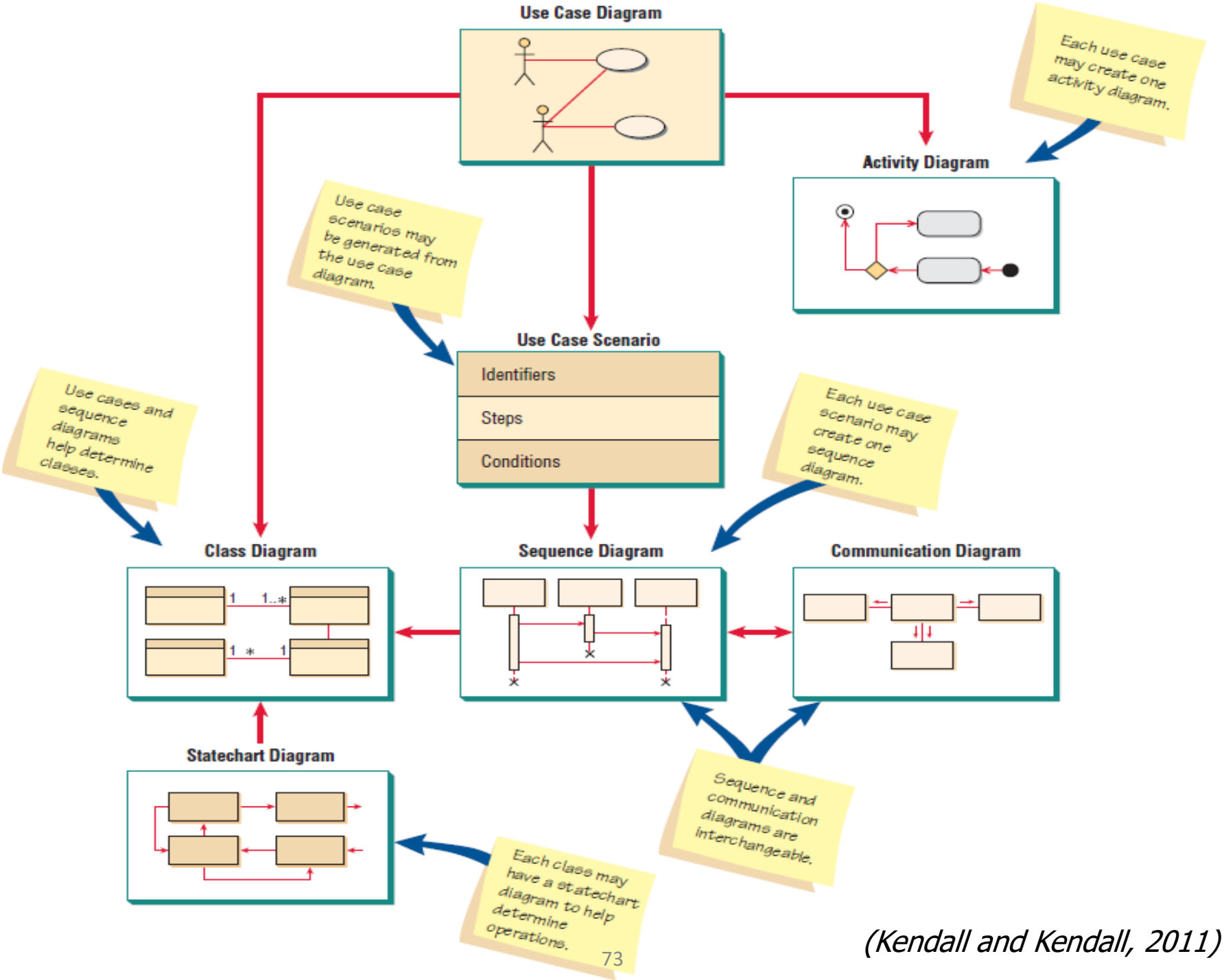
2. UML is too complex, difficult to understand quickly
   - Which UML diagrams should we use?

# UML based Software Analysis and Design (*Sparx Systems EA*)

1. Display the boundary of a system and its major functions using use cases and actors
2. Model the organization's business process with activity diagram
3. Illustrate use case realizations with sequence diagrams
4. Represent a static structure of a system using class diagrams
5. Reveal the physical implementation architecture with deployment diagrams

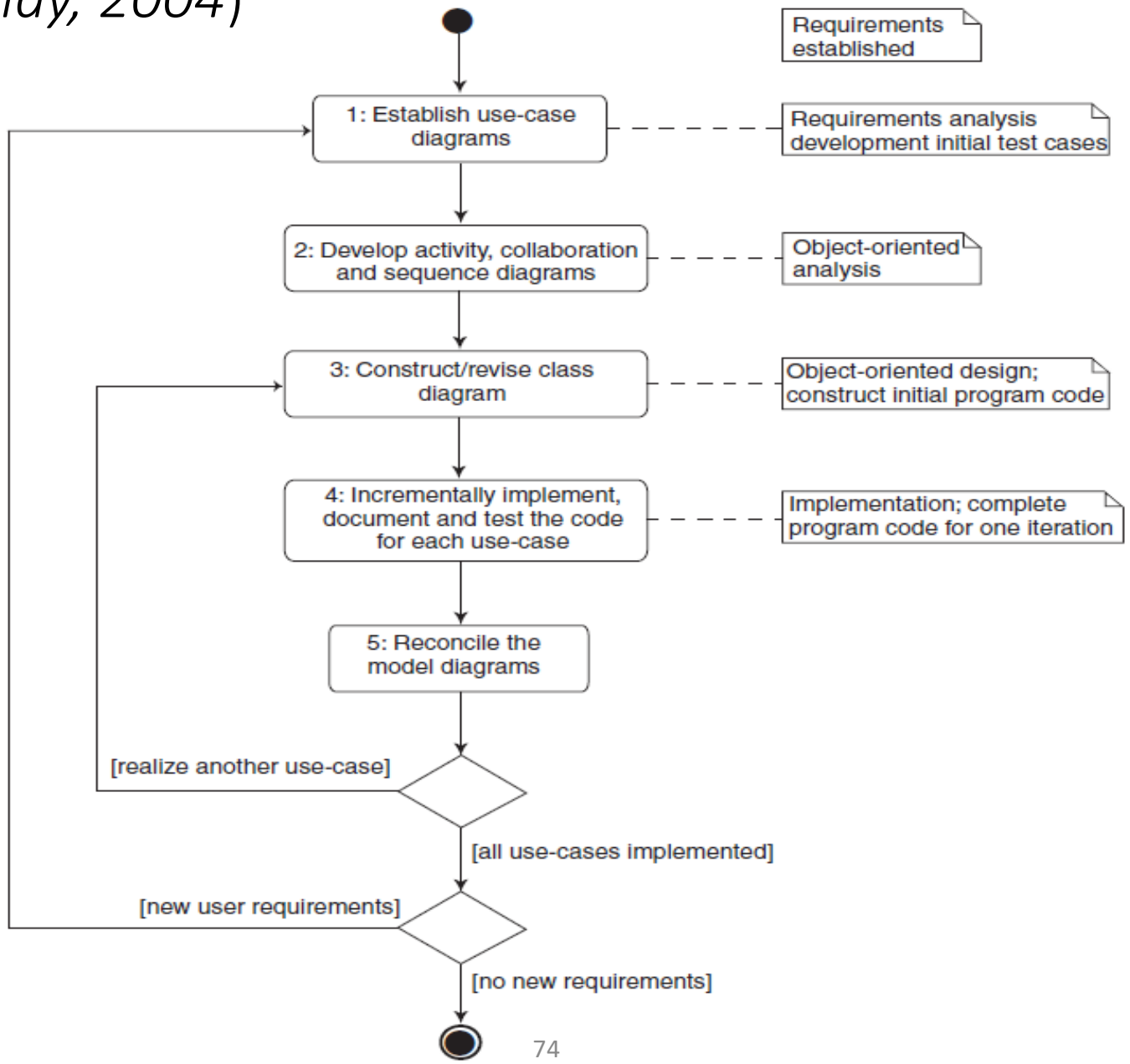
# UML based Software Analysis and Design (*Kendal, 2011*)

1. A use case diagram, describing how the system is used. Analysts start with a use case diagram
2. An activity diagram, illustrating the overall flow of activities. Each use case may create one activity diagram
3. Sequence diagrams, showing the sequence of activities and class relationships. Each use case may create one or more sequence diagrams
4. Class diagrams, showing the classes and relationships. Sequence diagrams are used to determine classes
5. Statechart diagrams, showing the state transitions. Each class may create a statechart diagram, which is useful for determining class methods



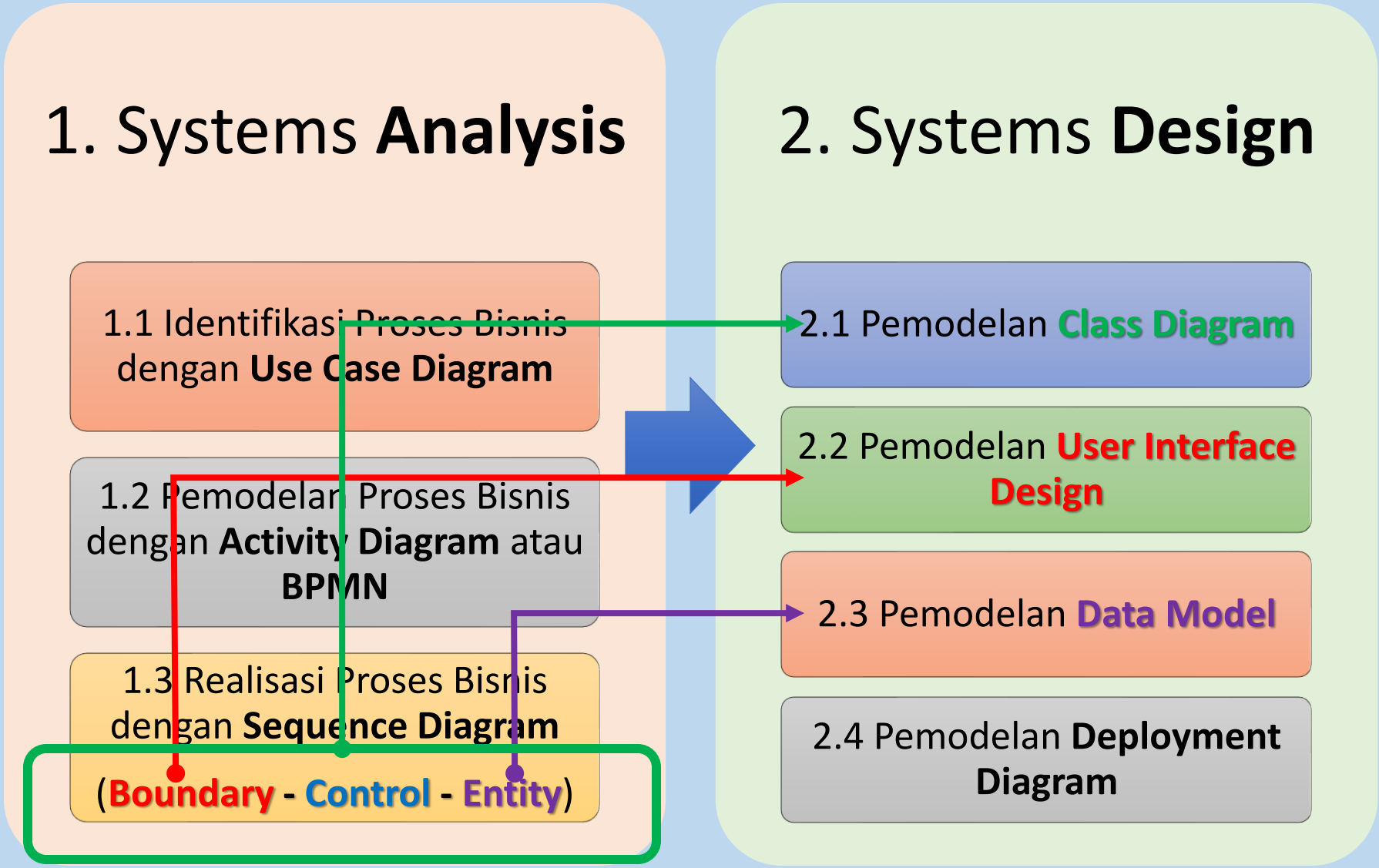

*(Kendall and Kendall, 2011)*

# UML based Software Analysis and Design (*Barclay, 2004*)

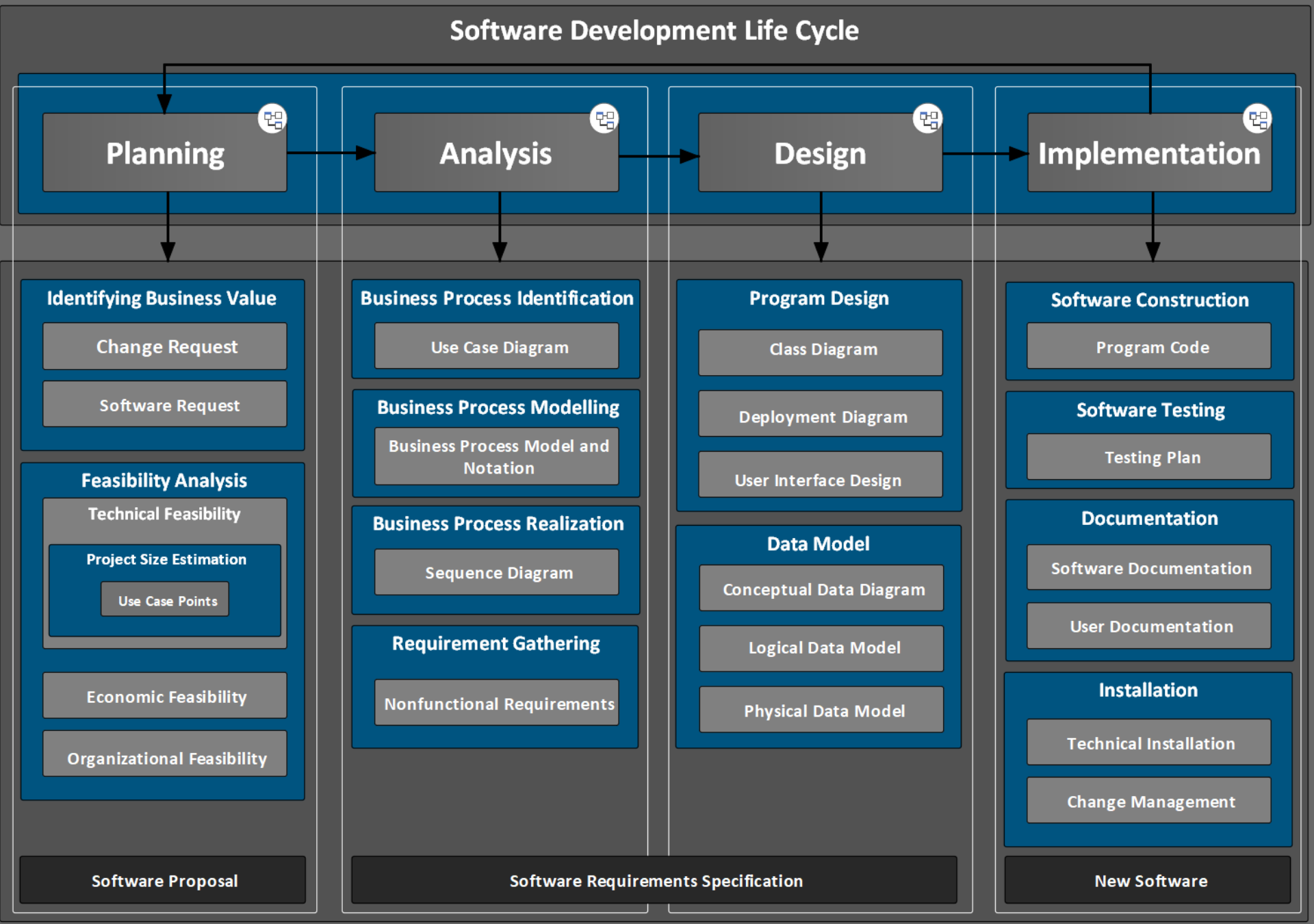


# UML based Software Analysis and Design

*(Wahono, 2009)*

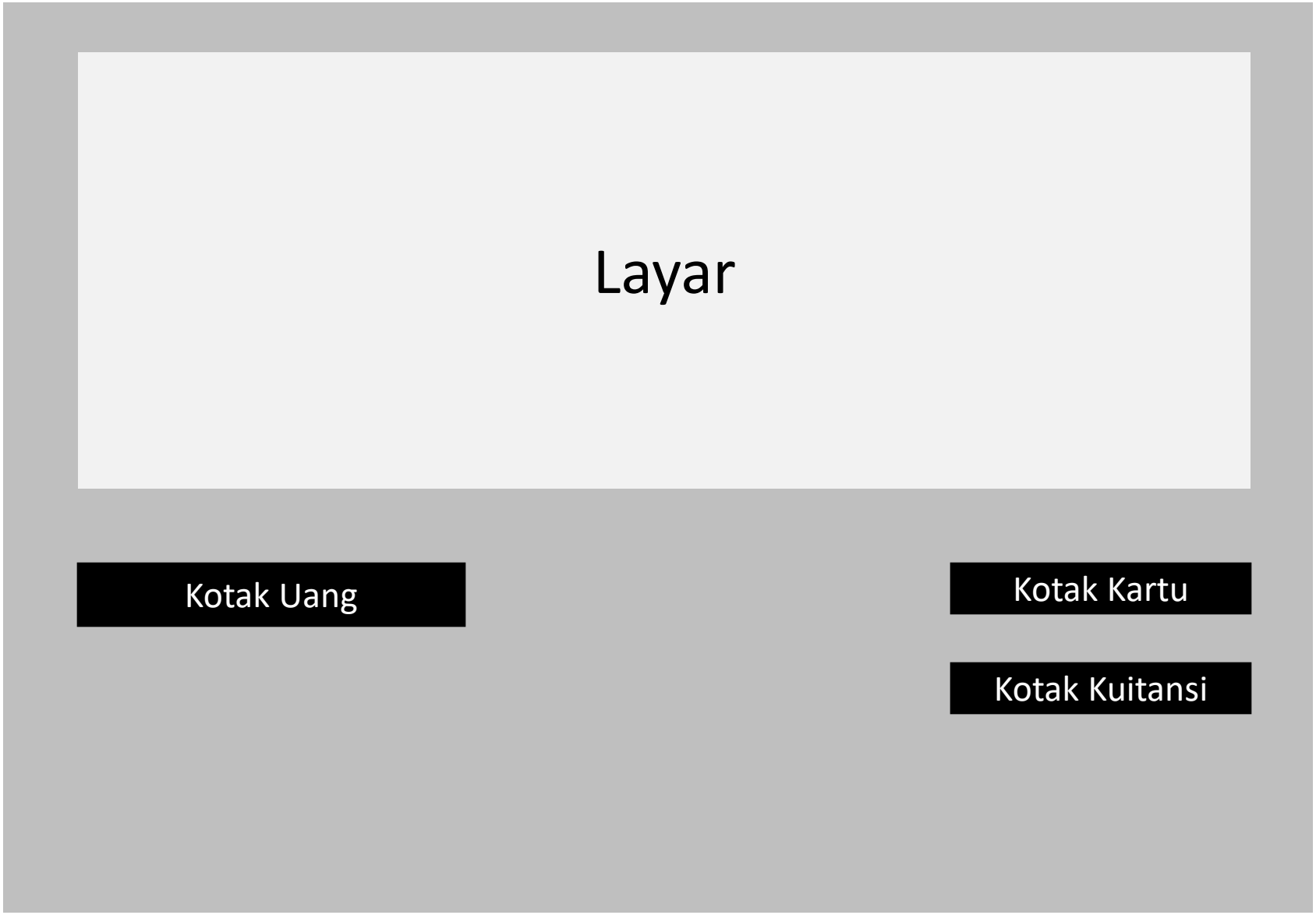


# Application Development Governance



# Studi Kasus: ATM System

# ATM System



# Menu PIN

Masukkan PIN:

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Utama

1. Mengecek Saldo
2. Mentransfer Uang
3. Mengambil Uang
4. Logout

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Pengecekan Saldo

1. Saldo anda adalah …

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Pengiriman Uang

1. No Account Penerima:

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Pengiriman Uang

1. Jumlah uang yang dikirim:

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Pengiriman Uang

1. Uang berhasil terkirim

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Pengambilan Uang

1. Jumlah uang yang diambil:

Kotak Uang

Kotak Kartu

Kotak Kuitansi

# Menu Pengambilan Uang

Uang berhasil diambil

Kotak Uang

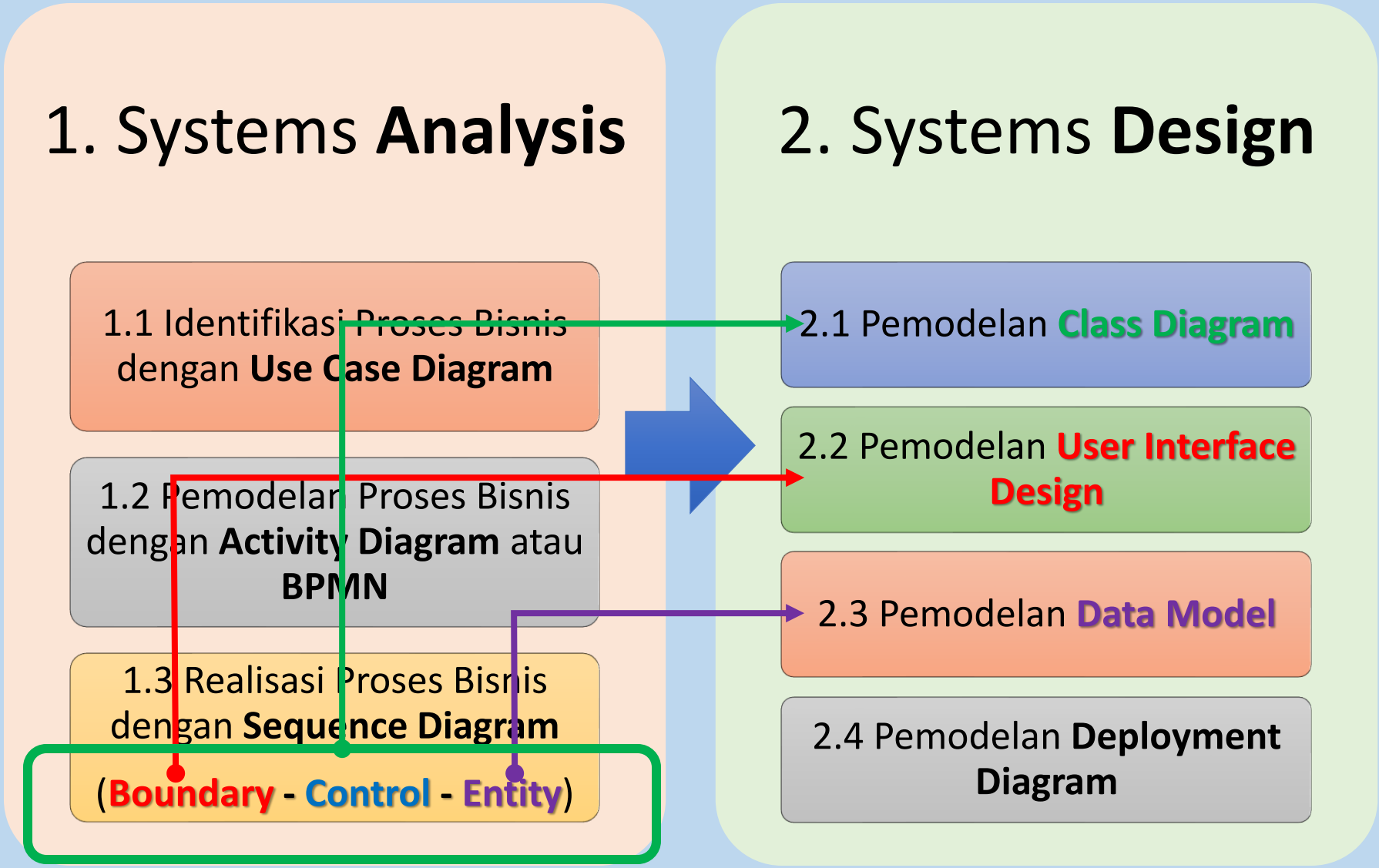
Kotak Kartu

Kotak Kuitansi

# 2.2 Identifikasi Proses Bisnis dengan Use Case Diagram

# UML based Software Analysis and Design

*(Wahono, 2009)*

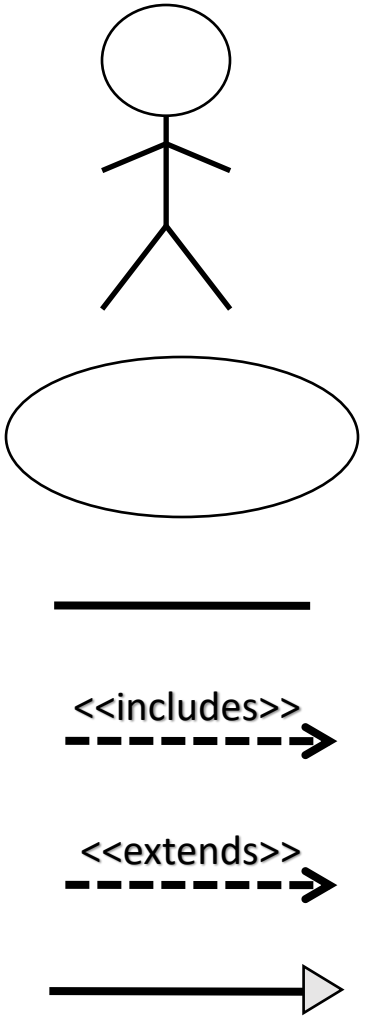


# Use Case Diagram

- Summarized into a single picture
- All of the use cases for the part of the system being modeled
- Use case represents the discrete activities performed by the user
- Use Case Diagram tells what the system will do
- Good for communicating with users

# Use Case Diagram Syntax

- Actor
  - person or system that derives benefit from and is external to the subject
- Use Case
  - Represents a major piece of system functionality
- Association Relationship
- Include Relationship
- Extend Relationship
- Generalization Relationship



# Use Case

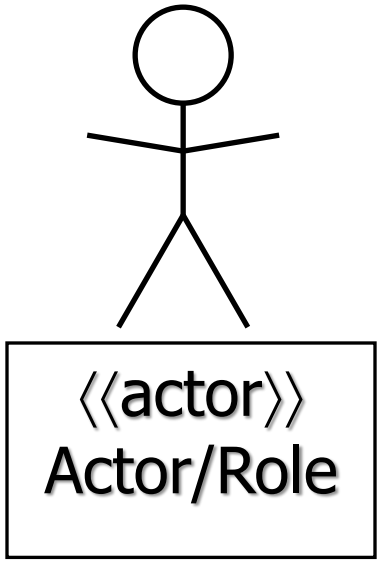
- A major piece of system functionality
- Can extend other Use Cases
- Placed inside system boundary
- Labeled with descriptive verb - noun phrase

# System Boundary

- Includes the name of the system inside or on top
- Represents the scope of the system
- Actors are outside the scope of the system

# Actor

- A person or another system that interacts with the current system
- A role, not a specific user
- Provides input, receives output, or both

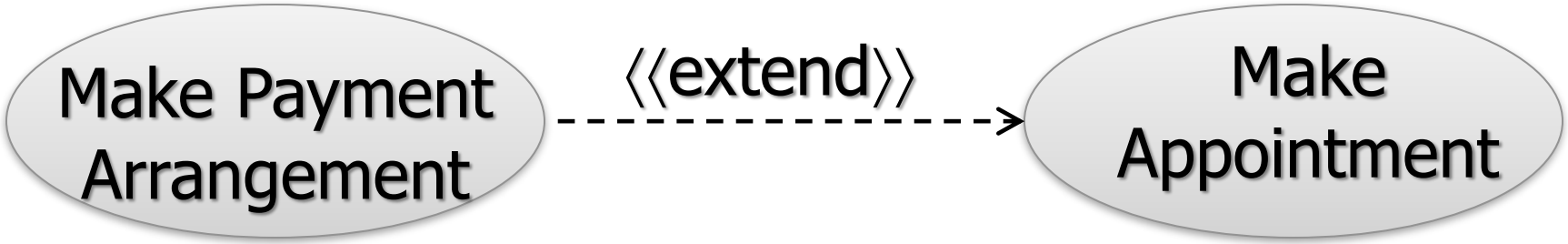


# Association Relationship

- Links actor and the Use Case
- Shows two-way communication
  - If one-way, arrows are used

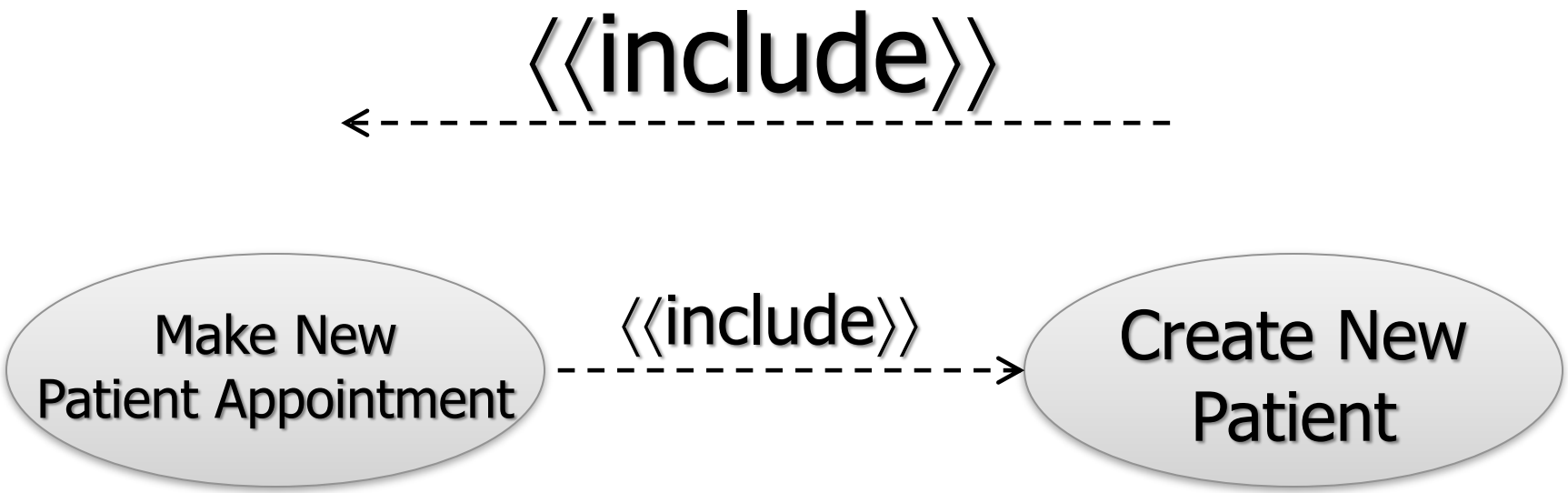
# Extends Relationship

- Extends Use Case to include Optional behavior
- Arrow points from the extension Use Case to the base Use Case

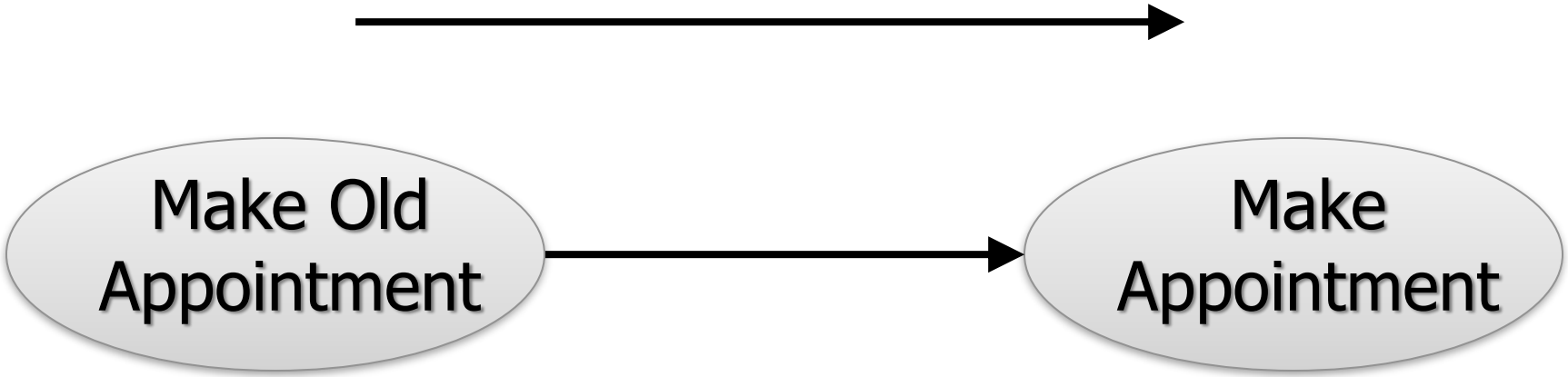
# Include Relationship

- Include one Use Case from within another
- Arrow points from base Use Case to the included Use Case

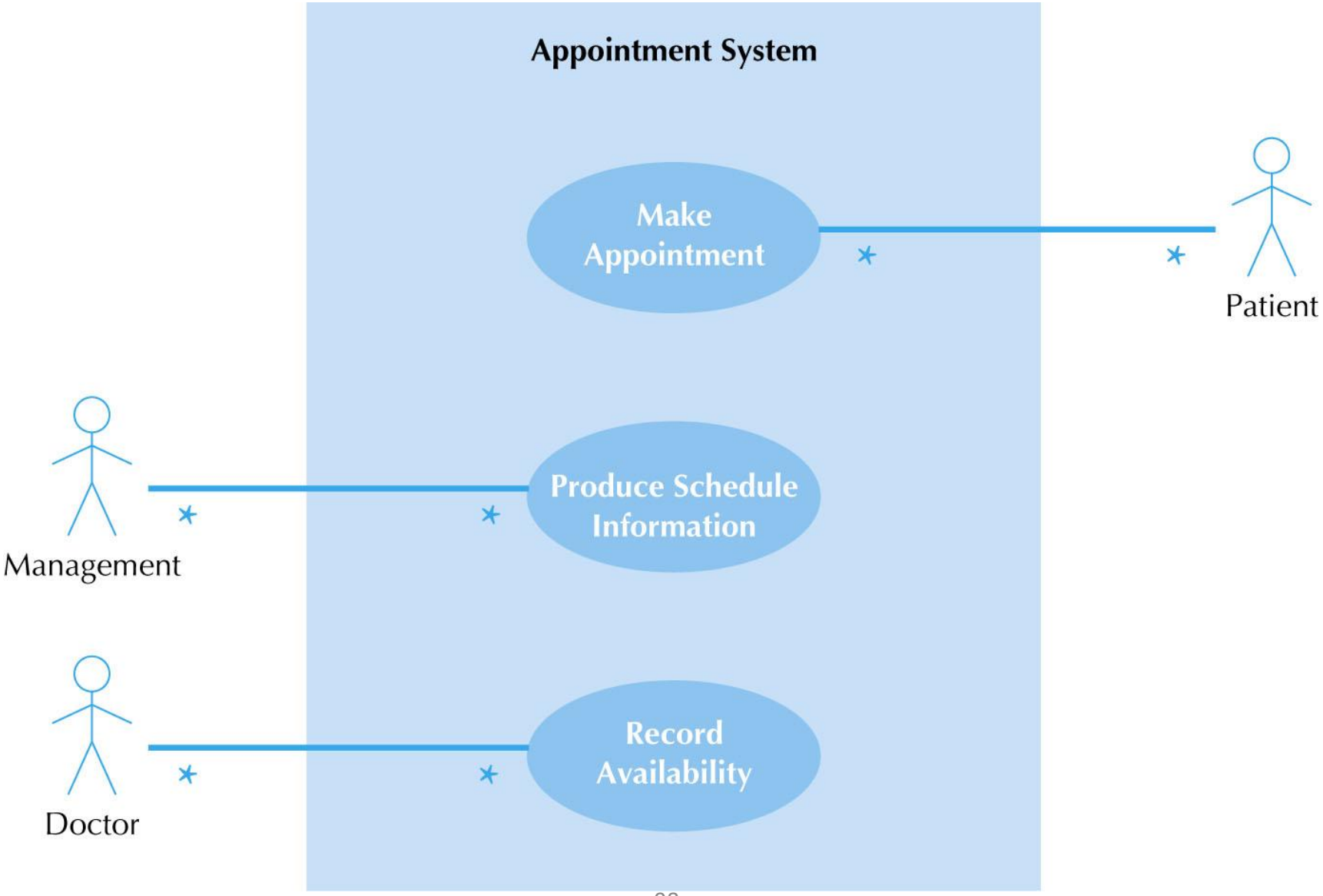


# Generalization Relationship

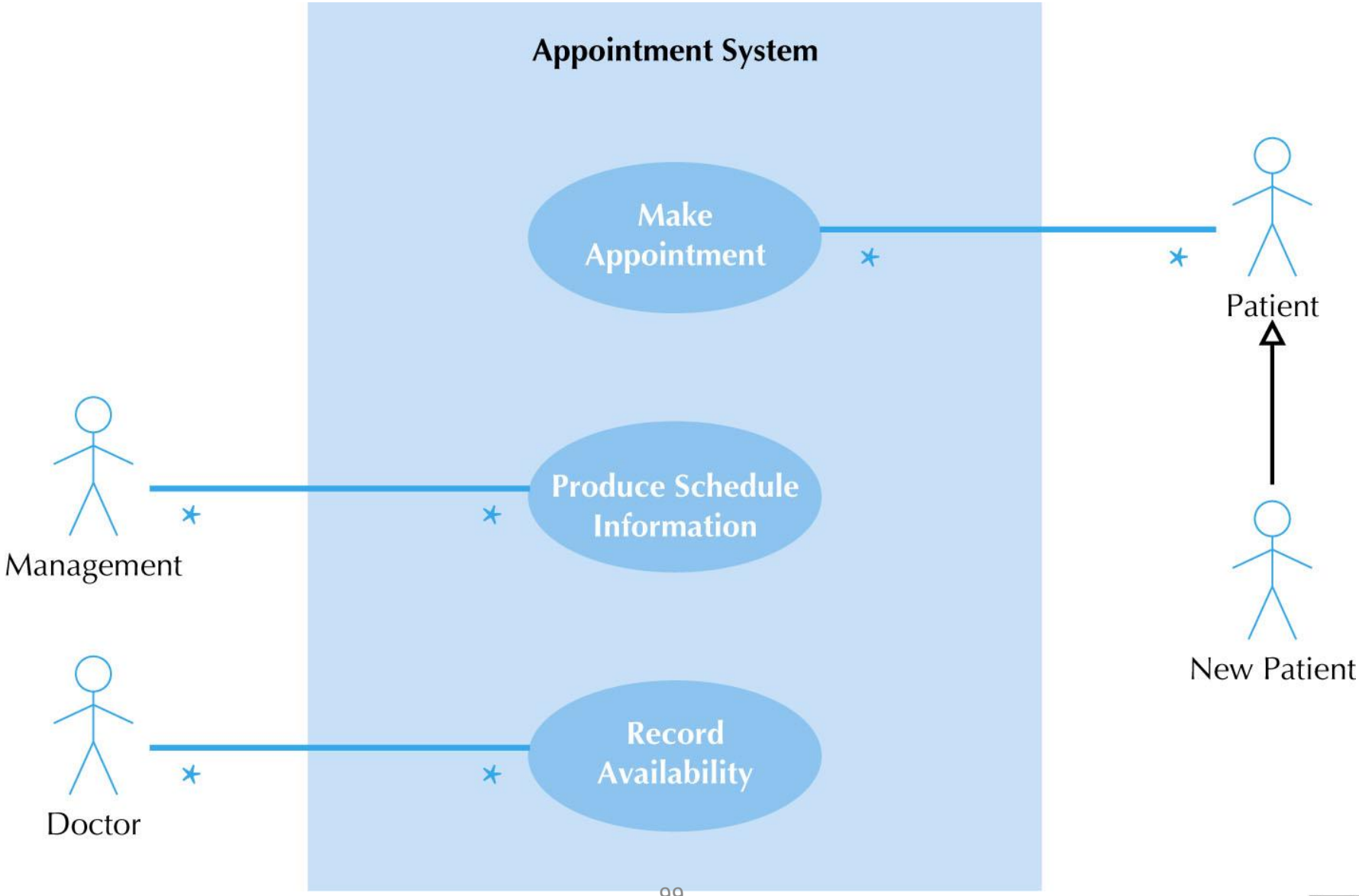
- A specialized Use Case to a more generalized Use Case
- Arrow points from specialized to general Use Case



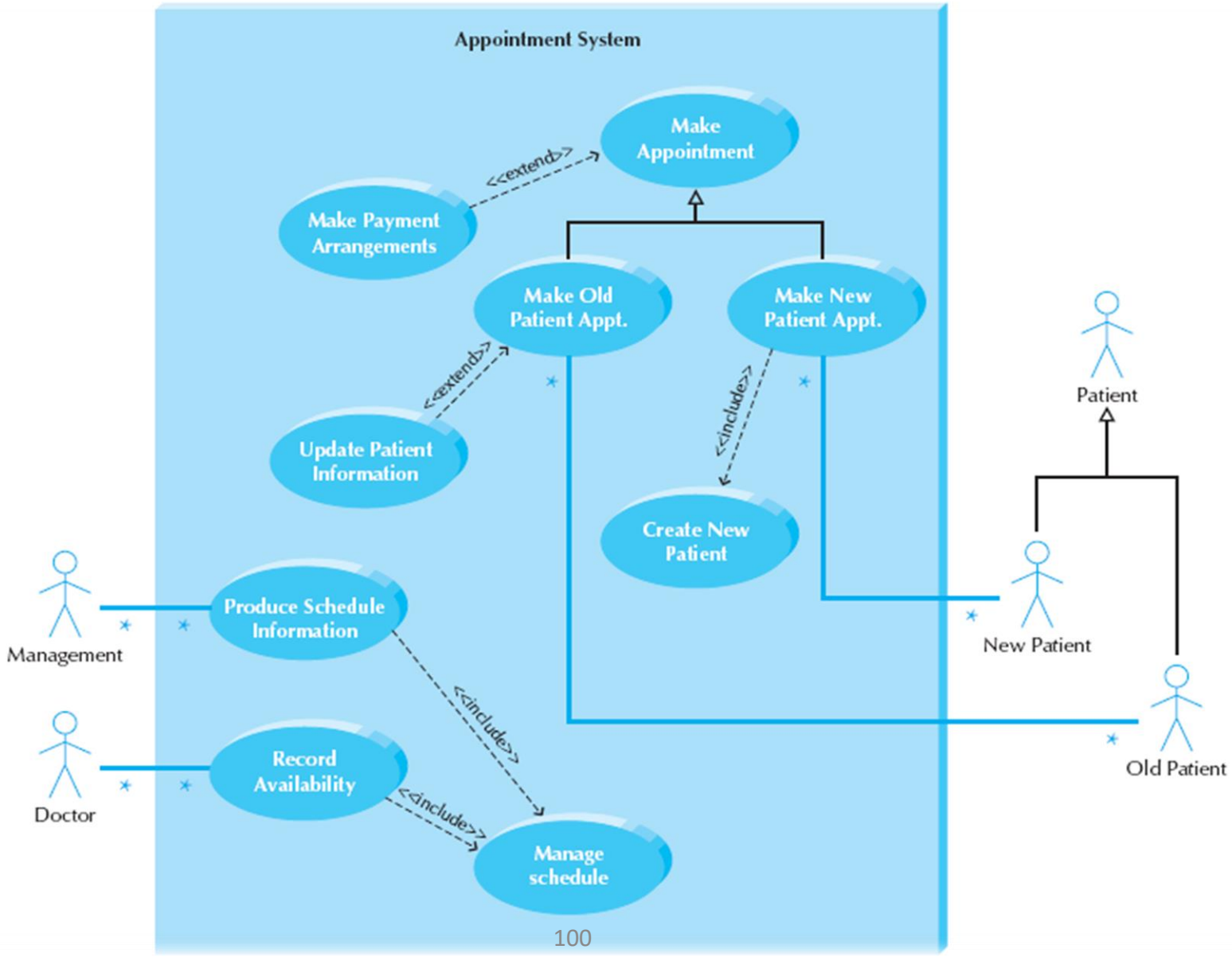
# Use Case Diagram for Appointment System



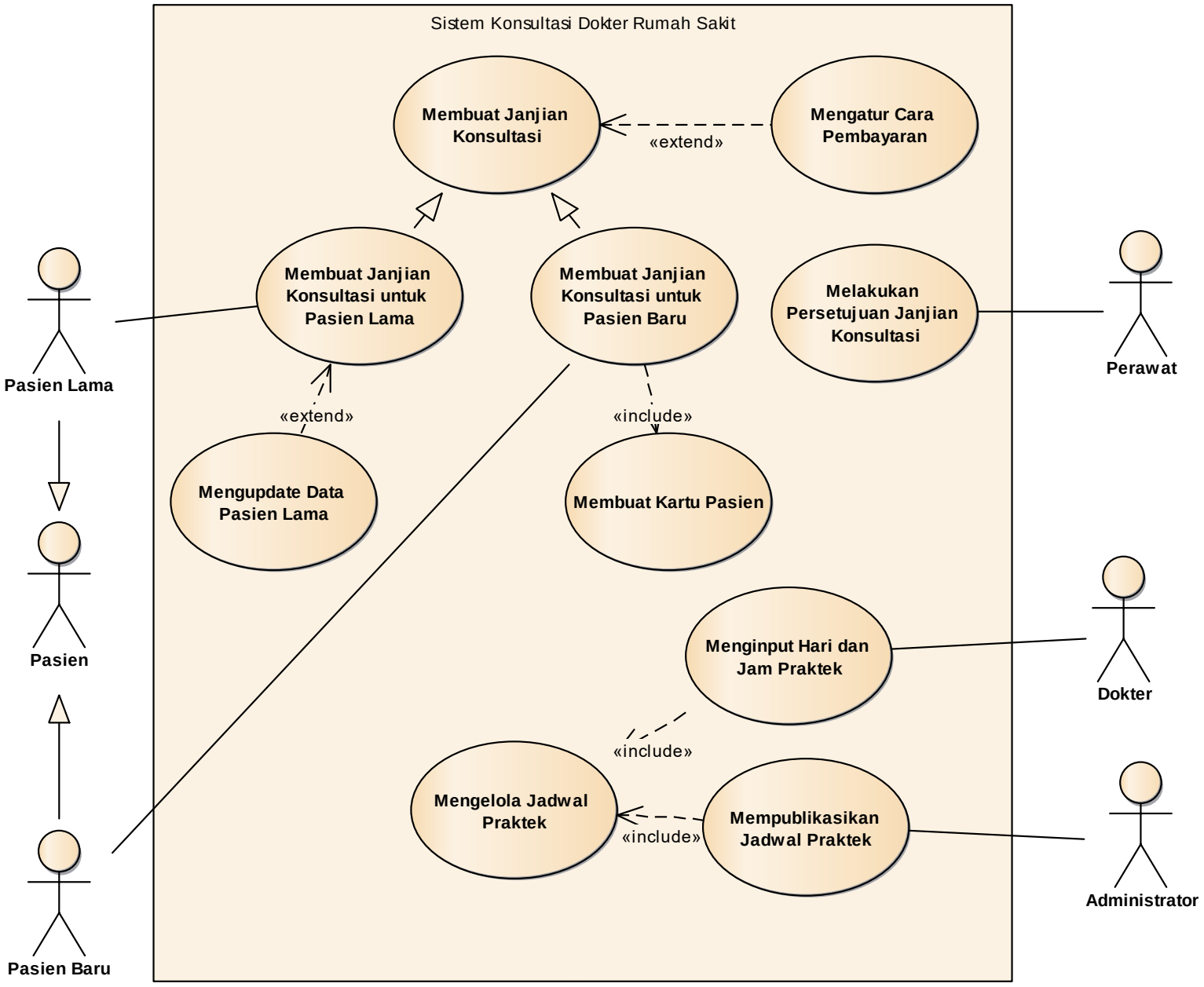
# Use Case Diagram with Specialized Actor



# Extend and Include Relationships

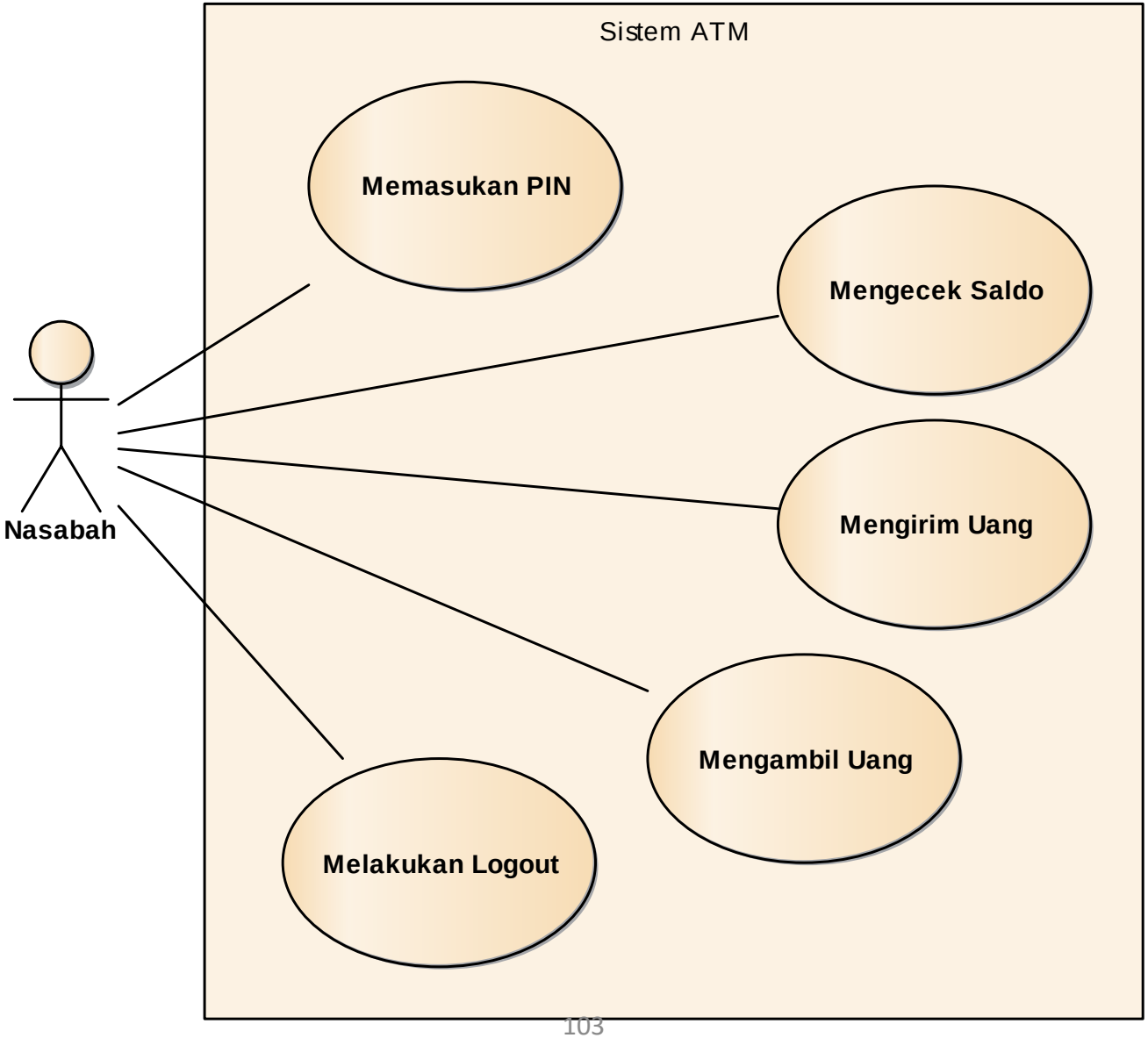


# Sistem Konsultasi Dokter Rumah Sakit

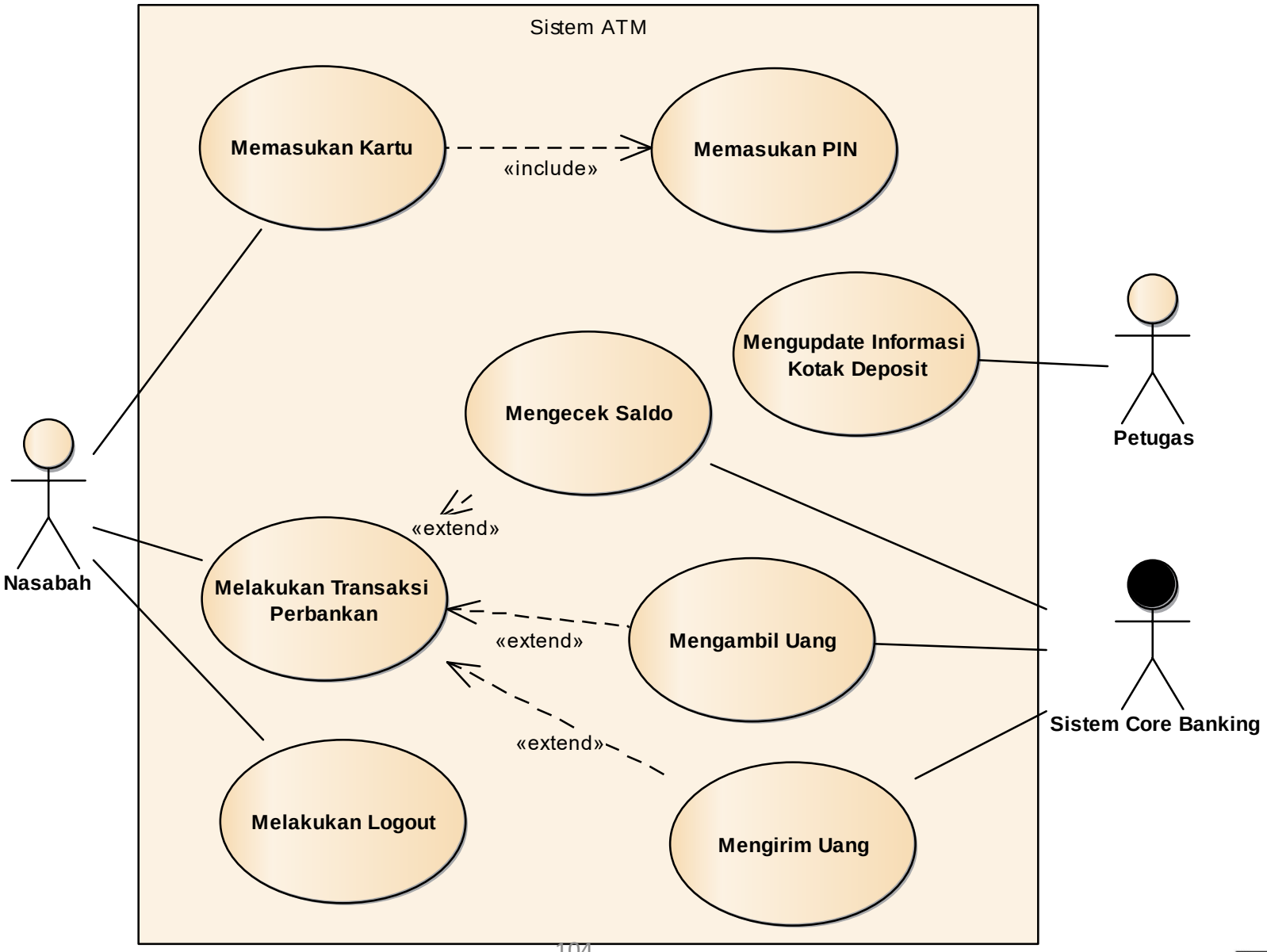


# Studi Kasus: Use Case Diagram Sistem ATM

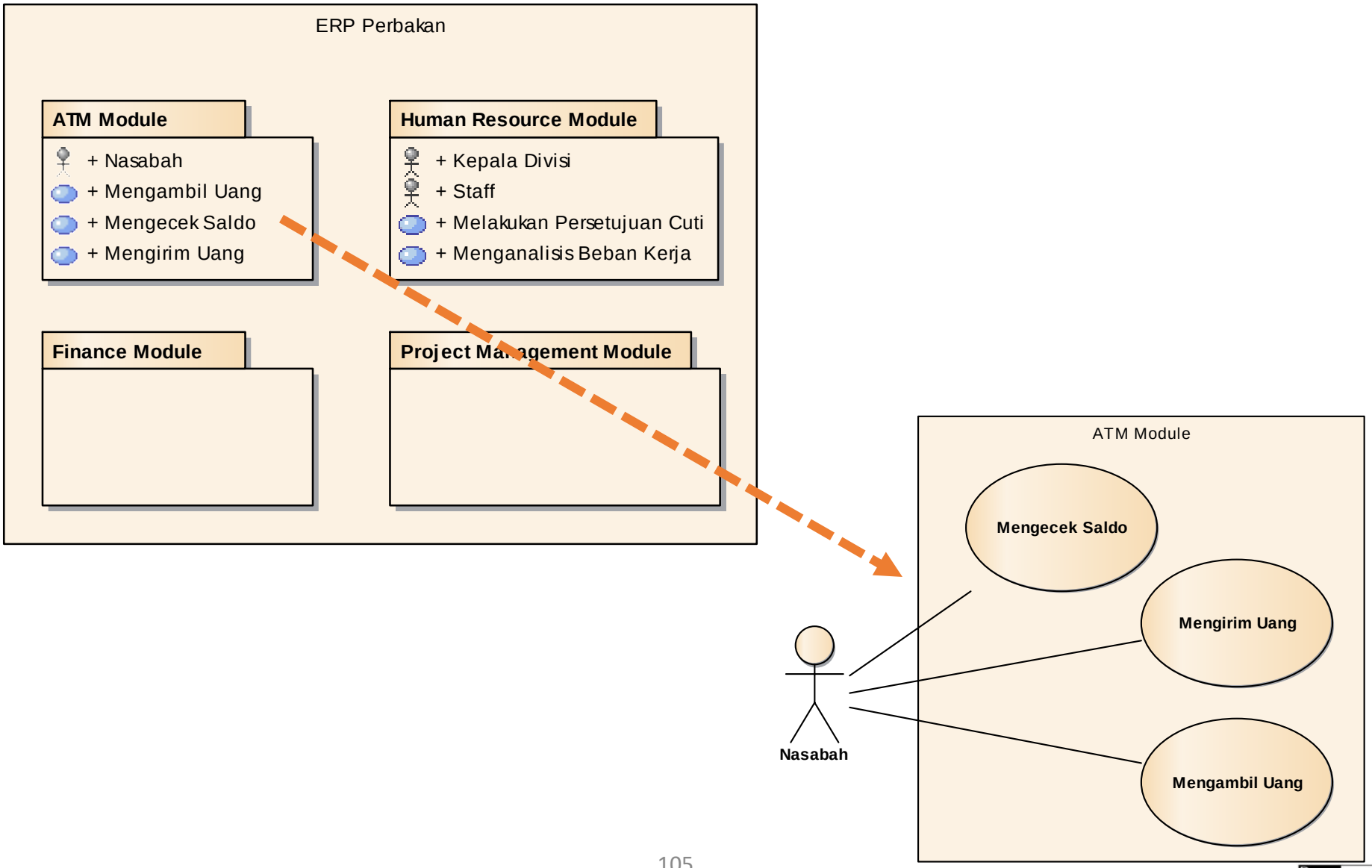
# Use Case Diagram Sistem ATM
## (versi Sederhana)



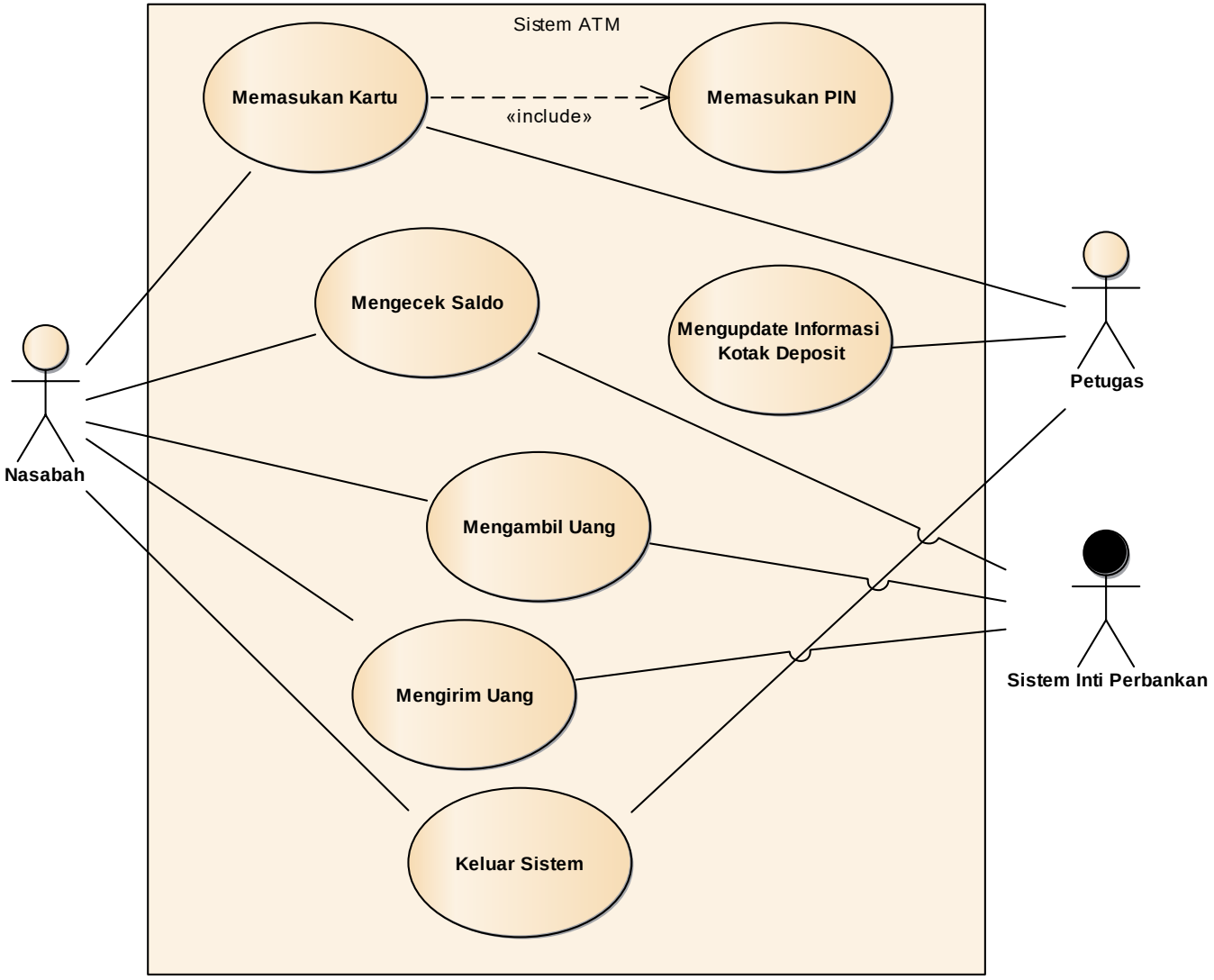
# Use Case Diagram Sistem ATM
## (Versi Include dan Extends)

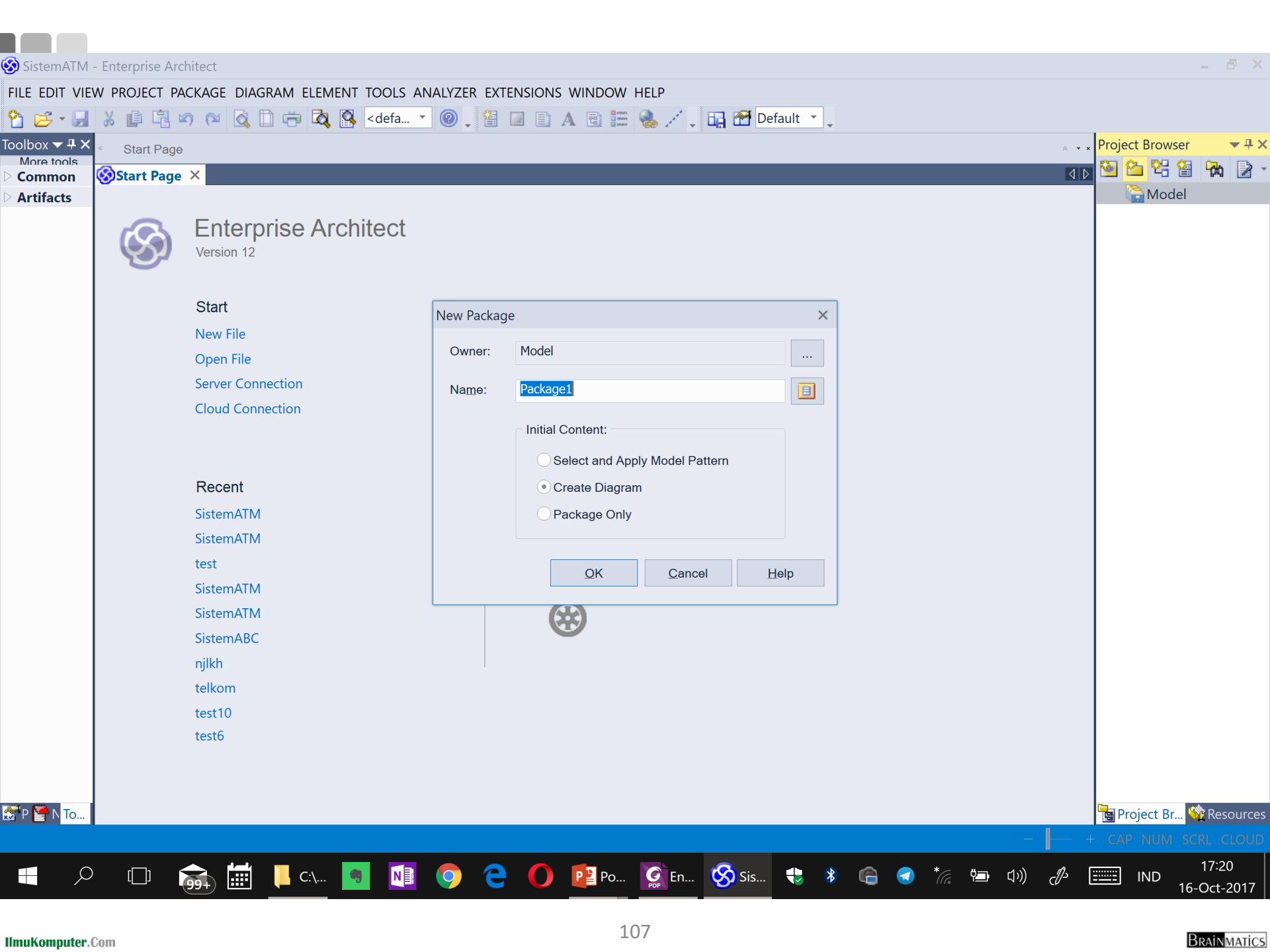


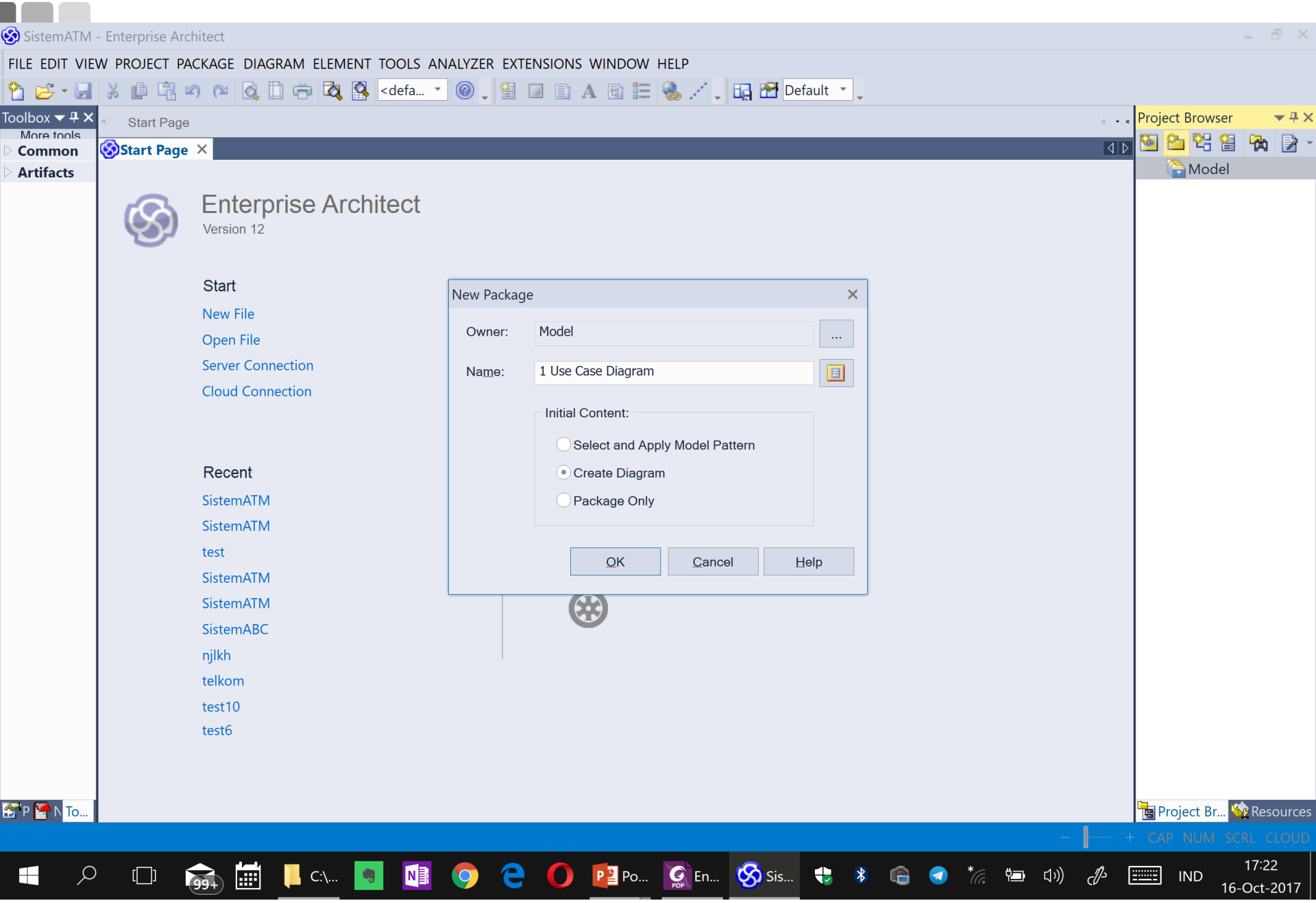
# Use Case Diagram ERP Perbankan
## (Sistem Lebih Kompleks)

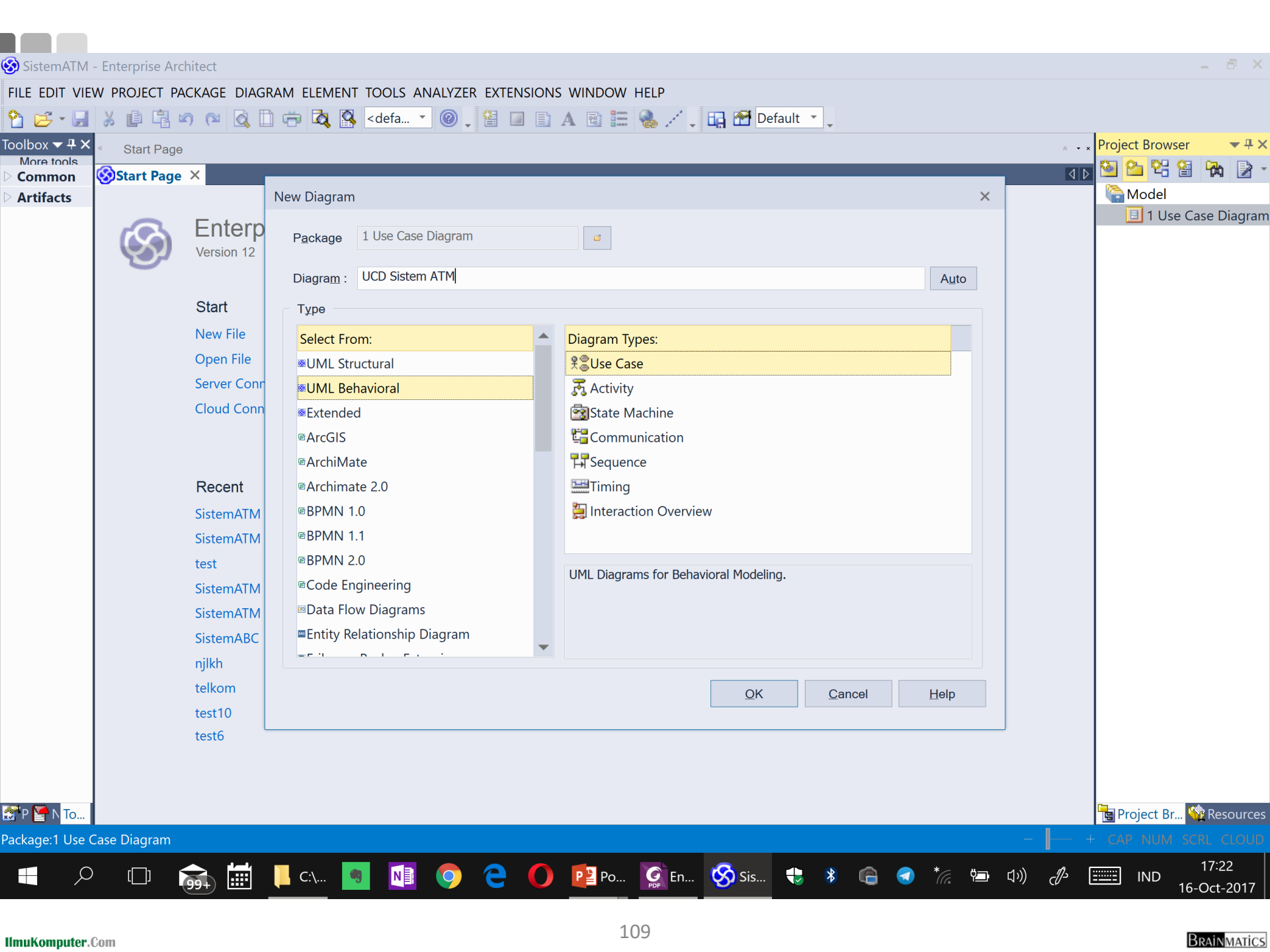


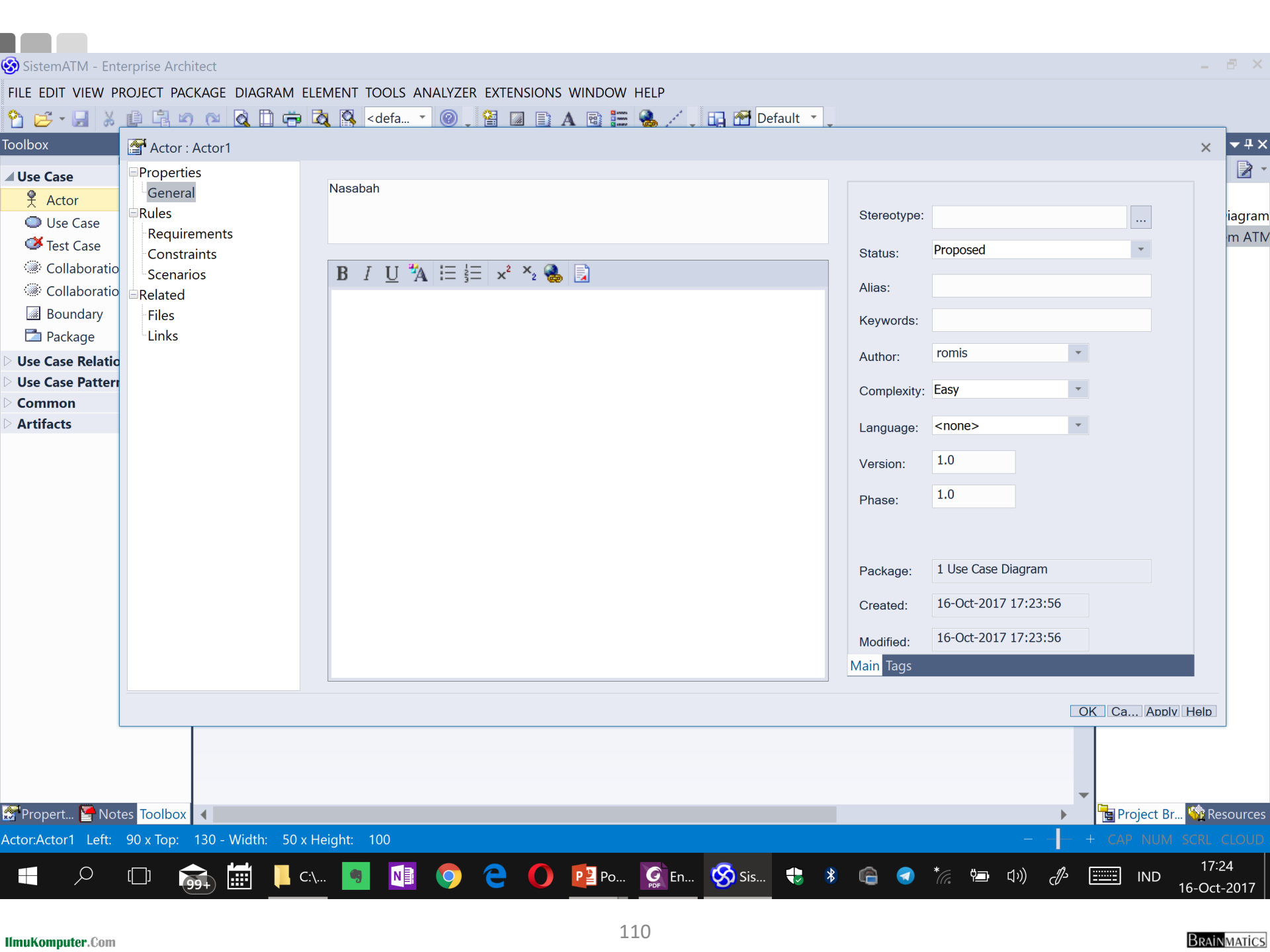
# Use Case Diagram Sistem ATM
## (Versi Normal)











SistemATM - Enterprise Architect

FILE EDIT VIEW PROJECT PACKAGE DIAGRAM ELEMENT TOOLS ANALYZER EXTENSIONS WINDOW HELP

Toolbox

More tools

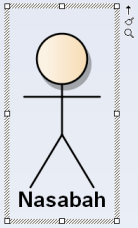
Use Case
- Actor
- Use Case
- Test Case
- Collaboration
- Collaboration Use
- Boundary
- Package

Use case and Actor elements for Usecase diagrams

Use Case Relationships
Use Case Patterns
Common
Artifacts

Use Case Diagram: "UCD Sistem ATM"

Start Page | *UCD Sistem ATM




Project Browser

Model
1 Use Case Diagram
UCD Sistem ATM
Nasabah

Propert... Notes Toolbox

Project Br... Resources

Actor:Nasabah Left: 90 x Top: 130 - Width: 50 x Height: 100

CAP NUM SCRL CLOUD

IND 17:24 16-Oct-2017

SistemATM - Enterprise Architect

FILE EDIT VIEW PROJECT PACKAGE DIAGRAM ELEMENT TOOLS ANALYZER EXTENSIONS WINDOW HELP

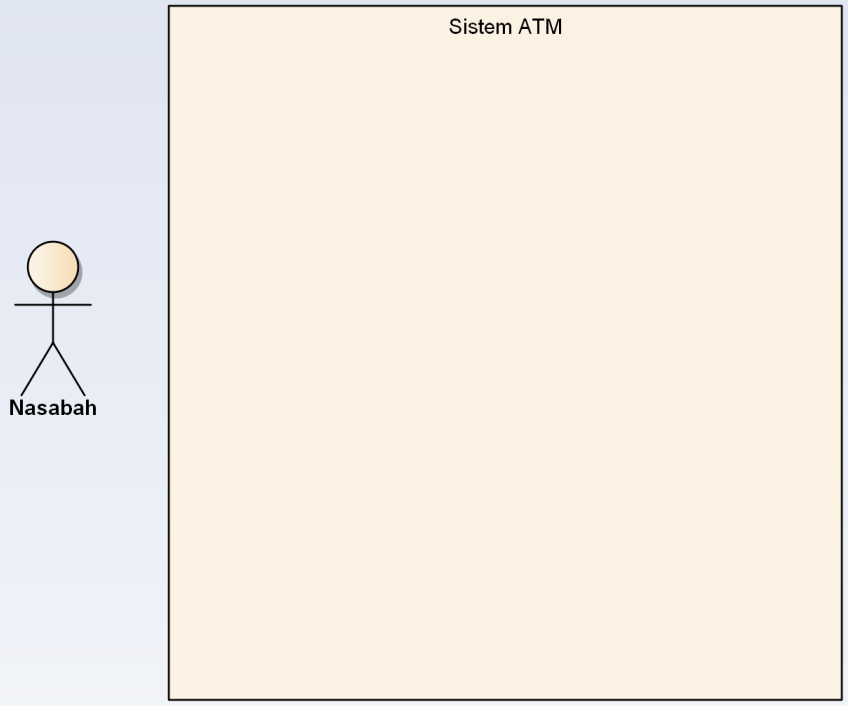
Toolbox

More tools

- Use Case
  - Actor
  - Use Case
  - Test Case
  - Collaboration
  - Collaboration Use
  - Boundary
  - Package
- Use Case Relationships
- Use Case Patterns
- Common
- Artifacts

Use Case Diagram: "UCD Sistem ATM"

Start Page | *UCD Sistem ATM




Project Browser

- Model
  - 1 Use Case Diagram
    - UCD Sistem ATM
    - Nasabah

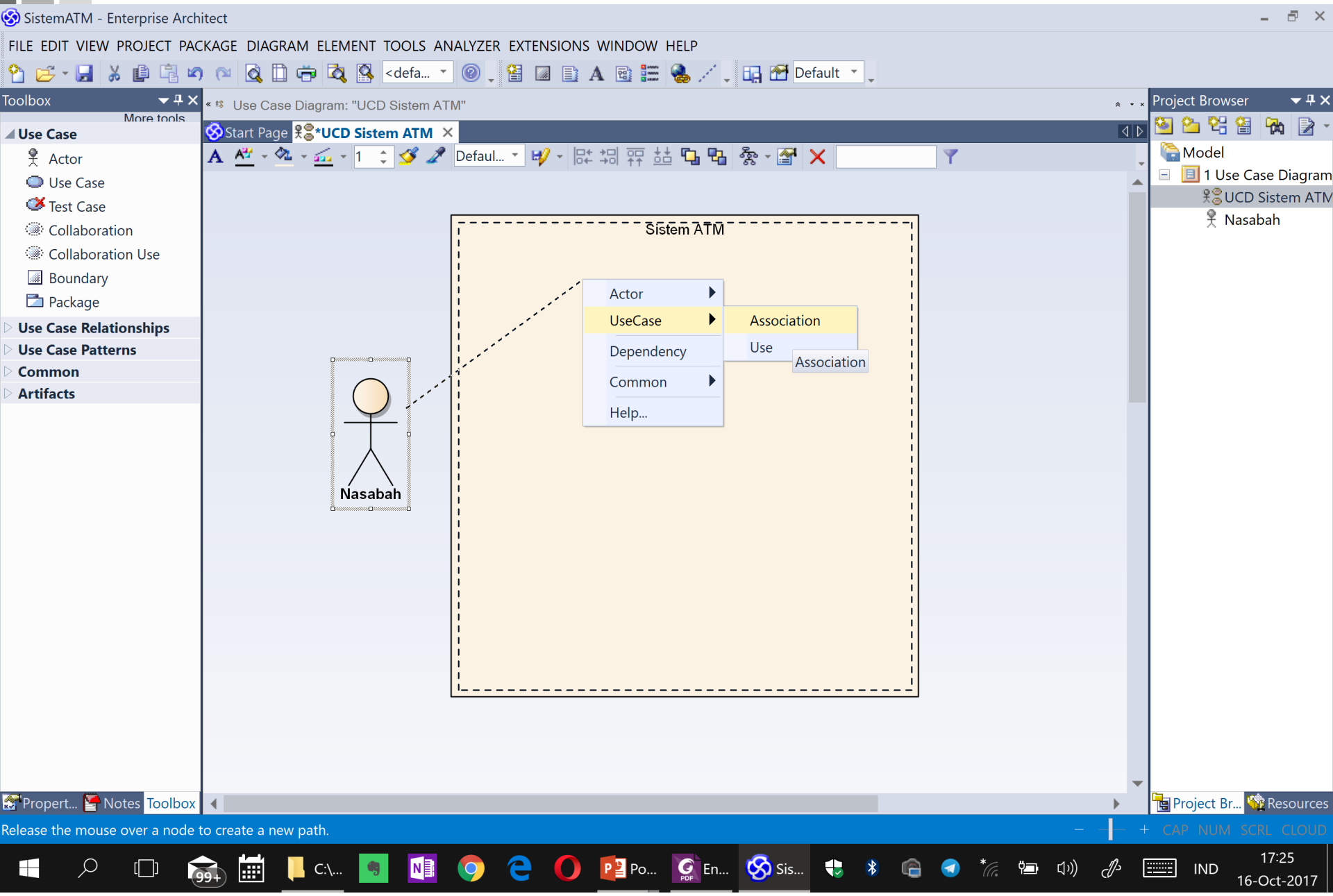
Propert... | Notes | Toolbox

Project Br... | Resources

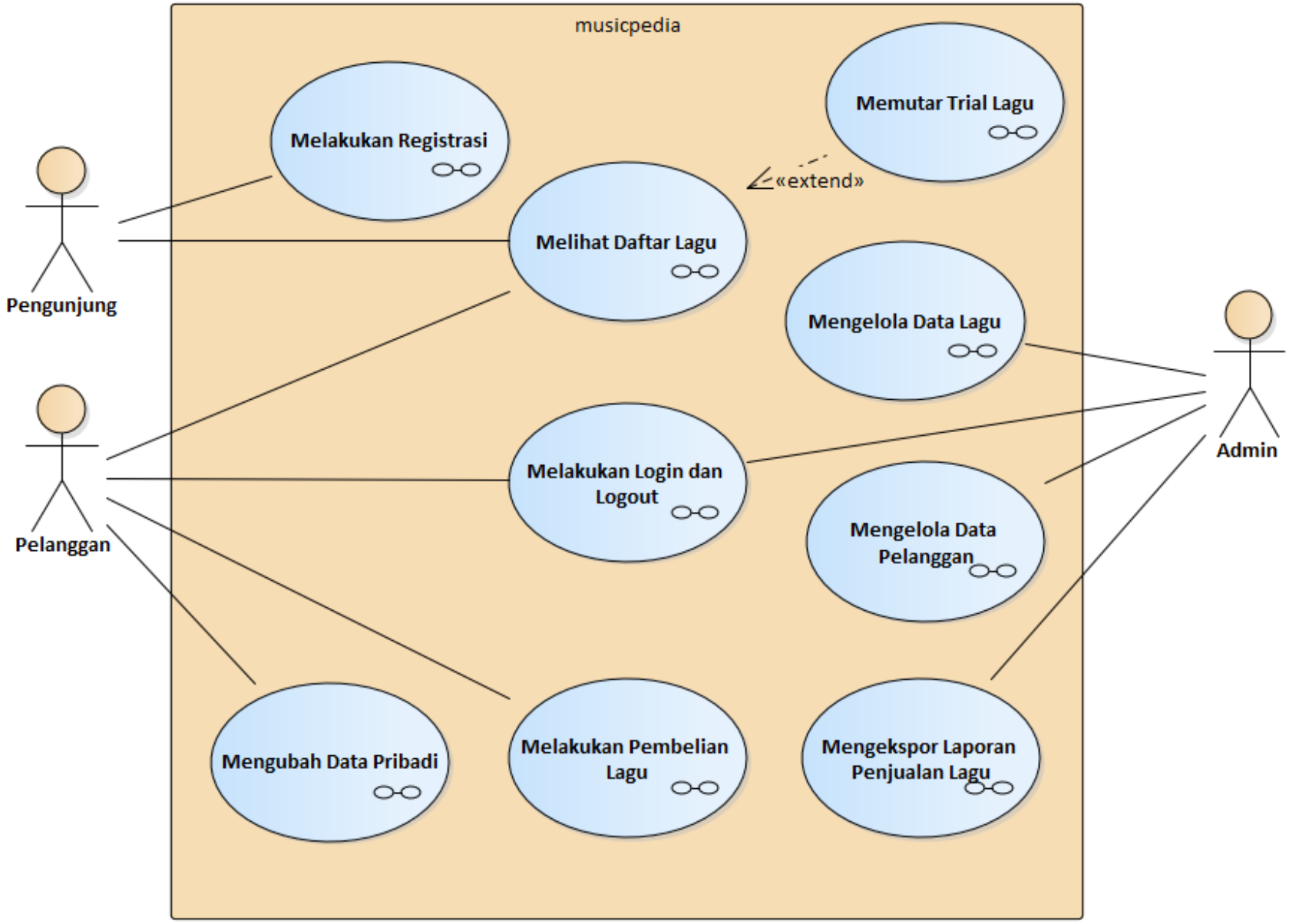
Use Case Diagram:UCD Sistem ATM: created: 16-Oct-2017 17:23:00 modified: 16-Oct-2017 17:24:51 100% 827 x 1169

CAP NUM SCRL CLOUD

IND 17:25 16-Oct-2017



# Use Case Diagram MusicPedia



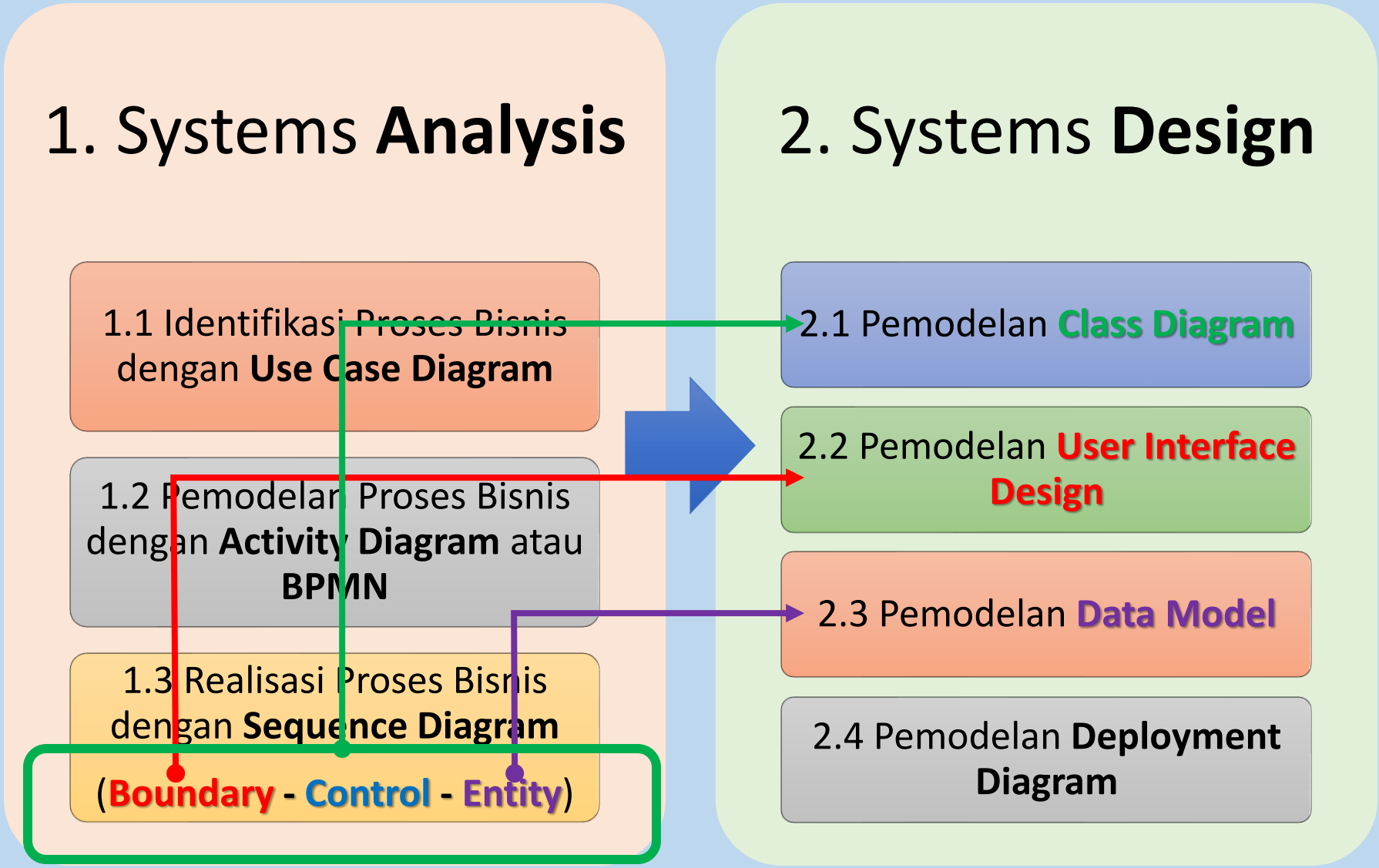
# 2.3 Pemodelan Proses Bisnis dengan AD atau BPMN

2.3.1 Activity Diagram

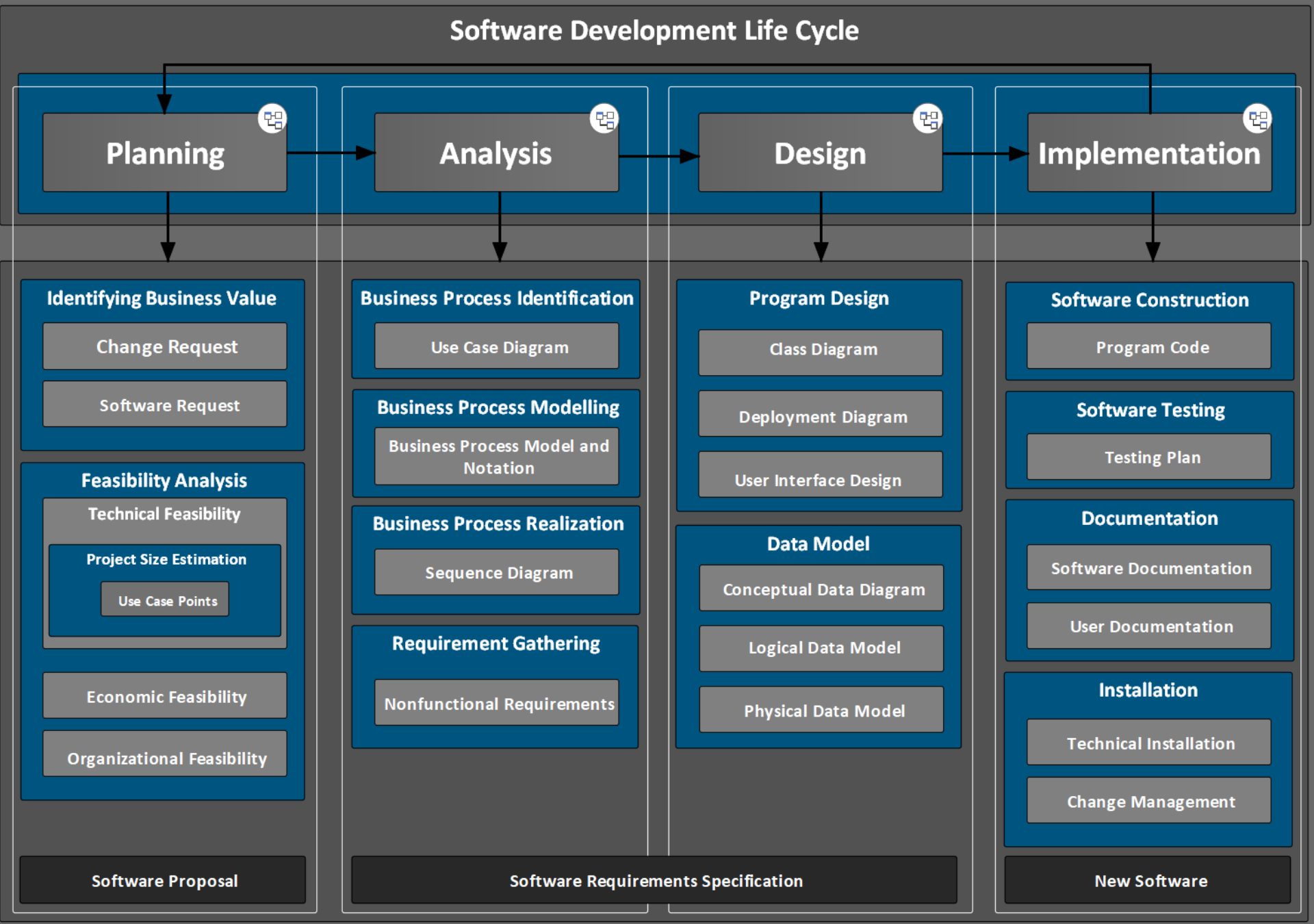
2.3.2 Business Process Model and Notation (BPMN)

# UML based Software Analysis and Design

*(Wahono, 2009)*

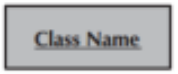
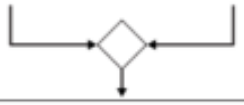


# Application Development Governance



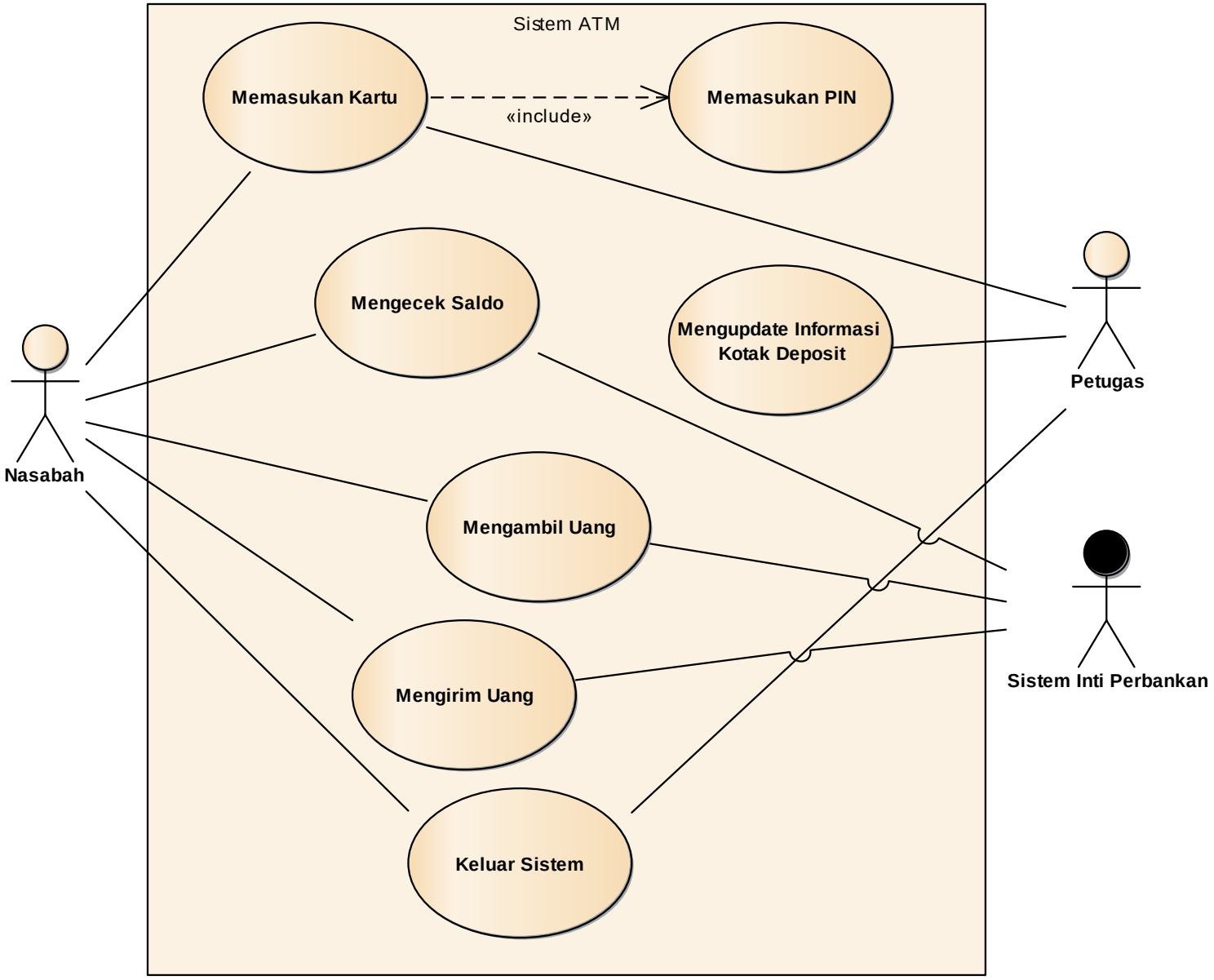
# 2.3.1 Activity Diagram

# Activity Diagram Syntax

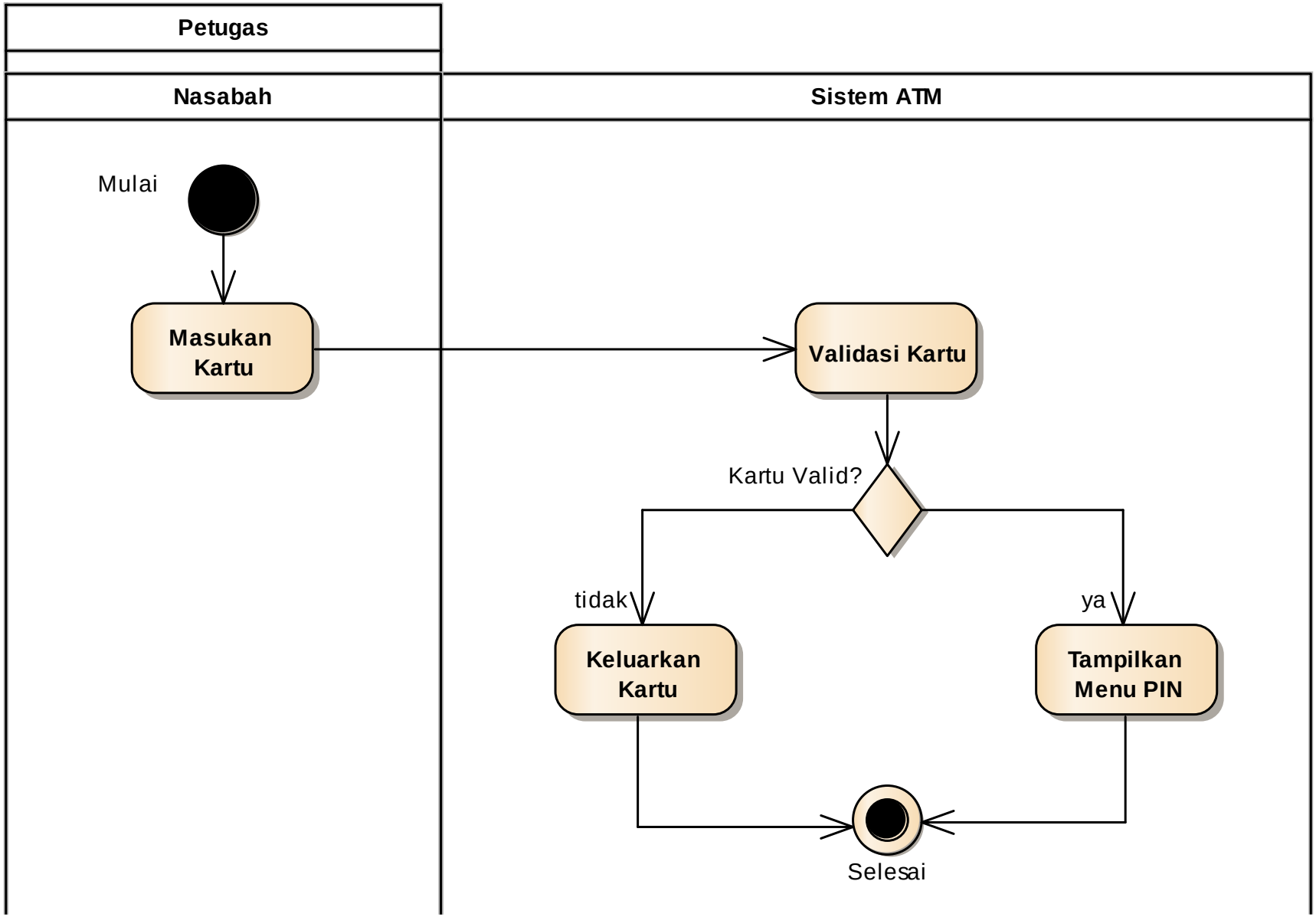
| | |
|---|---|
| **An action:**<br>■ Is a simple, nondecomposable piece of behavior.<br>■ Is labeled by its name. |  |
| **An activity:**<br>■ Is used to represent a set of actions.<br>■ Is labeled by its name. |  |
| **An object node:**<br>■ Is used to represent an object that is connected to a set of object flows.<br>■ Is labeled by its class name. |  |
| **A control flow:**<br>■ Shows the sequence of execution. |  |
| **An object flow:**<br>■ Shows the flow of an object from one activity (or action) to another activity (or action). |  |
| **An initial node:**<br>■ Portrays the beginning of a set of actions or activities. |  |
| **A final-activity node:**<br>■ Is used to stop all control flows and object flows in an activity (or action). |  |
| **A final-flow node:**<br>■ Is used to stop a specific control flow or object flow. |  |
| **A decision node:**<br>■ Is used to represent a test condition to ensure that the control flow or object flow only goes down one path.<br>■ Is labeled with the decision criteria to continue down the specific path. |  |
| **A merge node:**<br>■ Is used to bring back together different decision paths that were created using a decision node. |  |
| **A fork node:**<br>Is used to split behavior into a set of parallel or concurrent flows of activities (or actions) |  |
| **A join node:**<br>Is used to bring back together a set of parallel or concurrent flows of activities (or actions) |  |
| **A swimlane:**<br>Is used to break up an activity diagram into rows and columns to assign the individual activities (or actions) to the individuals or objects that are responsible for executing the activity (or action)<br>Is labeled with the name of the individual or object responsible |  |

# Studi Kasus: Activity Diagram Sistem ATM

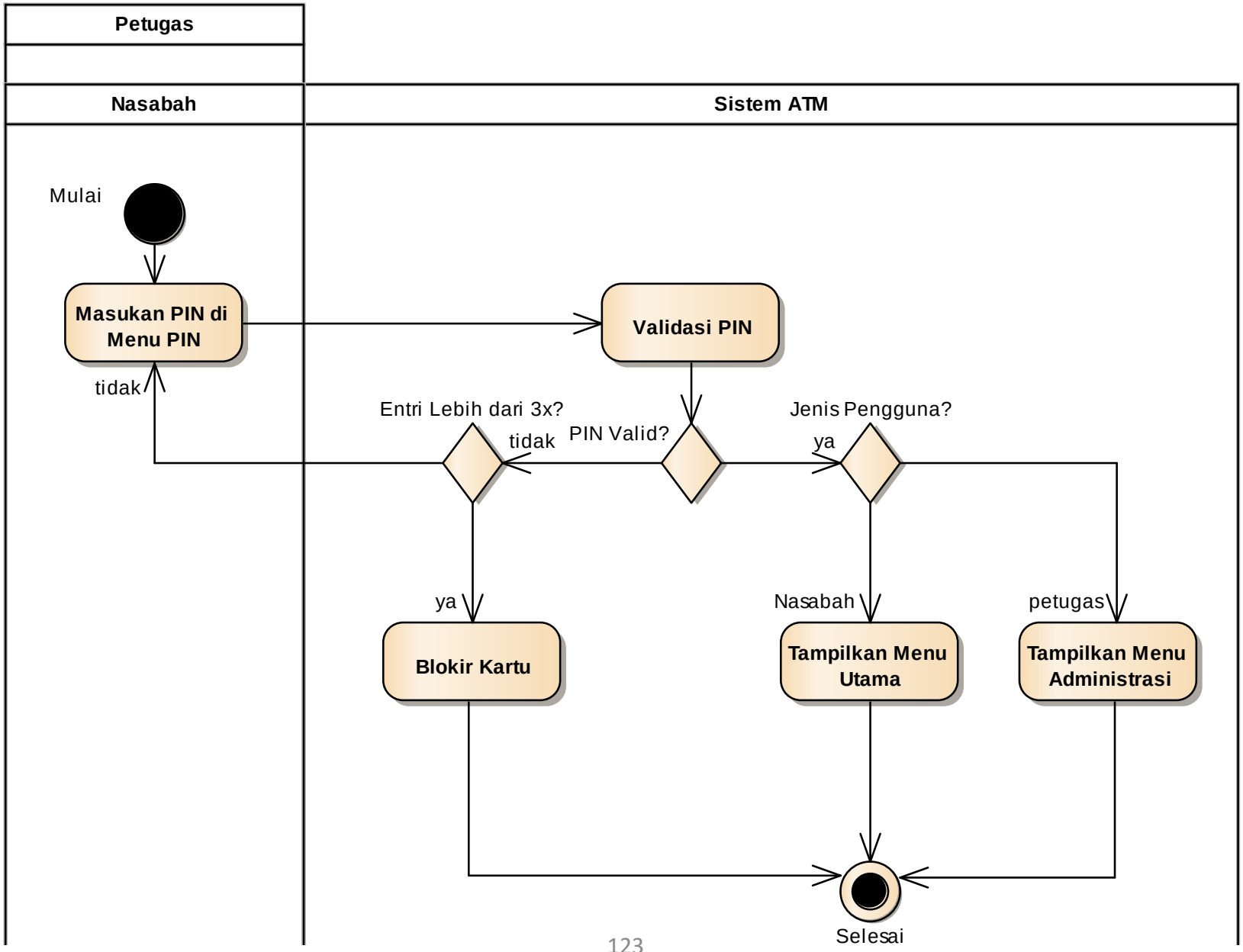
# Use Case Diagram Sistem ATM



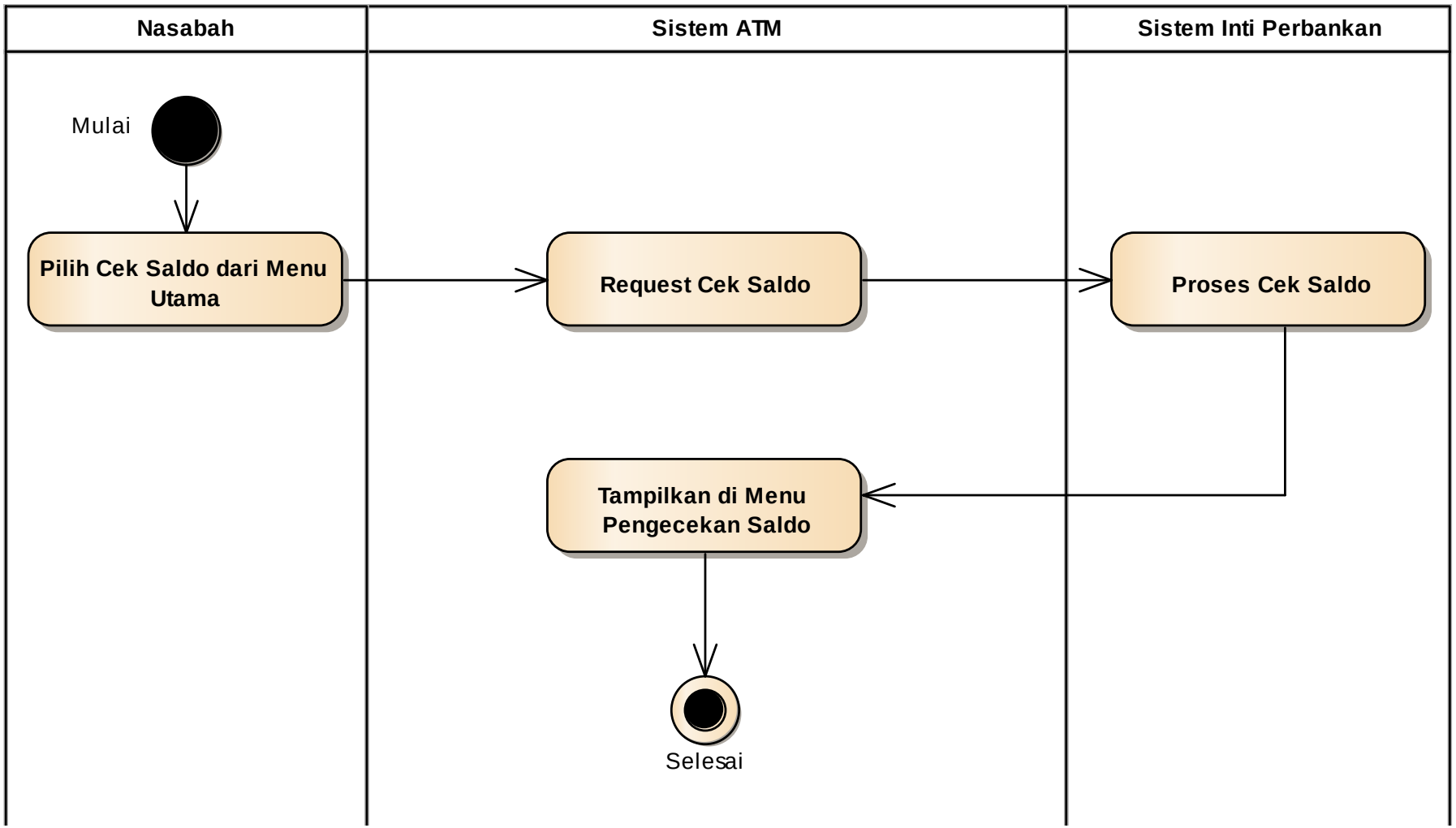
# Activity Diagram: Memasukkan Kartu



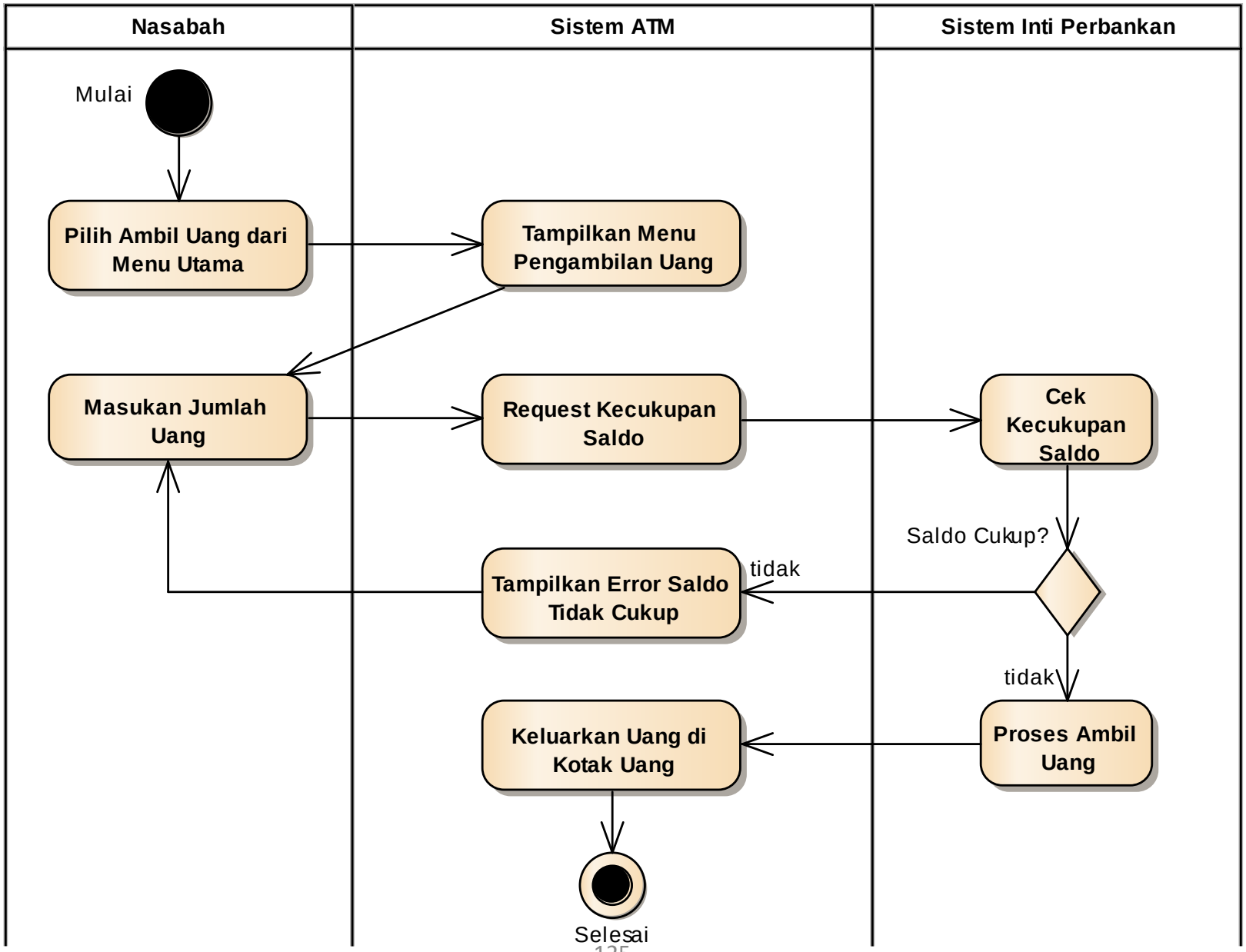
# Activity Diagram: Memasukkan PIN



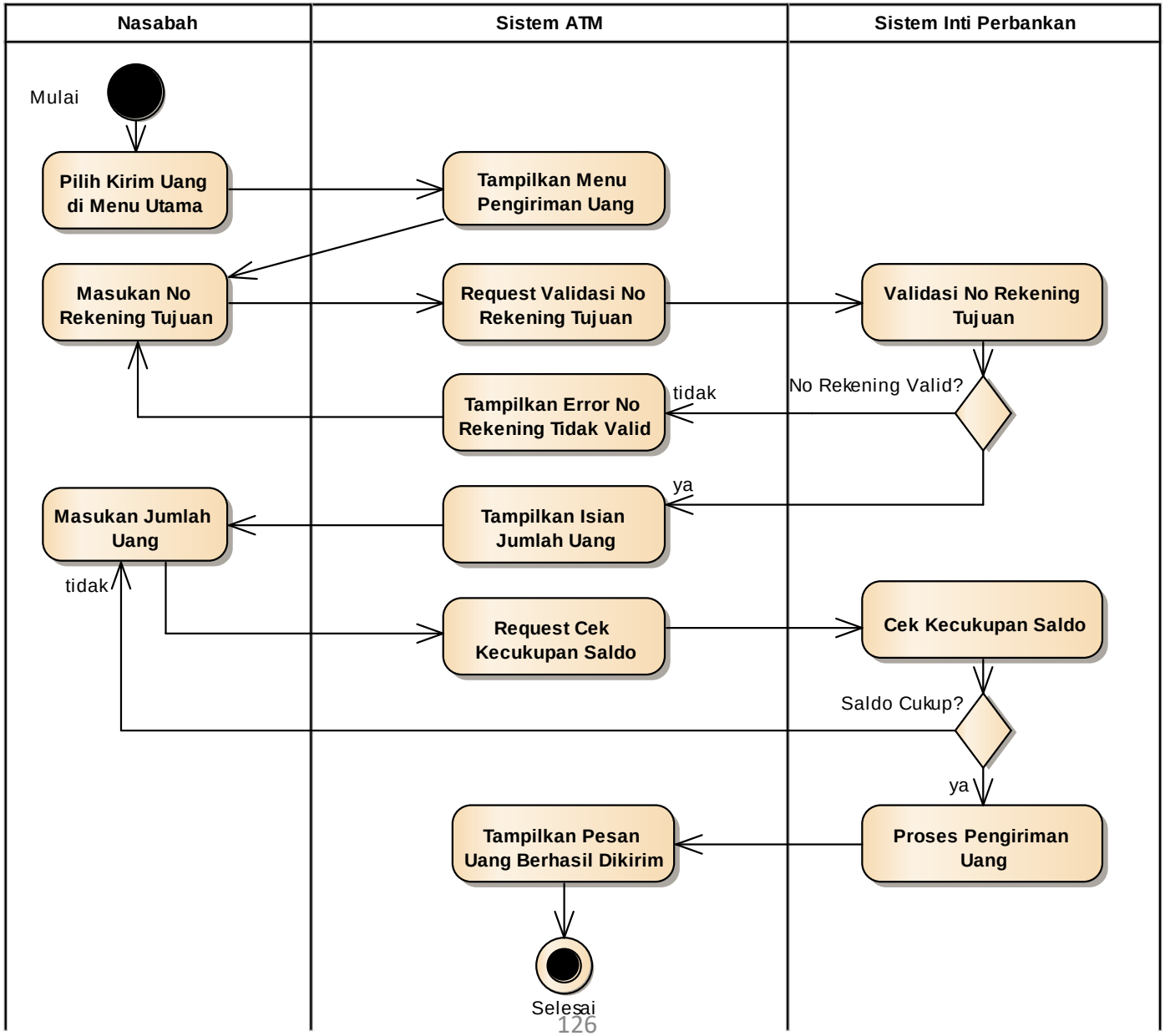
# Activity Diagram: Mengecek Saldo



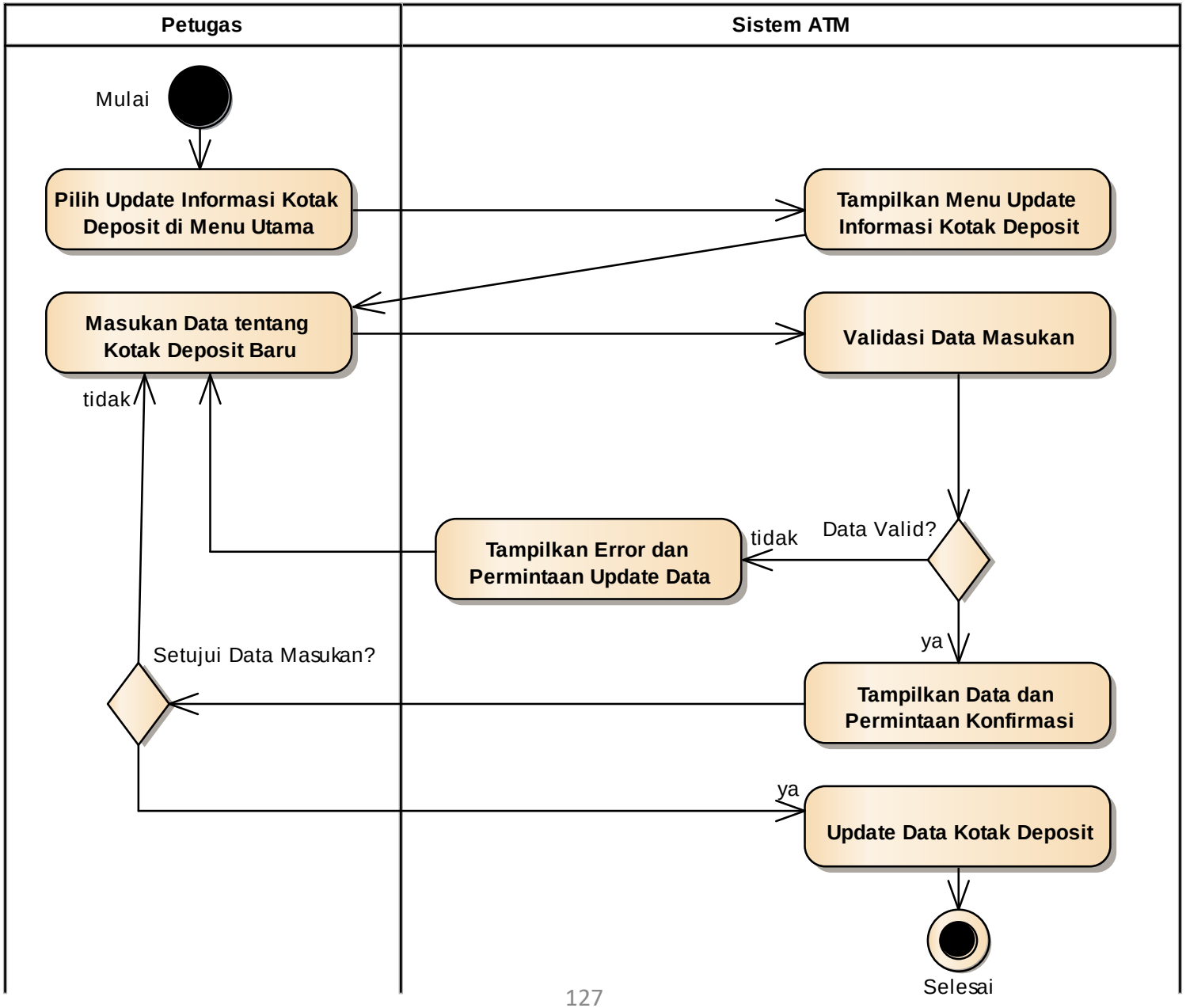
# Activity Diagram: Mengambil Uang



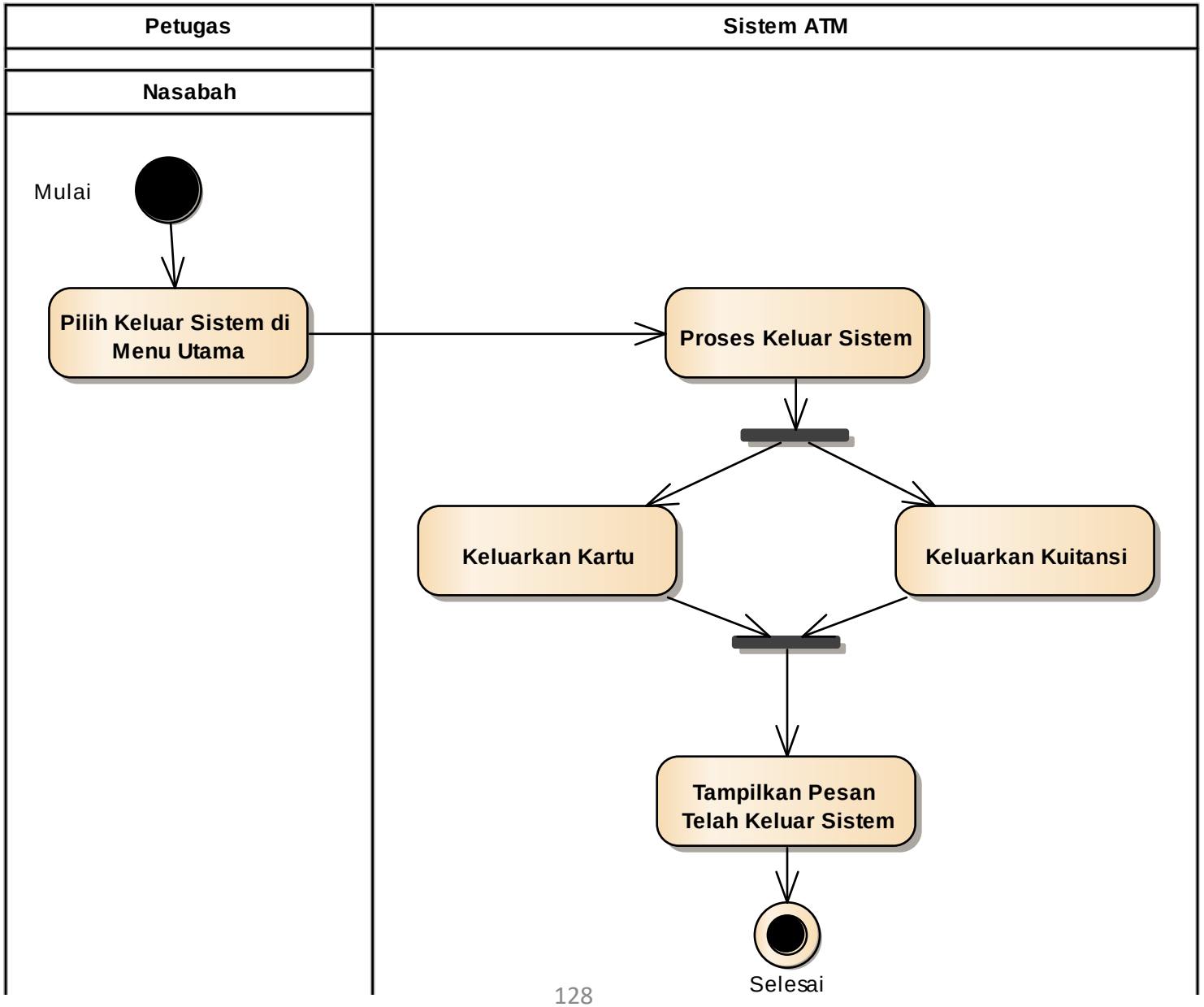
# Activity Diagram: Mengirim Uang



# Activity Diagram: Mengupdate Informasi Kotak Deposit

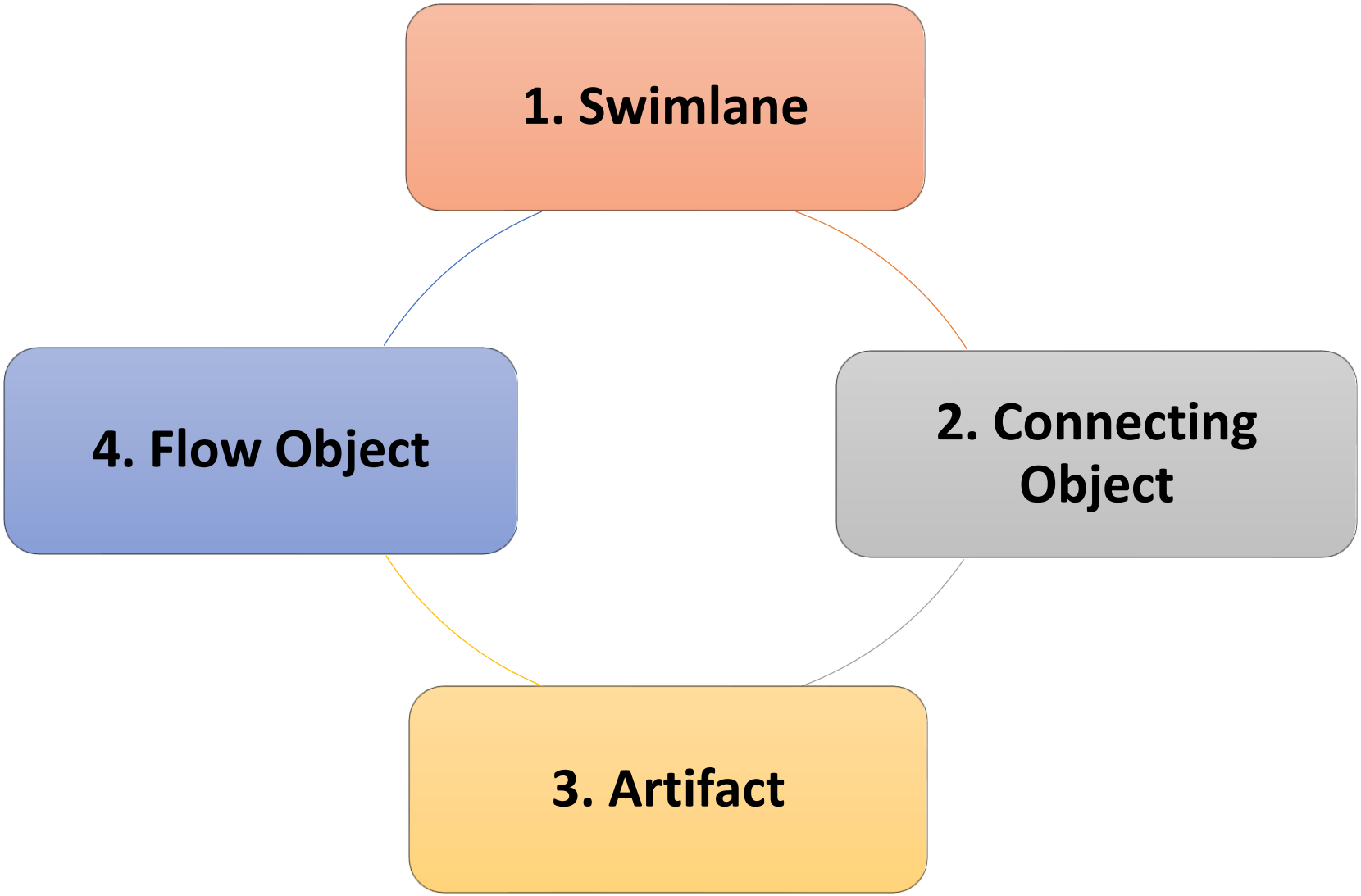


# Activity Diagram: Keluar Sistem



# 2.3.2 Business Process Model and Notation (BPMN)

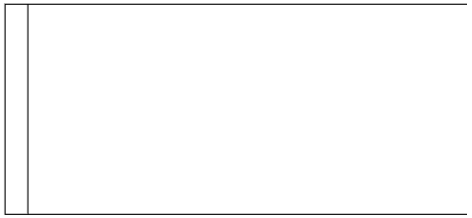
# Elemen BPMN



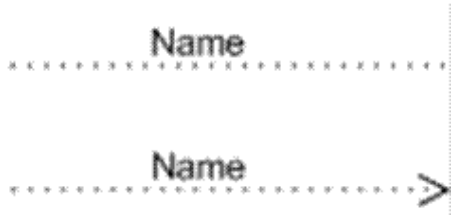
# Elemen dan Notasi BPMN

| ELEMEN | DESKRIPSI | NAMA NOTASI |
|---|---|---|
| **Swimlane** | Mekanisme untuk mengatur dan memisahkan peran atau penanggungjawab dari suatu proses | **Pool** |
| | | **Lane** |
| **Connecting Object** | Konektor dari obyek yang mengalir pada suatu proses | **Sequence Flow** |
| | | **Message Flow** |
| | | **Association** |
| **Artifact** | Informasi tambahan dalam suatu proses | **Annotation** |
| | | **Group** |
| | | **Data Object** |
| | | **Data Store** |
| **Flow Object** | Obyek yang mengalir pada suatu proses | **Event** |
| | | **Activity** |
| | | **Gateway** |

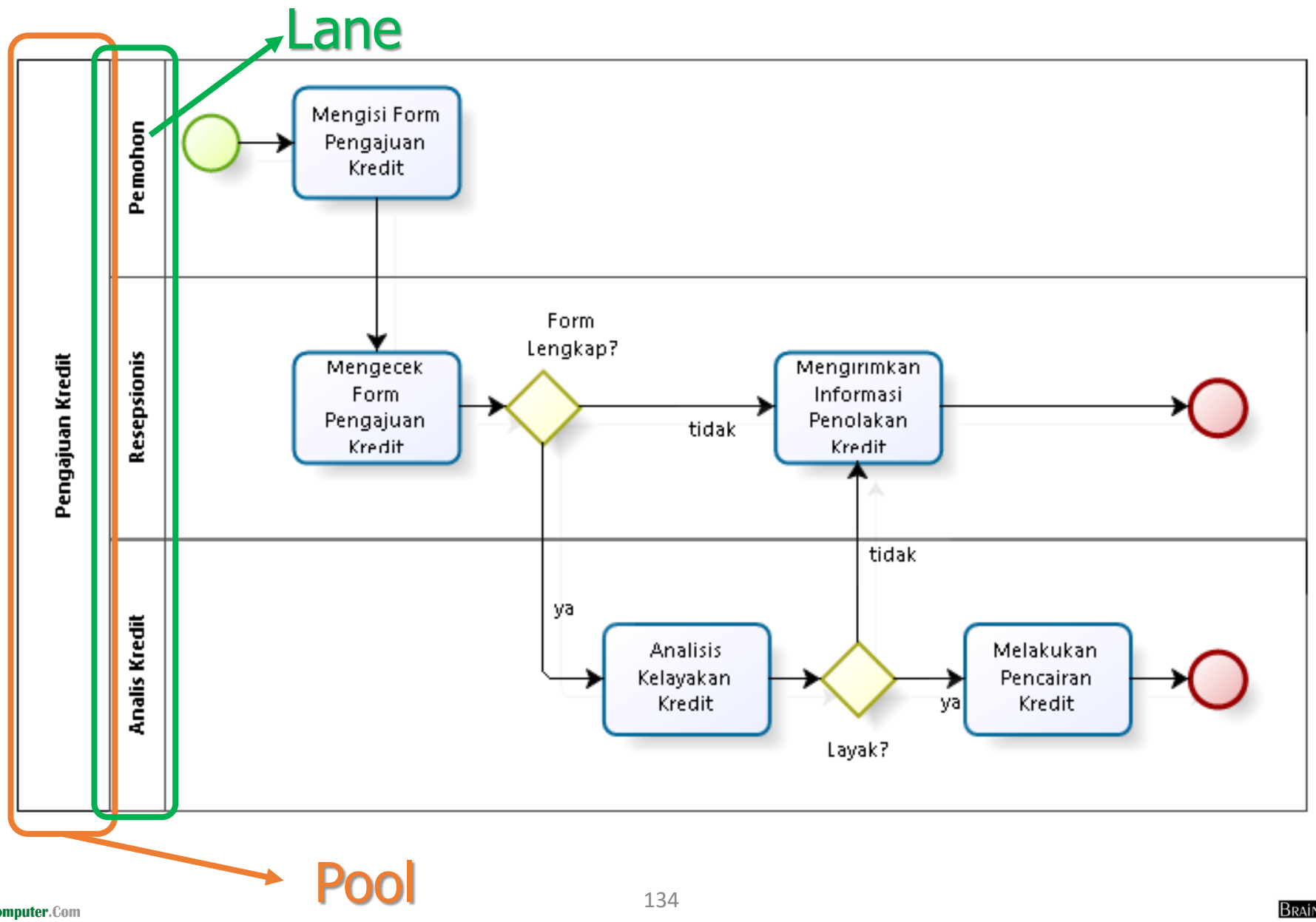
# Swimlane

| Nama Notasi | Deskripsi | Notasi |
|---|---|---|
| **Pool** | Kontainer dari satu proses |  |
| **Lane** | Partisi dari suatu proses, yang menunjukkan sub organisasi, jabatan, peran atau penanggungjawab |  |

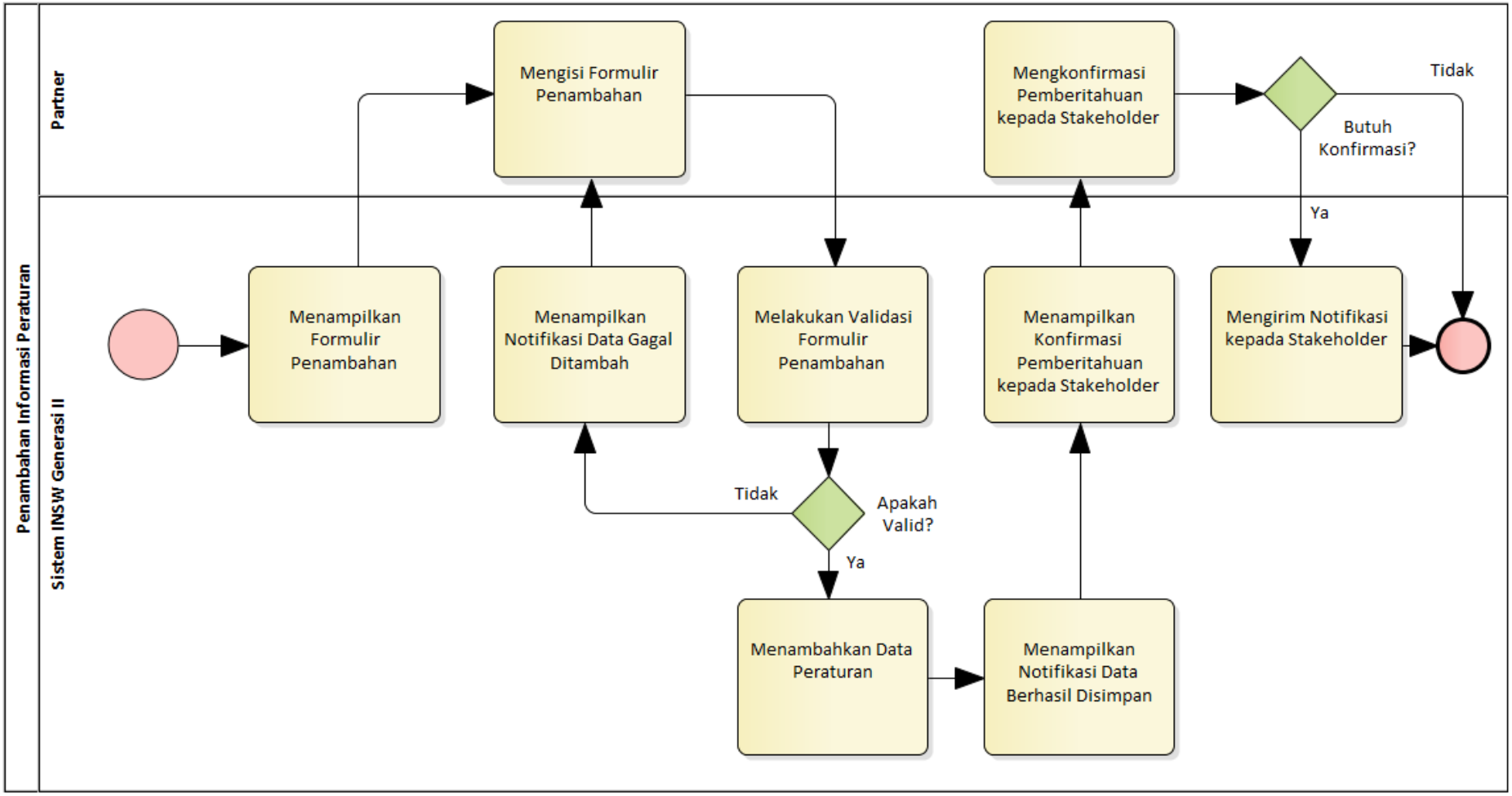
# Connecting Object

| NAMA NOTASI | DESKRIPSI | NOTASI |
|---|---|---|
| **Sequence Flow** | Konektor yang menghubungkan antar obyek yang mengalir dalam satu proses (satu pool) |  |
| **Message Flow** | Konektor yang menghubungkan antar obyek yang mengalir antar proses (beda pool) |   |
| **Association** | Konektor yang menghubungkan obyek yang mengalir ke artifact |   |

# Swimlane - Proses Bisnis Organisasi



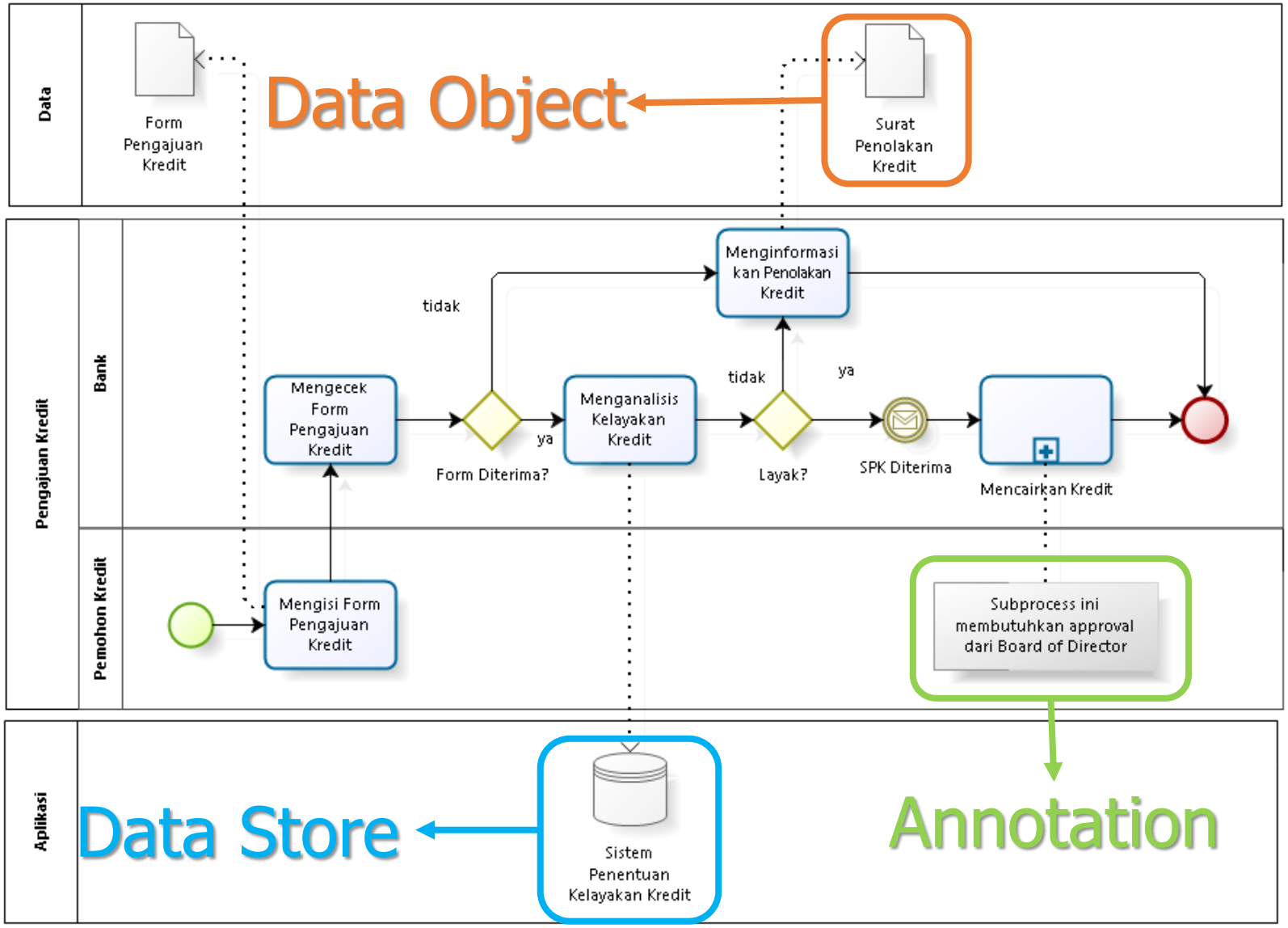
# Swimlane - Proses Bisnis Sistem



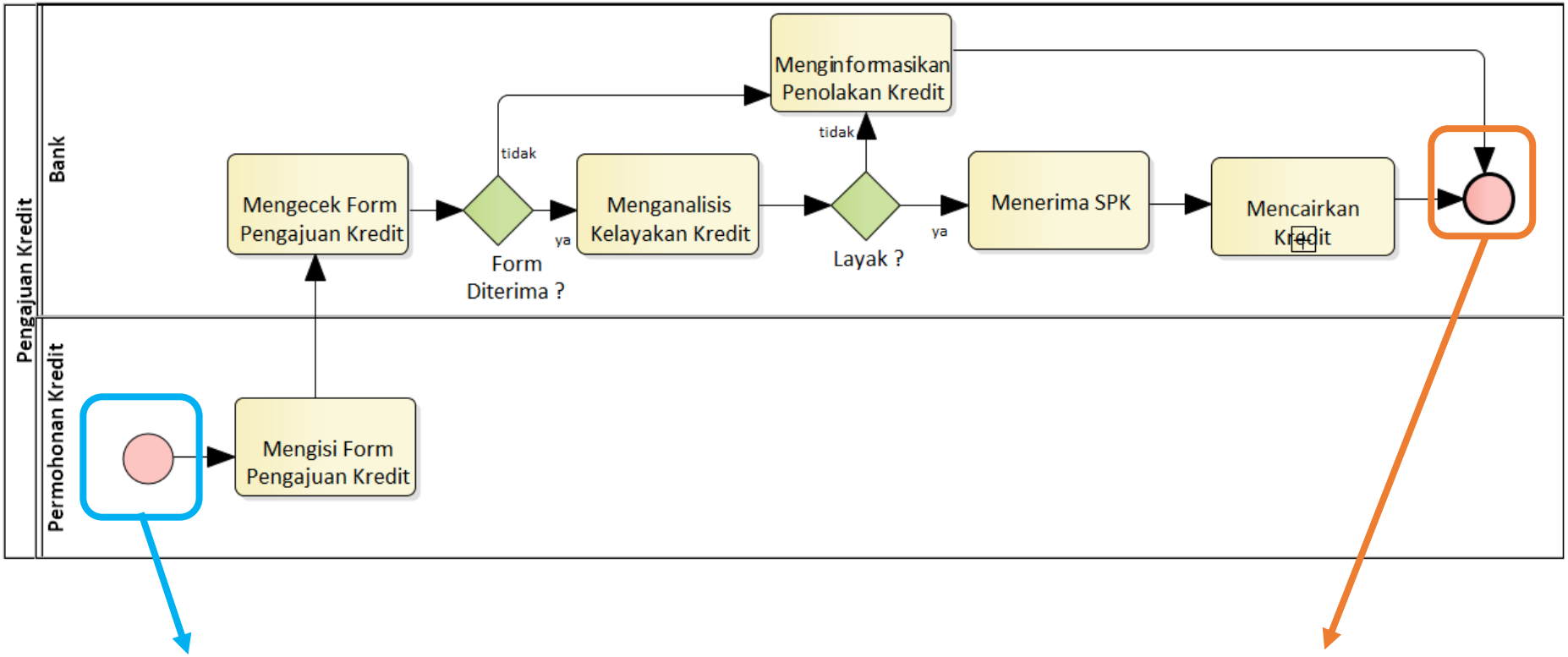
# Artifact

| NAMA NOTASI | DESKRIPSI | NOTASI |
|---|---|---|
| **Annotation** | Penjelasan dari suatu obyek yang mengalir |  |
| **Group** | Pengelompokan dari beberapa obyek yang mengalir |  |
| **Data Object** | File dan dokumen yang digunakan dan dihasilkan oleh suatu aktifitas |  |
| **Data Store** | Sistem dan aplikasi yang digunakan dan dihasilkan oleh suatu aktifitas |  |

# Annotation, Data Object dan Data Store



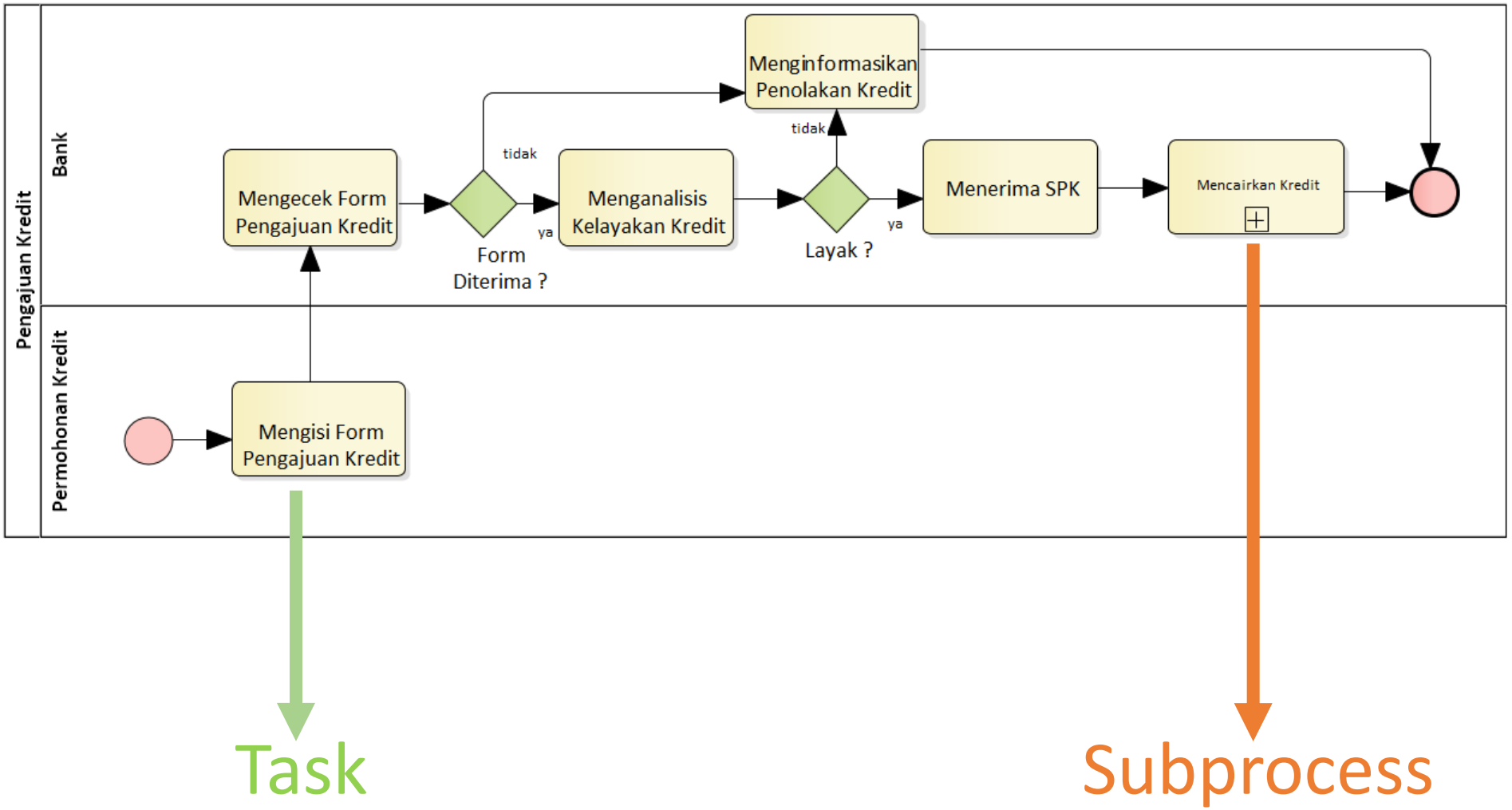
# Event




Start Event

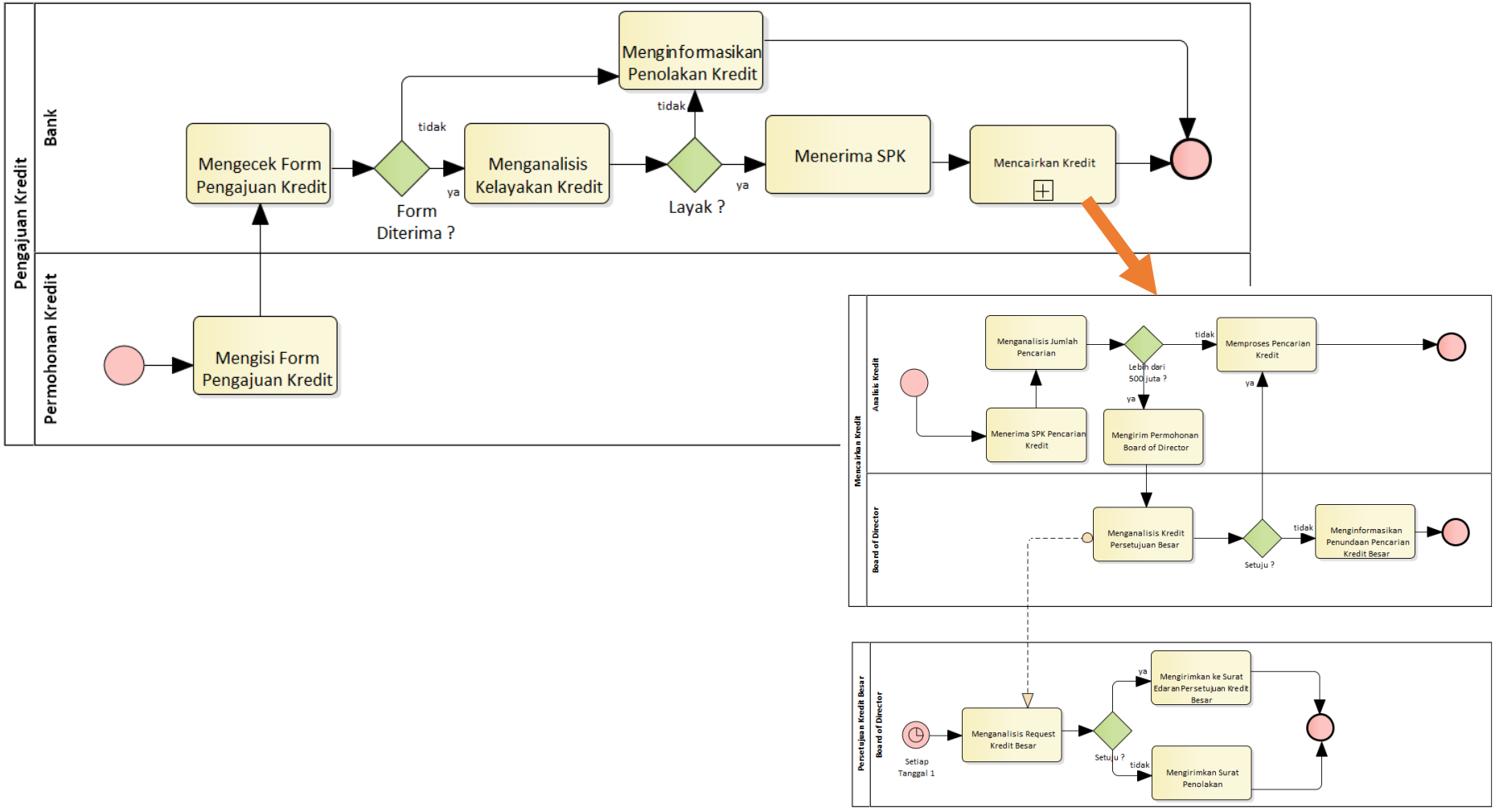
End Event

# Activity


Pengajuan Kredit
Bank
Permohonan Kredit
Menginformasikan Penolakan Kredit
tidak
tidak
Mengecek Form Pengajuan Kredit
Form Diterima ?
ya
Menganalisis Kelayakan Kredit
Layak ?
ya
Menerima SPK
Mencairkan Kredit
Mengisi Form Pengajuan Kredit
Task
Subprocess

# Subprocess

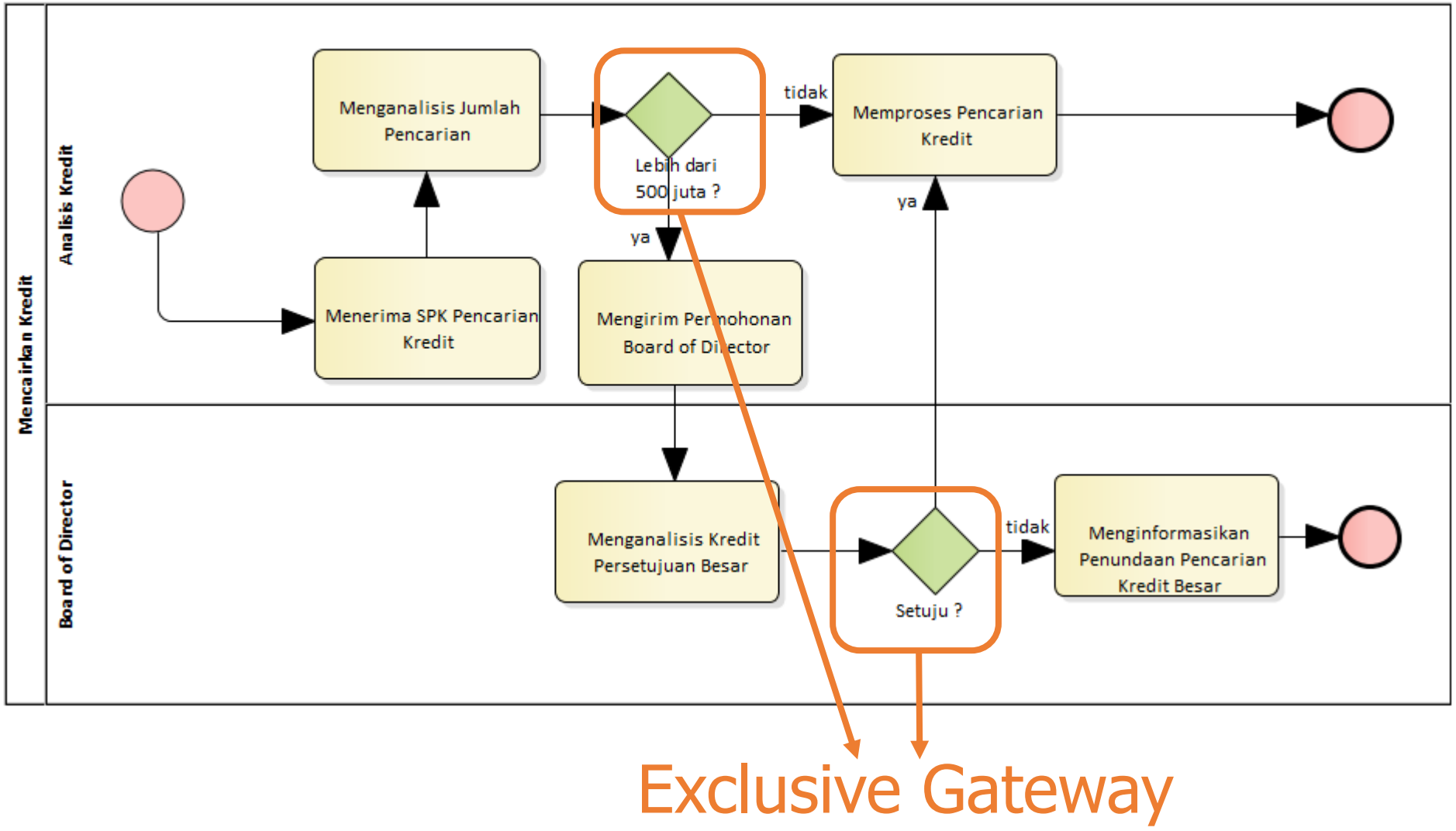
Leveling pada proses bisnis menggunakan BPMN menggunakan subprocess



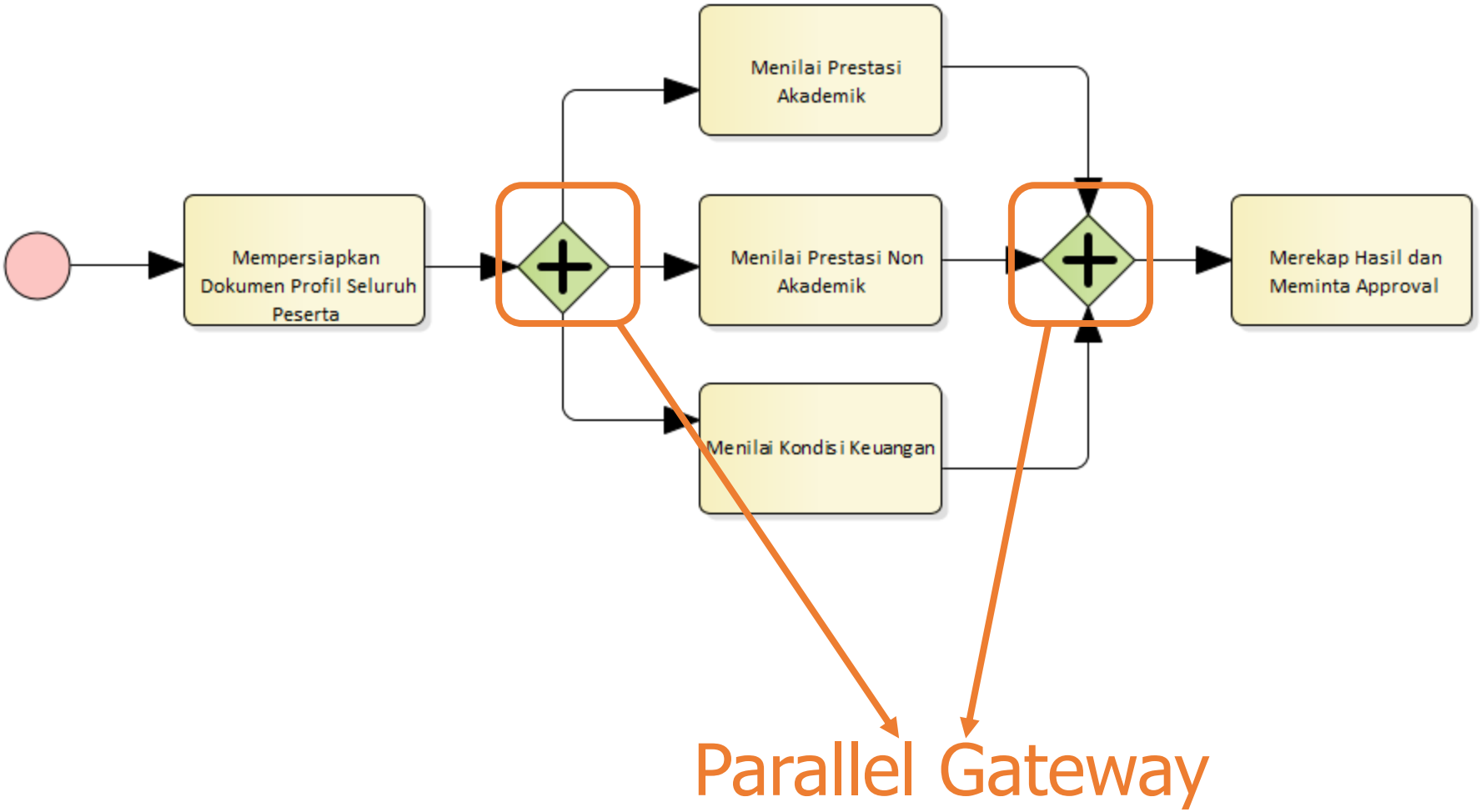
# Gateway

| Nama Notasi | Deskripsi | Notasi |
|---|---|---|
| **Exclusive Gateway** | Pilih salah satu |  |
| **Parallel Gateway** | Kegiatan bersamaan (paralel) dalam satu waktu |  |

# Exclusive Gateway



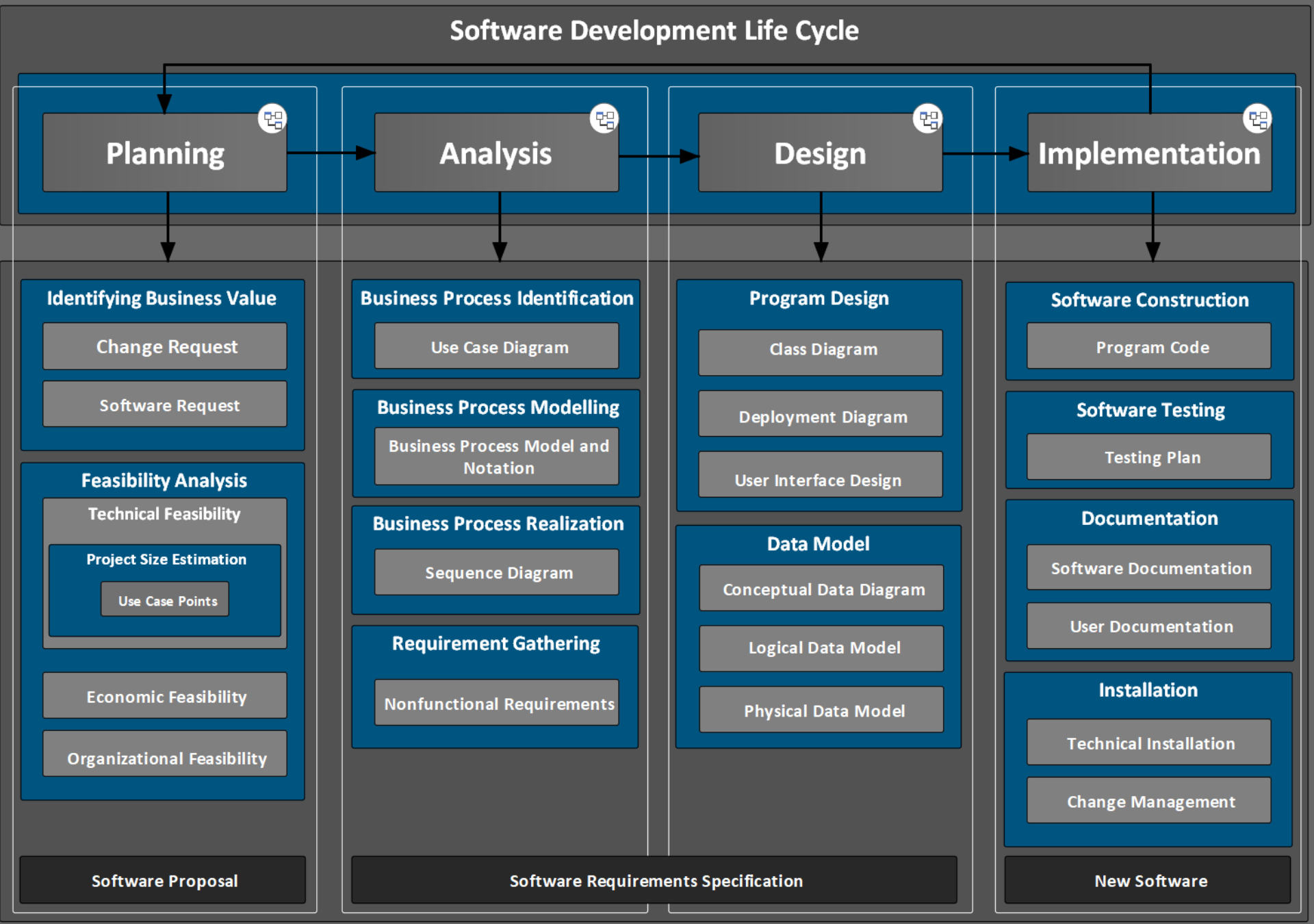
# Parallel Gateway



# Studi Kasus: BPMN MusicPedia

User Interface Diagram langsung digambarkan (prototyping) untuk mempermudah pengguna memahami konteks dari alur proses berjalannya software

# Application Development Governance



# Software Request

| musicpedia | | | | |
|---|---|---|---|---|
| **Date** | 26 Oktober 2018 | | | |
| **Description** | Musicpedia adalah aplikasi layanan download musik & audio dimana saja dan kapan saja, menawarkan akses lengkap ke jutaan lagu dari semua artis papan atas di industri musik barat maupun musik lokal, mendengarkan musik baru dan top musik dunia dalam bentuk audio mp3 secara offline | | | |
| **Project Sponsor** | Wahyu Utomo, VP Business Development, PT Musika Indonesia | | | |
| **Business Need** | 1. Tidak Setuju | 2. Ragu-Ragu | 3. Setuju | 4. Sangat Setuju |
| Aplikasi yang dikembangkan mampu meningkatkan pendapatan perusahaan? | ☐ | ☐ | ☐ | ☑ |
| Aplikasi yang dikembangkan mampu mengurangi biaya operasional perusahaan? | ☐ | ☐ | ☑ | ☐ |
| Aplikasi yang dikembangkan mampu meningkatkan produktifitas kerja pegawai? | ☐ | ☑ | ☐ | ☐ |
| Aplikasi yang dikembangkan mampu meningkatkan nilai tambah perusahaan yang bersifat intangible? | ☐ | ☐ | ☑ | ☐ |
| **Business Value** | Intangible Value:<br>a. Meningkatkan brand recognition perusahaan di dunia internet<br>b. Meningkatkan produktivitas kerja pegawai dan mengurangi kuantitas pegawai<br><br>Tangible Value:<br>a. Mengurangi biaya operasional perusahaan:<br>- Sewa ruangan: Rp120.000.000,-<br>- Biaya komunikasi: Rp6.000.000,-<br>b. Meningkatkan penjualan musik: Rp400.000.000,- | | | |

# *Technical Feasibility*

| musicpedia | | | | |
|---|---|---|---|---|
| **Date: 26 Oktober 2018** | | | | |
| Penjelasan isian | 1. Sangat Kurang | 2. Kurang | 3. Baik | 4. Sangat Baik |

| Kefamiliaran dengan Aplikasi | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Pengguna familiar terhadap pengoperasian aplikasi ini. | ☐ | ☐ | ☑ | ☐ |
| Pengembang familiar terhadap pengembangan aplikasi ini. | ☐ | ☐ | ☐ | ☑ |
| | | | | |
| **Kefamiliaran dengan Teknologi** | **1** | **2** | **3** | **4** |
| Pengguna familiar dengan teknologi pendukung aplikasi. | ☐ | ☑ | ☐ | ☐ |
| Pengembang familiar mengembangkan aplikasi dengan platform, bahasa pemrograman dan tool IDE yang dipilih. | ☐ | ☐ | ☐ | ☑ |
| | | | | |
| **Ukuran Proyek** | | | | |
| Jumlah pengembang yang dibutuhkan. | 7 Man/Month | | | |
| Waktu yang dibutuhkan dalam mengembangkan aplikasi ini. | 6 Month | | | |
| | | | | |
| **Kompatibilitas** | **1** | **2** | **3** | **4** |
| Kebutuhan pengguna terhadap kompatibilitas aplikasi untuk terintegrasi dengan aplikasi lain. | ☐ | ☐ | ☑ | ☐ |
| Kompatibilitas aplikasi terhadap teknologi yang ada pada organisasi. | ☐ | ☐ | ☐ | ☑ |
| | | | | |

| Secara analisi kelayakan teknis, apakah aplikasi layak dikembangkan sesuai kriteria di atas? | ☑ Layak | ☐ Tidak Layak |
|---|---|---|

# *Technical Feasibility*

**Use Case Points**

| Tahap 1 - Menghitung Person Hours (PH) | | | | |
|---|---|---|---|---|
| Use Case Points (UCP) | | Person Hours Multiplier (PHM) | Person Hours (PH) | |
| 51 | | 20 | 1020 | |
| 51 | | 28 | 1428 | |
| **Tahap 2 - Menghitung Person Month (PM)** | | | | |
| PHM | Person Hours (PH) | Lama Bekerja Perhari | Jumlah Bekerja Sebulan | Person Months (PM) |
| 20 | 1020 | 8 | 22 | 5.80 |
| | 1020 | 10 | 26 | 3.92 |
| 28 | 1428 | 8 | 22 | 8.11 |
| | 1428 | 10 | 26 | 5.49 |
| **Tahap 3 - Menghitung Time (Month)** | | | | |
| PHM | Formula Penghitung Waktu | | Jumlah Bekerja Sebulan | Waktu dalam Bulan (M) |
| 20 | 3 * PM^(1/3) | | 22 | 5.39 |
| | | | 26 | 4.73 |
| 28 | | | 22 | 6.03 |
| | | | 26 | 5.29 |

# Economic Feasibility

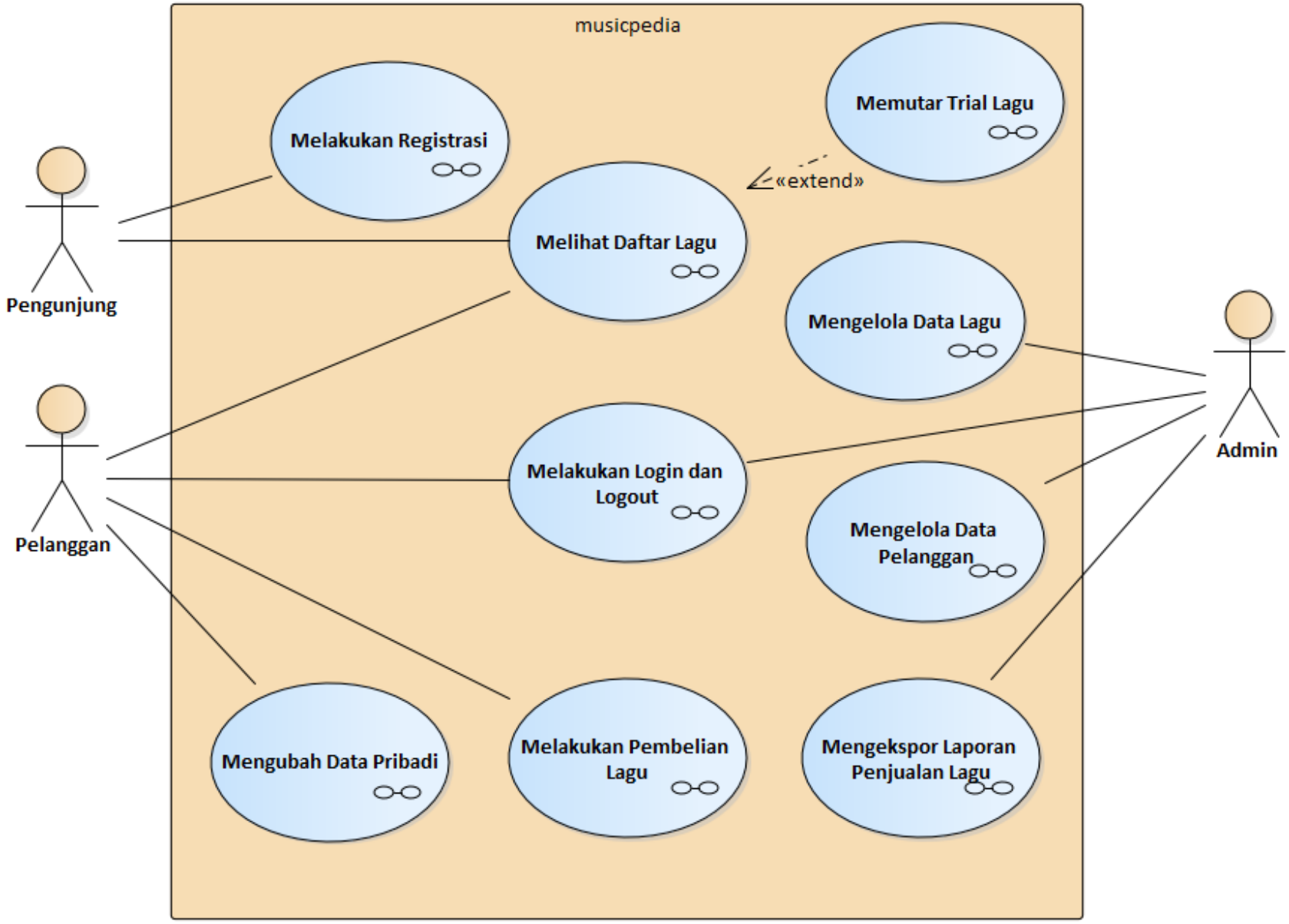
**Cost-Benefit Analysis**

| Tahun | 2019 | 2020 | 2021 | 2022 |
|---|---|---|---|---|
| Peningkatan Pendapatan Penjualan Lagu | | 400,000,000 | 400,000,000 | 400,000,000 |
| Pengurangan Biaya Sewa Ruangan | | 120,000,000 | 120,000,000 | 120,000,000 |
| Pengurangan Biaya Komunikasi | | 6,000,000 | 6,000,000 | 6,000,000 |
| **Total Benefits** | **0** | **526,000,000** | **526,000,000** | **526,000,000** |
| **PV of Benefits** | **0** | **468,138,127** | **441,639,743** | **416,641,267** |
| **PV of All Benefits** | **0** | **468,138,127** | **909,777,870** | **884,779,394** |
| Honor Tim (Analysis, Design and Implementation) | 250,000,000 | 120,000,000 | 120,000,000 | 120,000,000 |
| | | | | |
| **Total Development Costs** | **250,000,000** | **120,000,000** | **120,000,000** | **120,000,000** |
| Honor Pengelola Web | 72,000,000 | 72,000,000 | 72,000,000 | 72,000,000 |
| Biaya Lisensi Software | 10,000,000 | 10,000,000 | 10,000,000 | 10,000,000 |
| Hardware upgrades | 50,000,000 | 50,000,000 | 50,000,000 | 50,000,000 |
| Biaya Komunikasi | 1,000,000 | 1,000,000 | 1,000,000 | 1,000,000 |
| Biaya Marketing | 50,000,000 | 50,000,000 | 50,000,000 | 50,000,000 |
| **Total Operational Costs** | **183,000,000** | **183,000,000** | **183,000,000** | **183,000,000** |
| **Total Costs** | **433,000,000** | **303,000,000** | **303,000,000** | **303,000,000** |
| **PV of Costs** | **408,490,566** | **269,668,921** | **153,650,329** | **144,953,140** |
| **PV of all Costs** | **408,490,566** | **678,159,487** | **831,809,816** | **976,762,957** |
| **Total Project Costs Less Benefits** | **-433,000,000** | **223,000,000** | **223,000,000** | **223,000,000** |
| **Yearly NPV** | **-408,490,566** | **198,469,206** | **187,235,100** | **176,636,887** |
| **Cumulative NPV** | **-408,490,566** | **-210,021,360** | **-22,786,260** | **153,850,627** |
| **Return on Investment (ROI)** | **-100.00%** | **-0.309693168** | **-0.027393593** | **0.15751071** |
| **Break-even Point (BEP)** | | | | **3.129000574** |

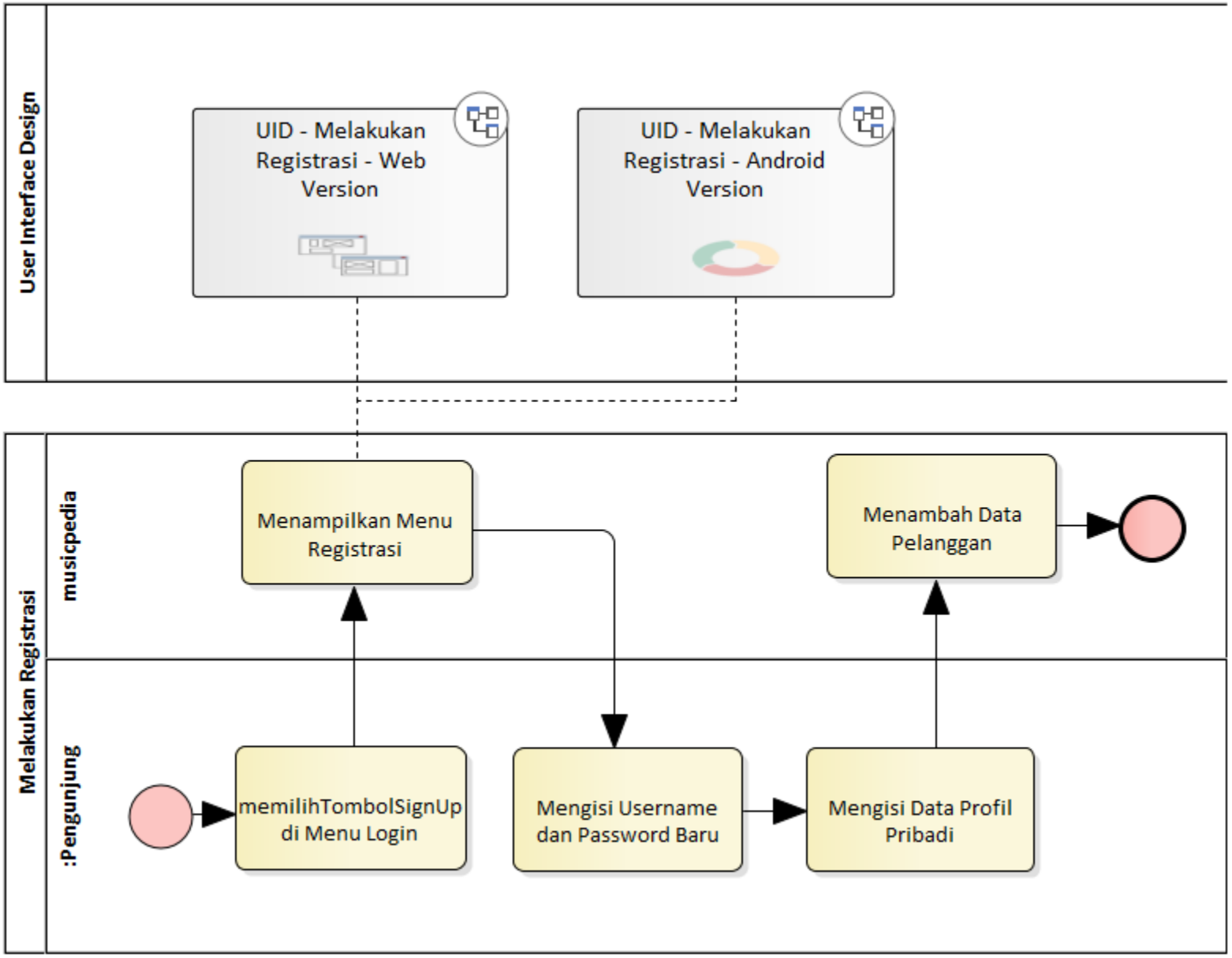
# *Organizational Feasibility*

| musicpedia | | | |
|---|---|---|---|
| **Date** | **26 Oktober 2018** | | |
| **Anggota Tim** | | | |
| User/Product Owner | Wahyu Utomo | | |
| Project Manager | Haris Dermawan | | |
| System Analyst | Risa Dhani Horasman Purba | | |
| Business Analyst | Mulyana | | |
| Programmer | Achmad Fatkarrofiqi | | |
| Tester | Januar Sapareza | | |
| **Apakah aplikasi ini mendukung visi dan misi organisasi?** | | | |
| Ya | | | |
| **Apakah aplikasi ini sesuai dengan tugas, fungsi dan KPI unit kerja anda?** | | | |
| Ya | | | |
| **Apakah aplikasi ini selaras dengan proses bisnis unit kerja anda?** | | | |
| Ya | | | |
| Secara analisis kelayakan organisasi, apakah aplikasi layak dikembangkan sesuai kriteria di atas? | | ☑ Layak | ☐ Tidak Layak |

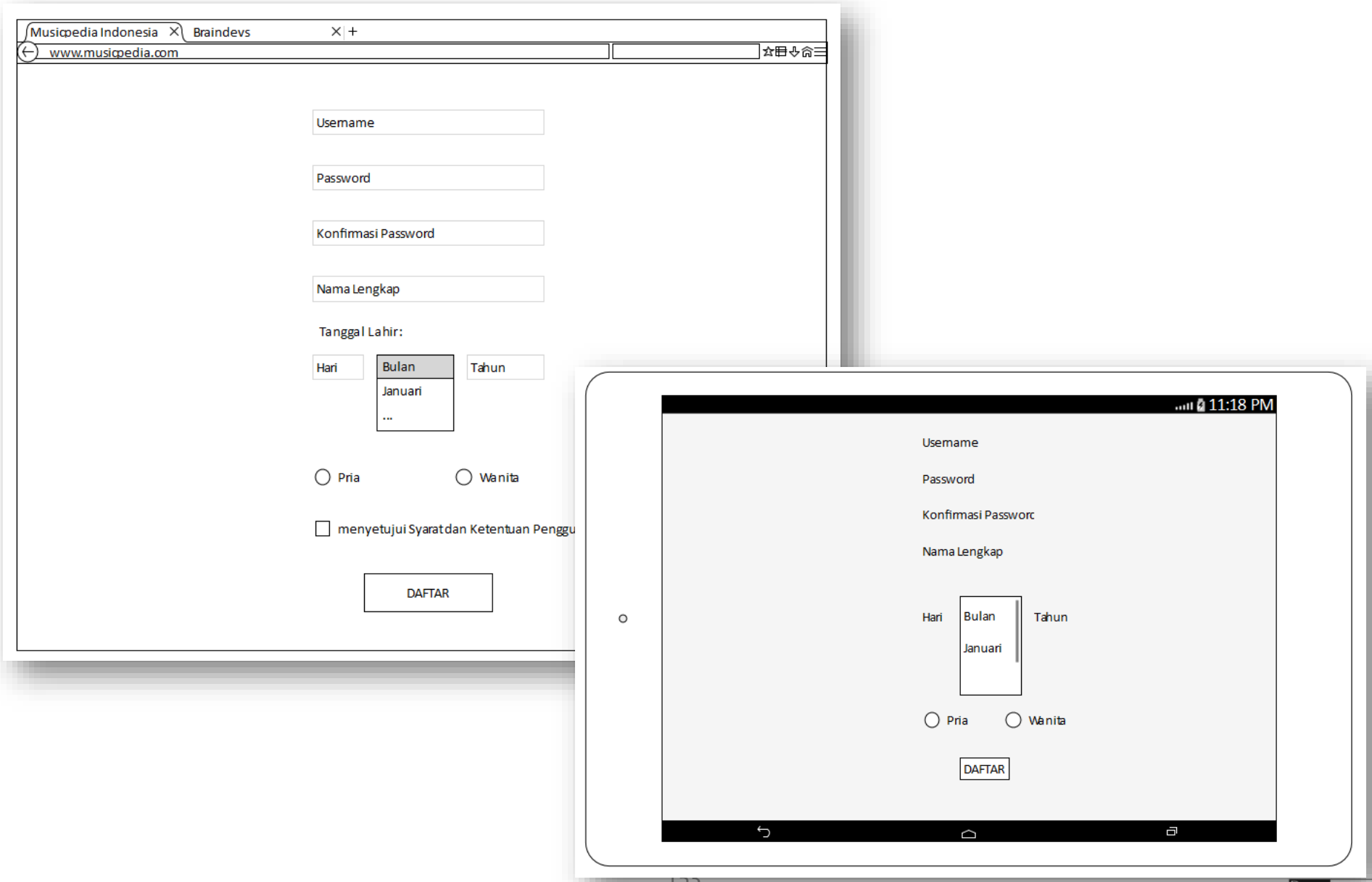
# Use Case Diagram MusicPedia



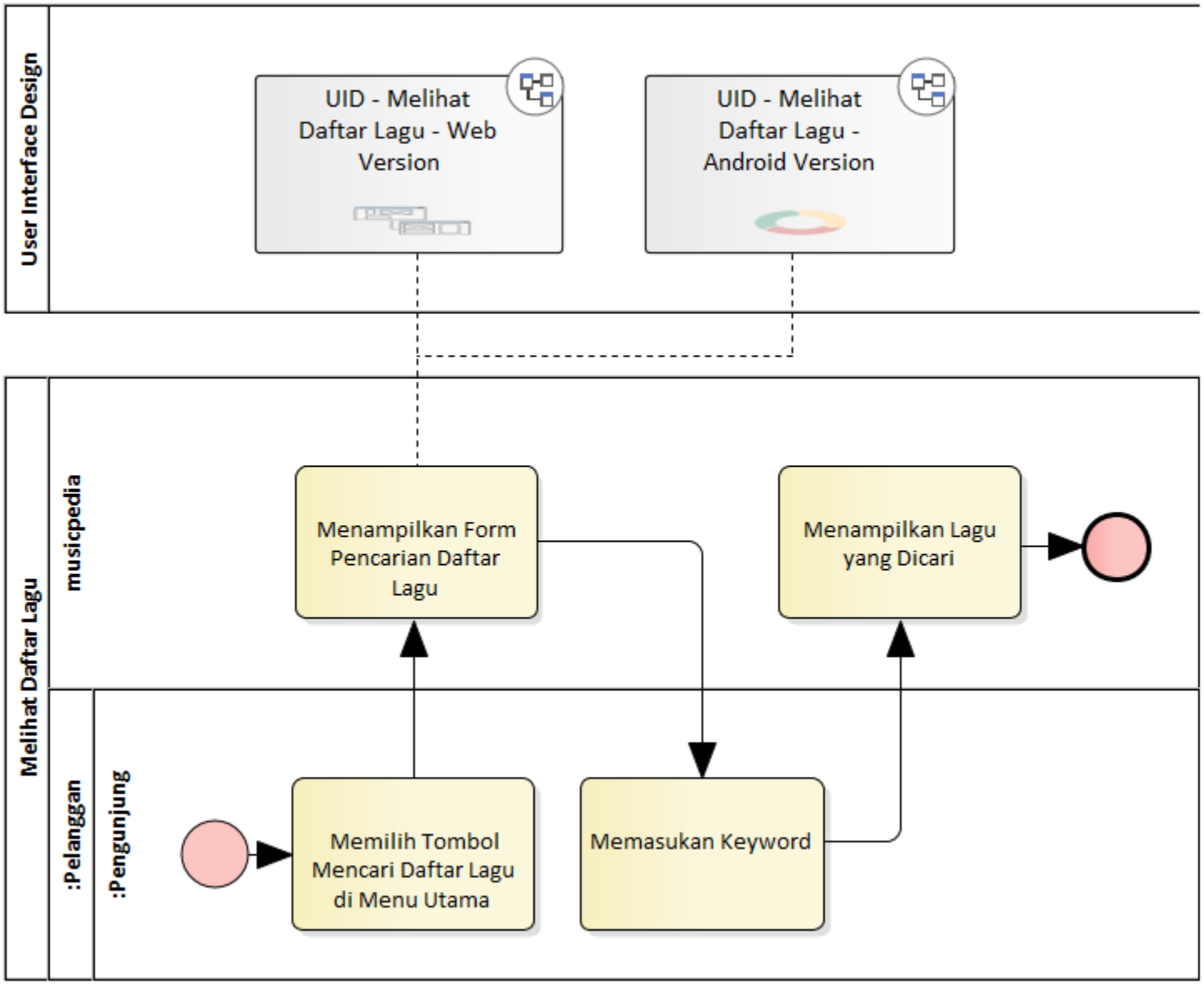
# BPMN Melakukan Registrasi



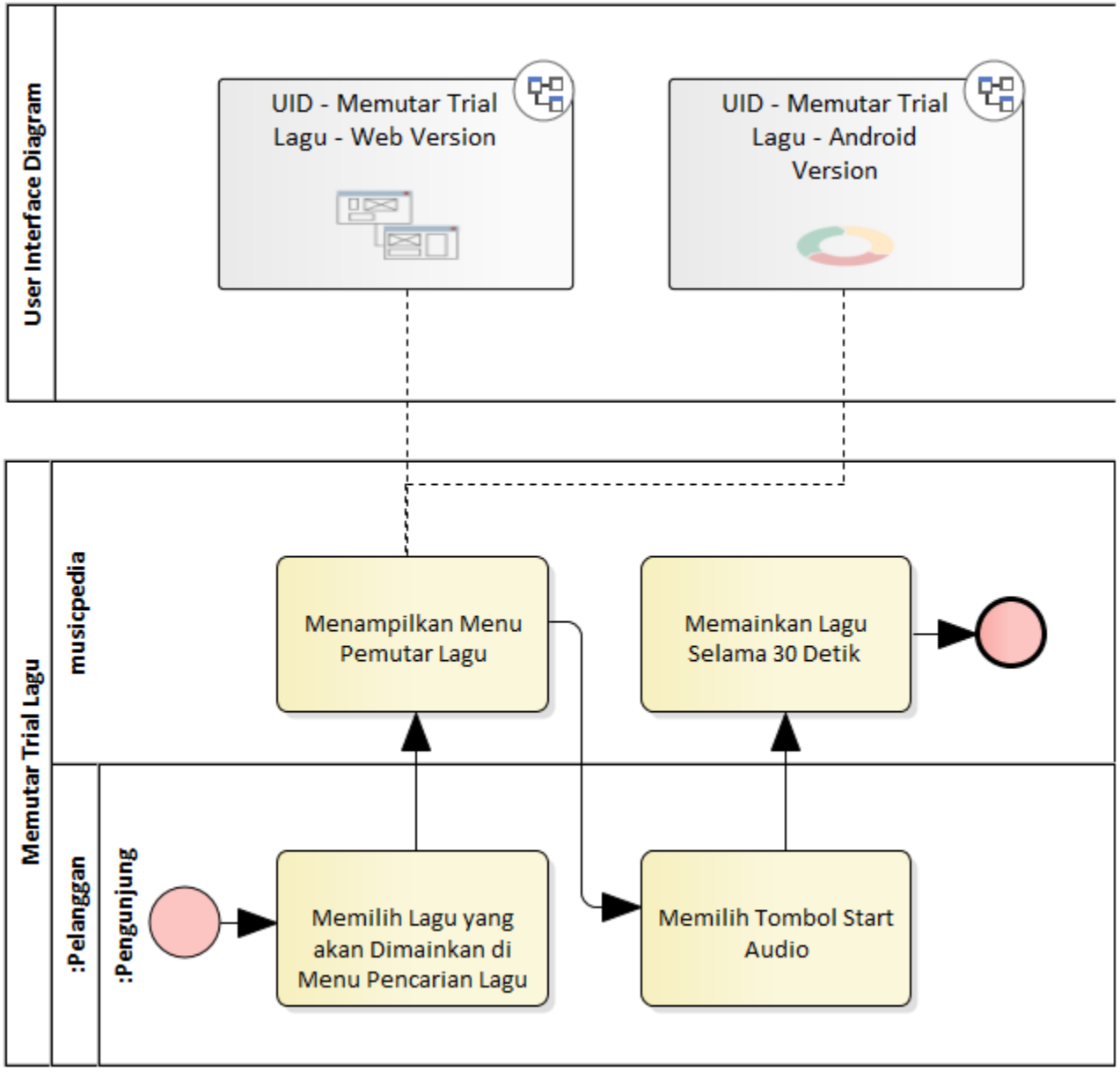
# User Interface Design Melakukan Registrasi (versi Web dan versi Android)



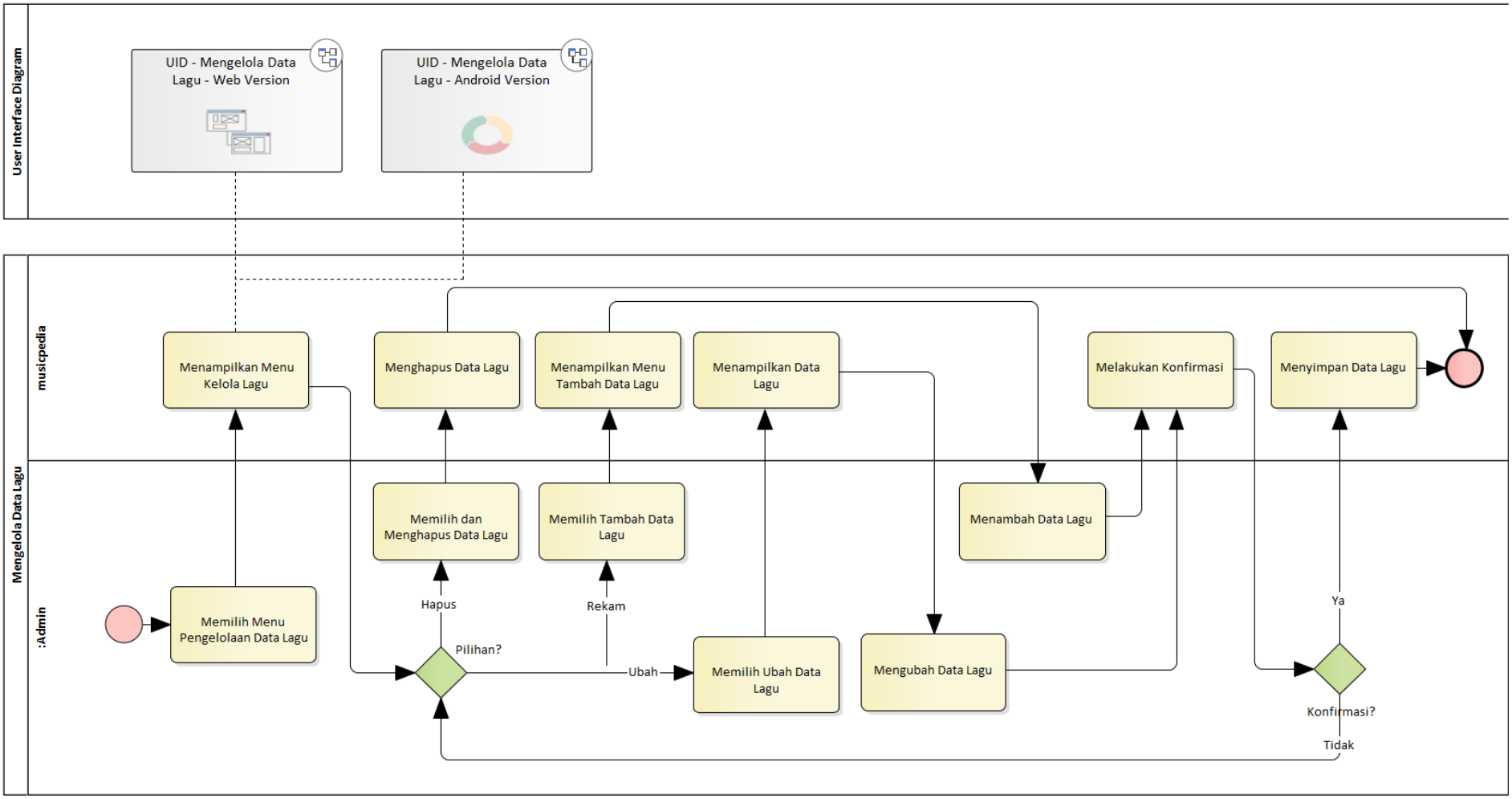
# BPMN Melihat Daftar Lagu



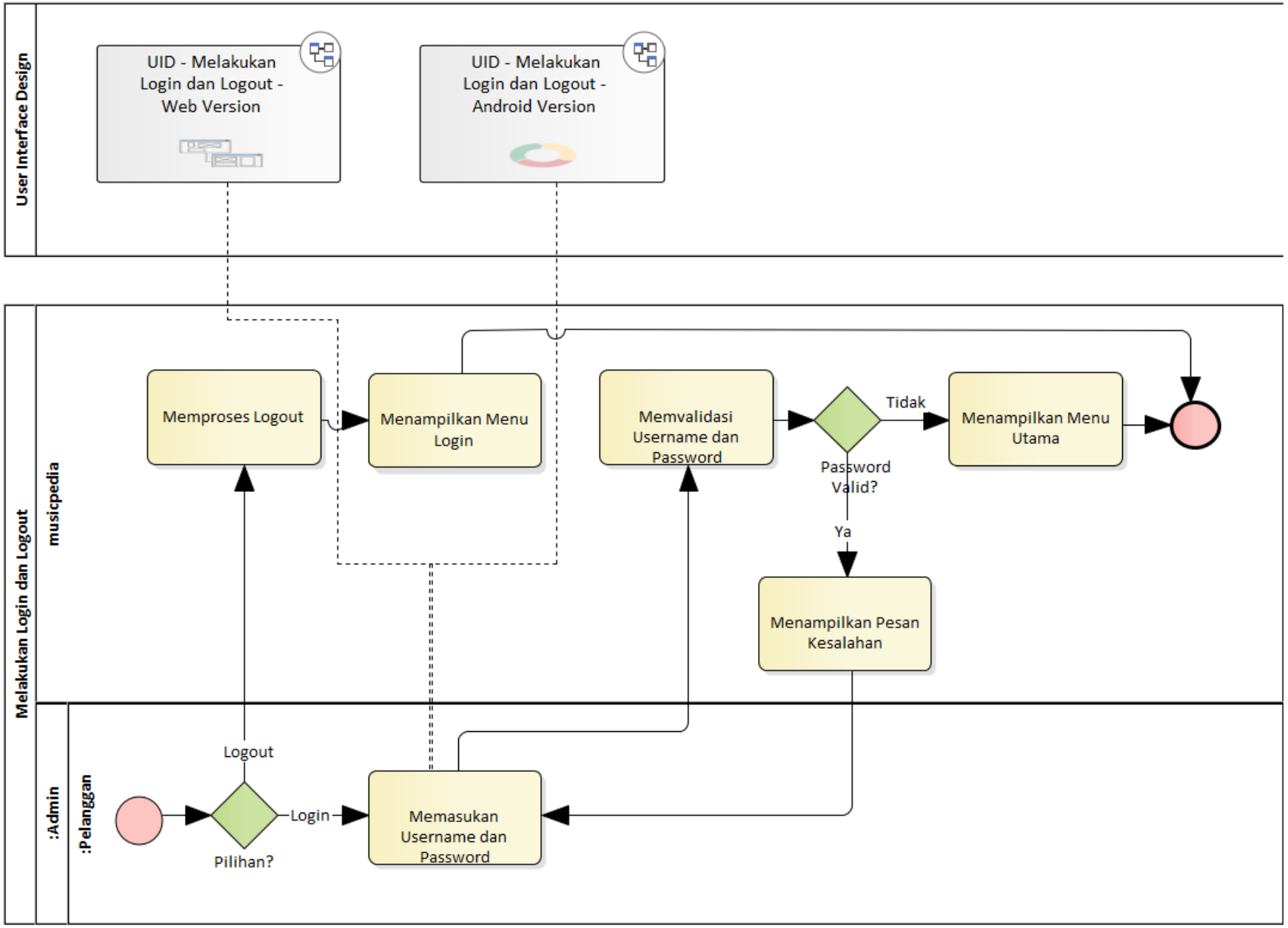
# BPMN Memutar Trial Lagu



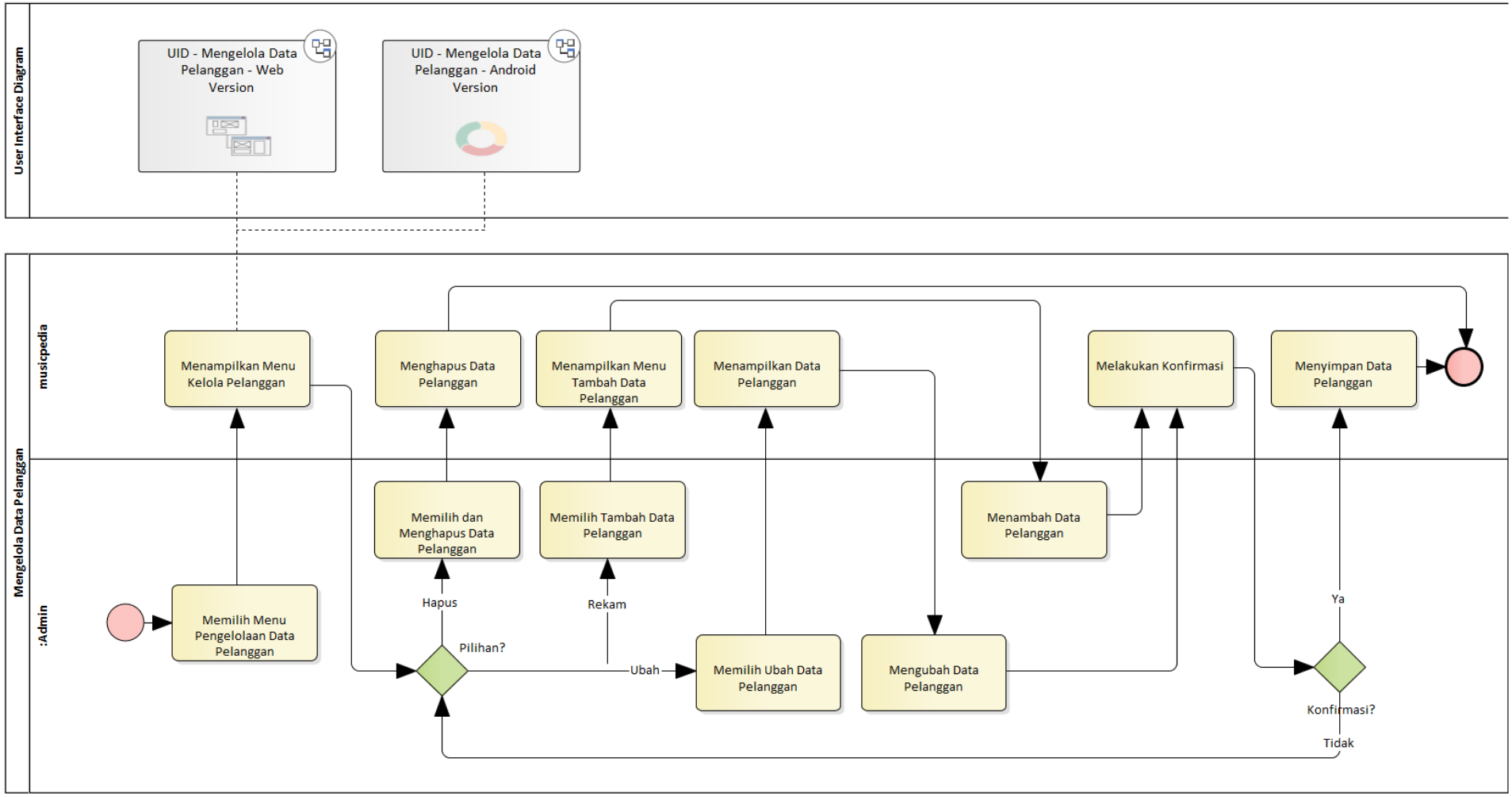
# BPMN Mengelola Data Lagu



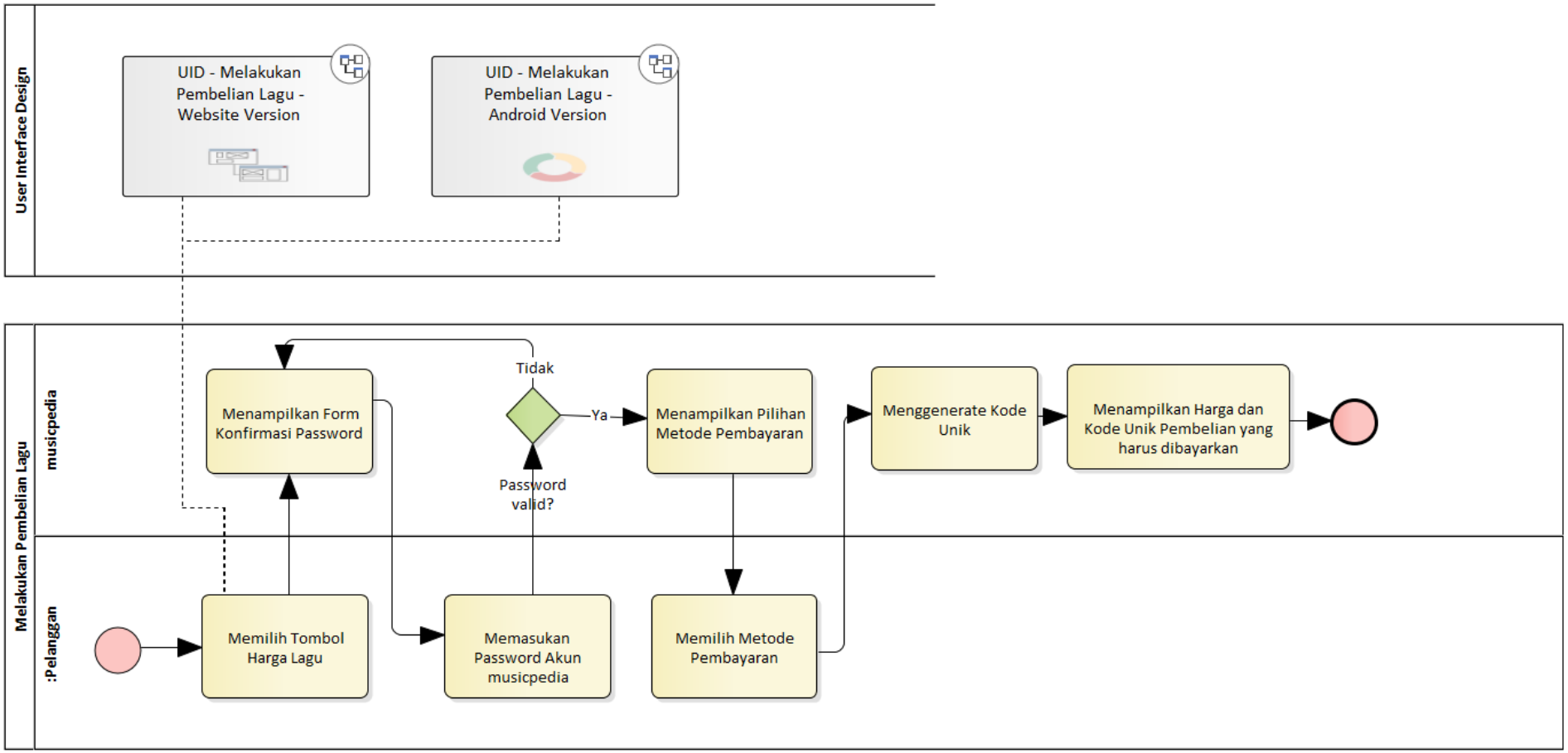
# BPMN Melakukan Login dan Logout



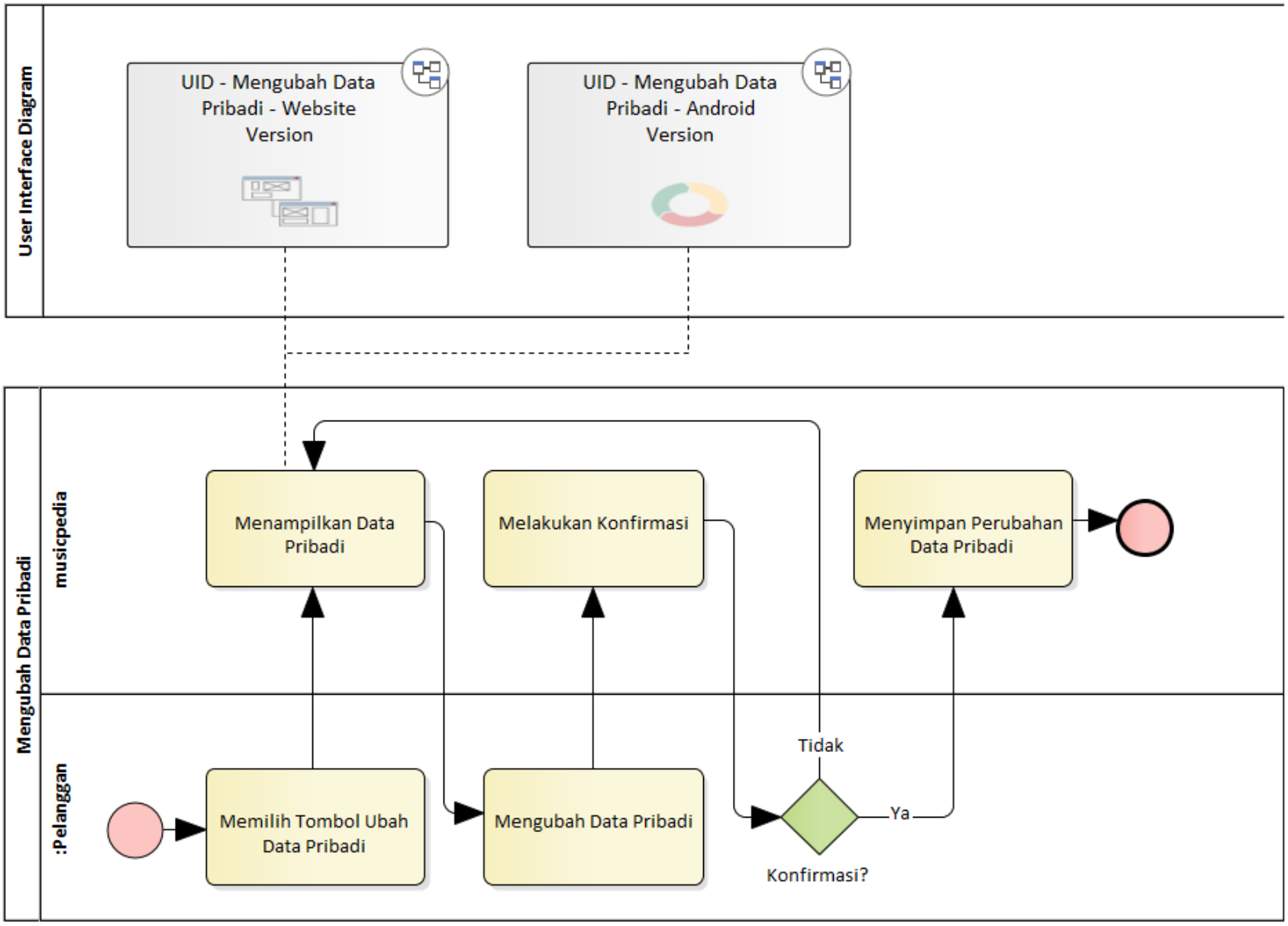
# BPMN Mengelola Data Pelanggan



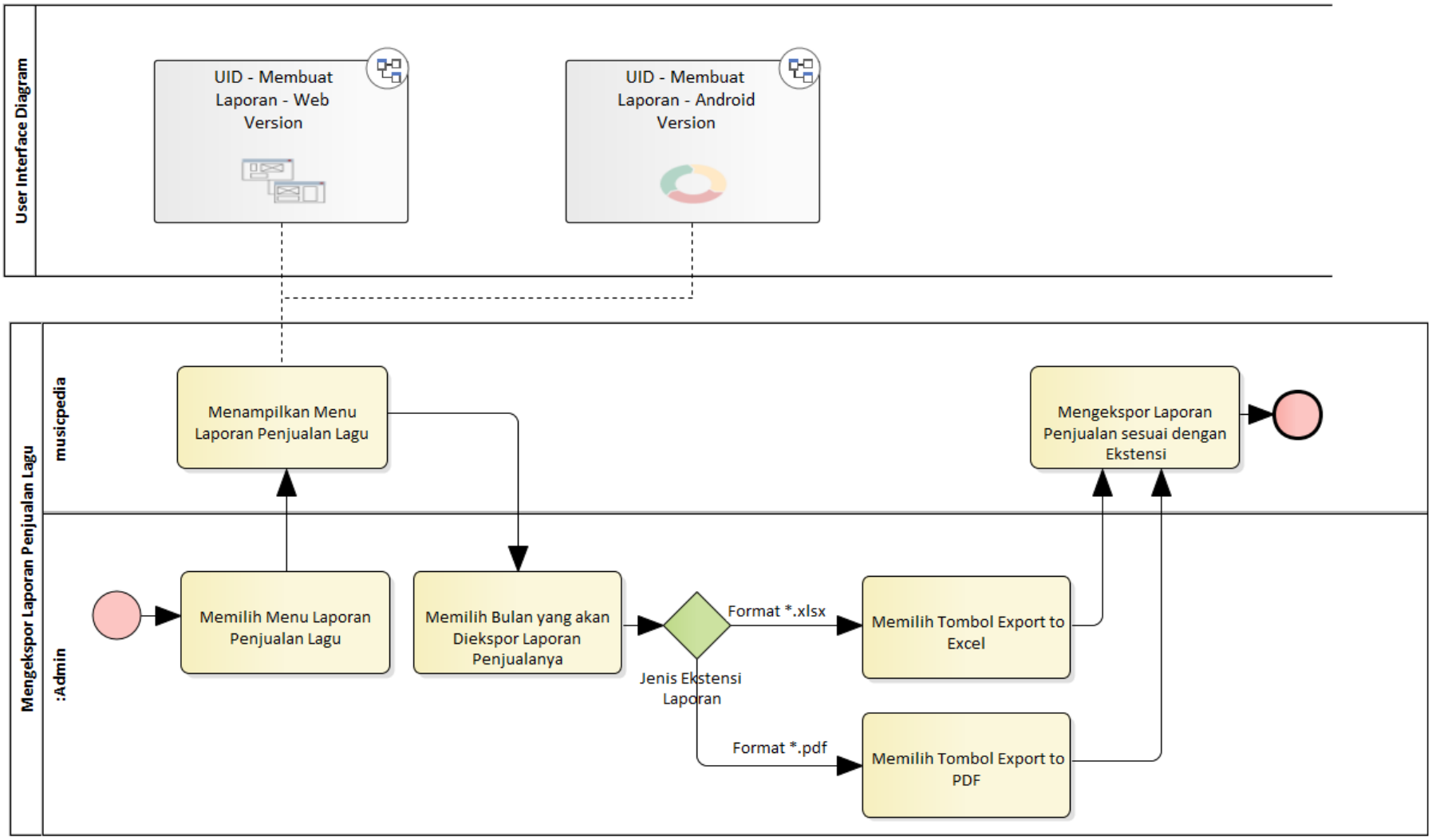
# BPMN Melakukan Pembelian lagu



# BPMN Mengubah Data Pribadi



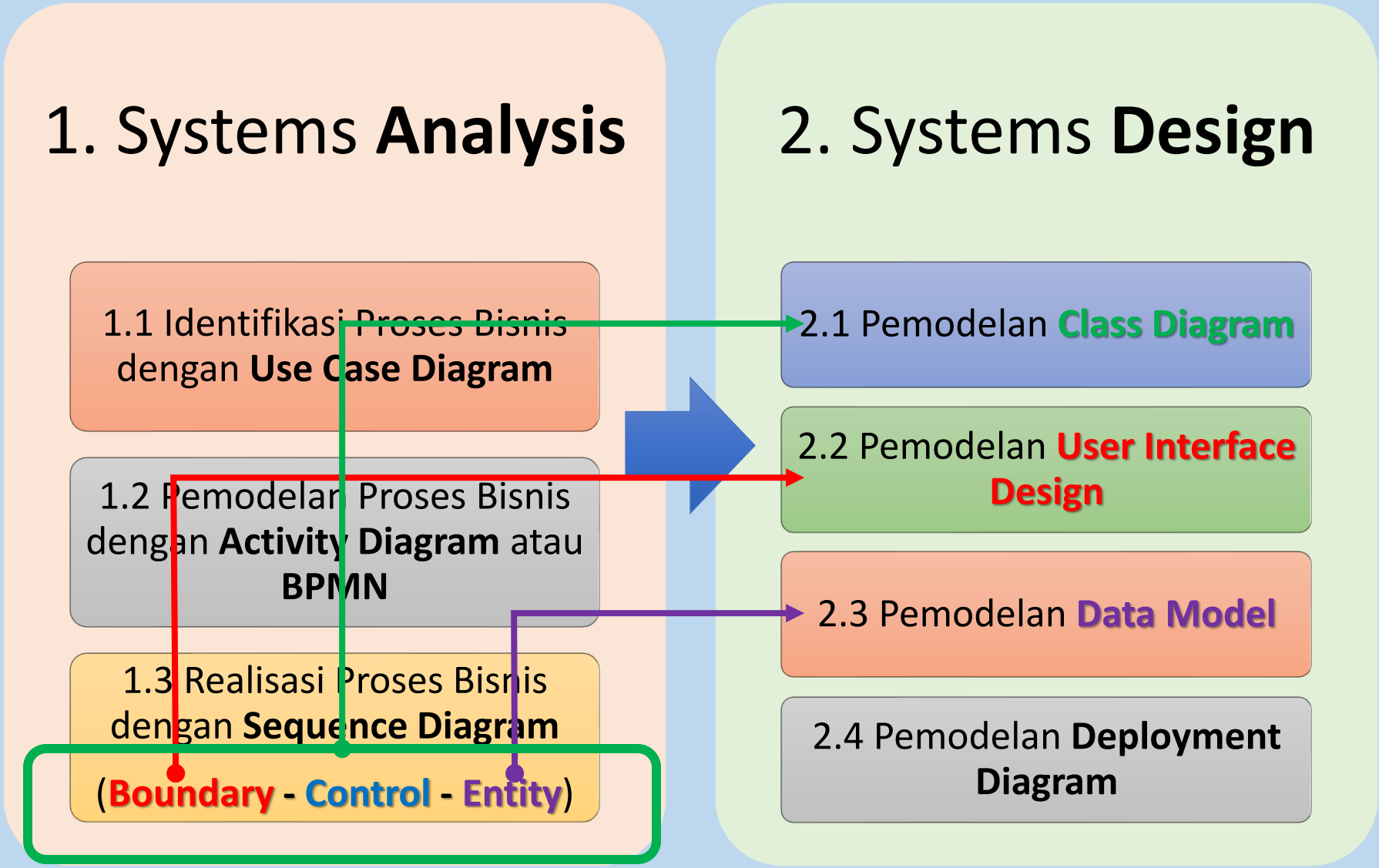
# BPMN Mengekspor Laporan Penjualan Lagu



# 2.4 Realisasi Proses Bisnis dengan Sequence Diagram

# UML based Software Analysis and Design

*(Wahono, 2009)*

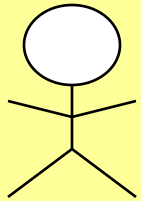
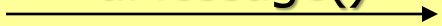


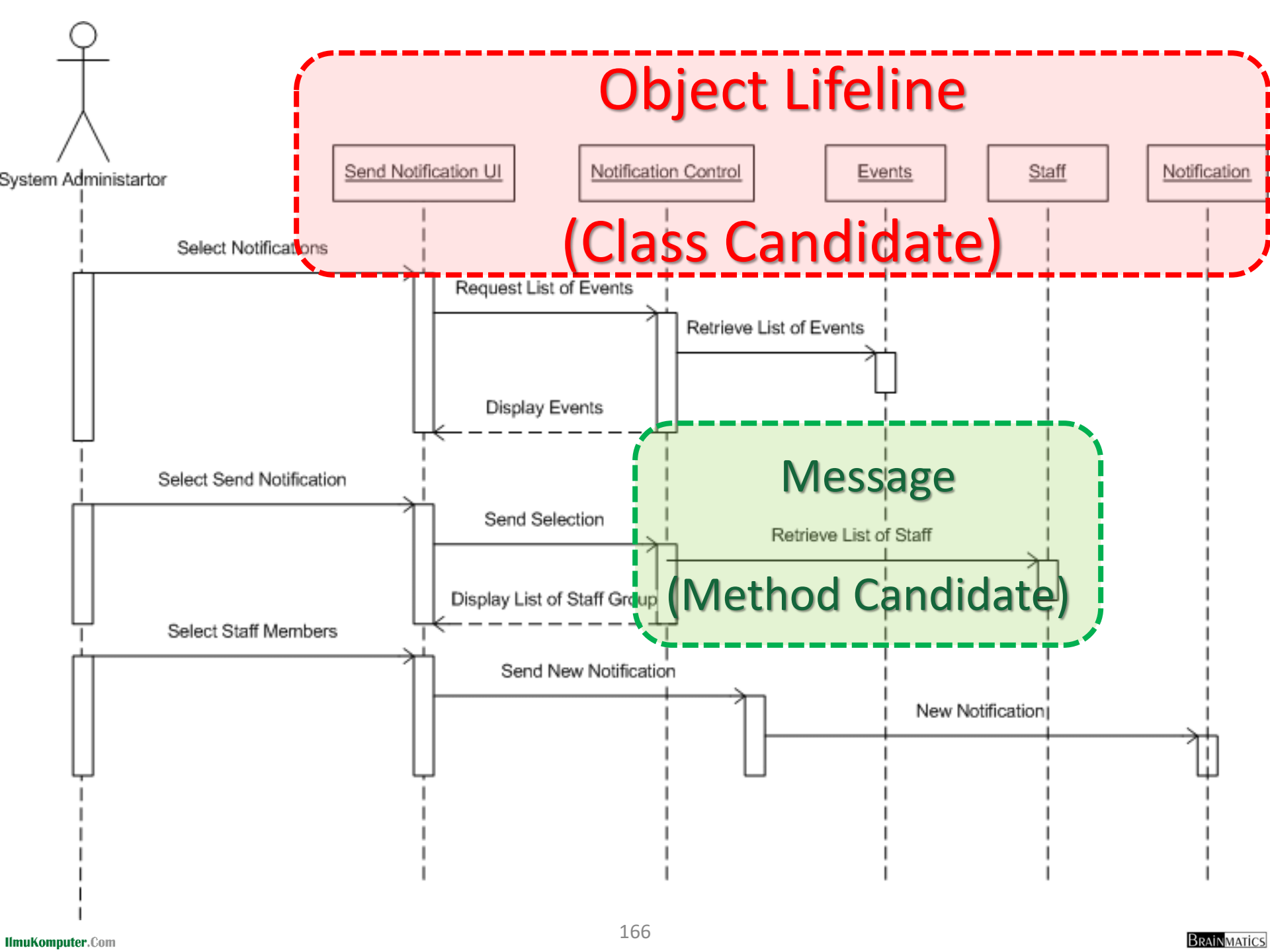
# Sequence Diagram

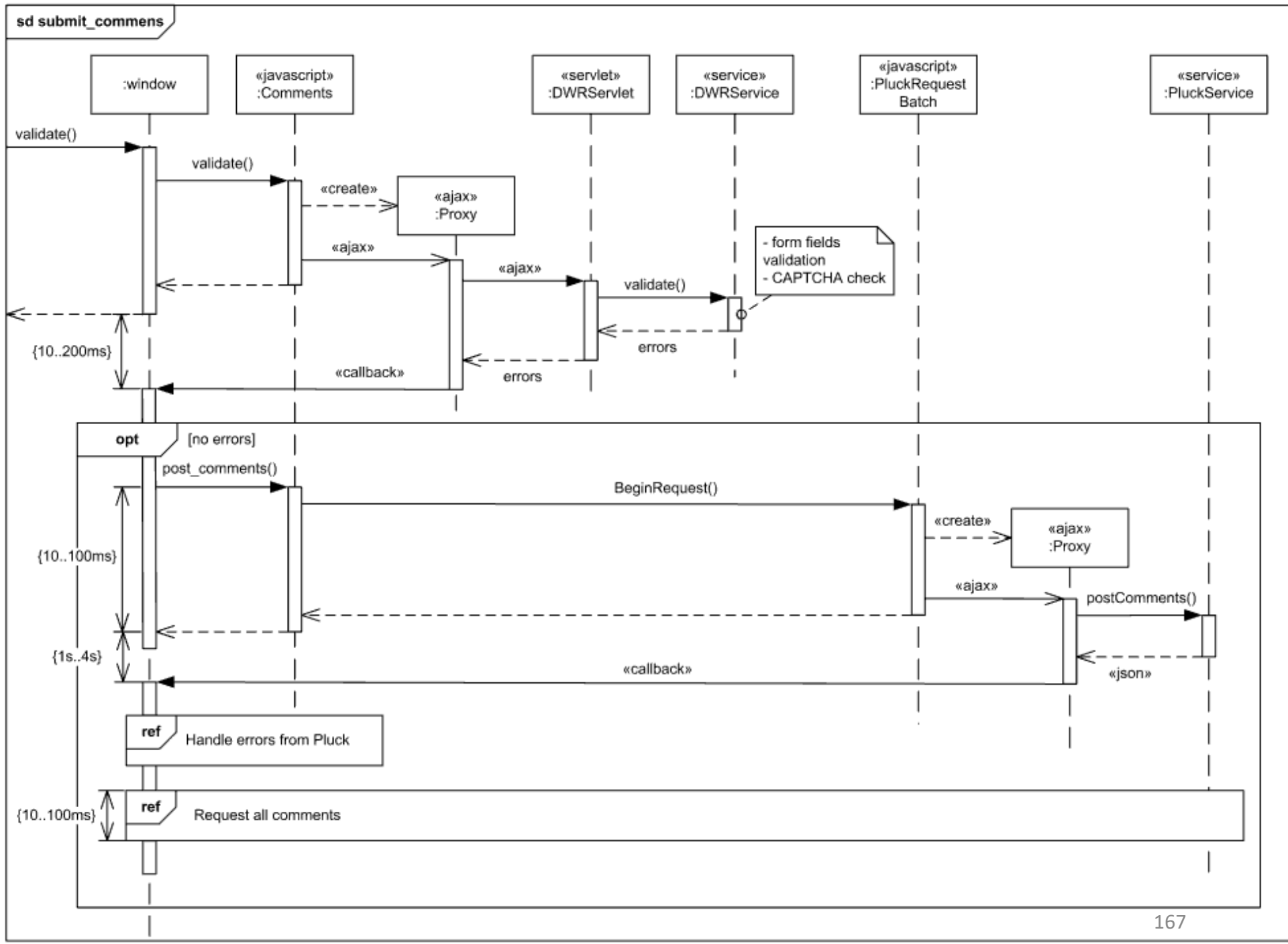
- Sequence diagram menggambarkan interaksi antar *object* (*class*) dalam sofware pada suatu sekuensial waktu
  - Interaksi object (class) berdasarkan alur proses bekerjanya software, dimana interaksi tersebut menggambarkan pesan yang dikirimkan secara sekuensial antara object (class)
  - Garis vertikal (lifeline) menunjukkan object (class), garis horizontal menunjukkan pesan yang mengalir antara object (class) tersebut

- Sequence diagram adalah alat komunikasi System Analyst dengan Programmer, menggambarkan alur proses bekerjanya software sekaligus dengan komposisi software akan seperti apa

# Sequence Diagram Syntax

| | |
|---|---|
| AN ACTOR |  |
| AN OBJECT | anObject:aClass |
| A LIFELINE |  |
| A FOCUS OF CONTROL |  |
| A MESSAGE | aMessage()  |
| OBJECT DESTRUCTION | X |

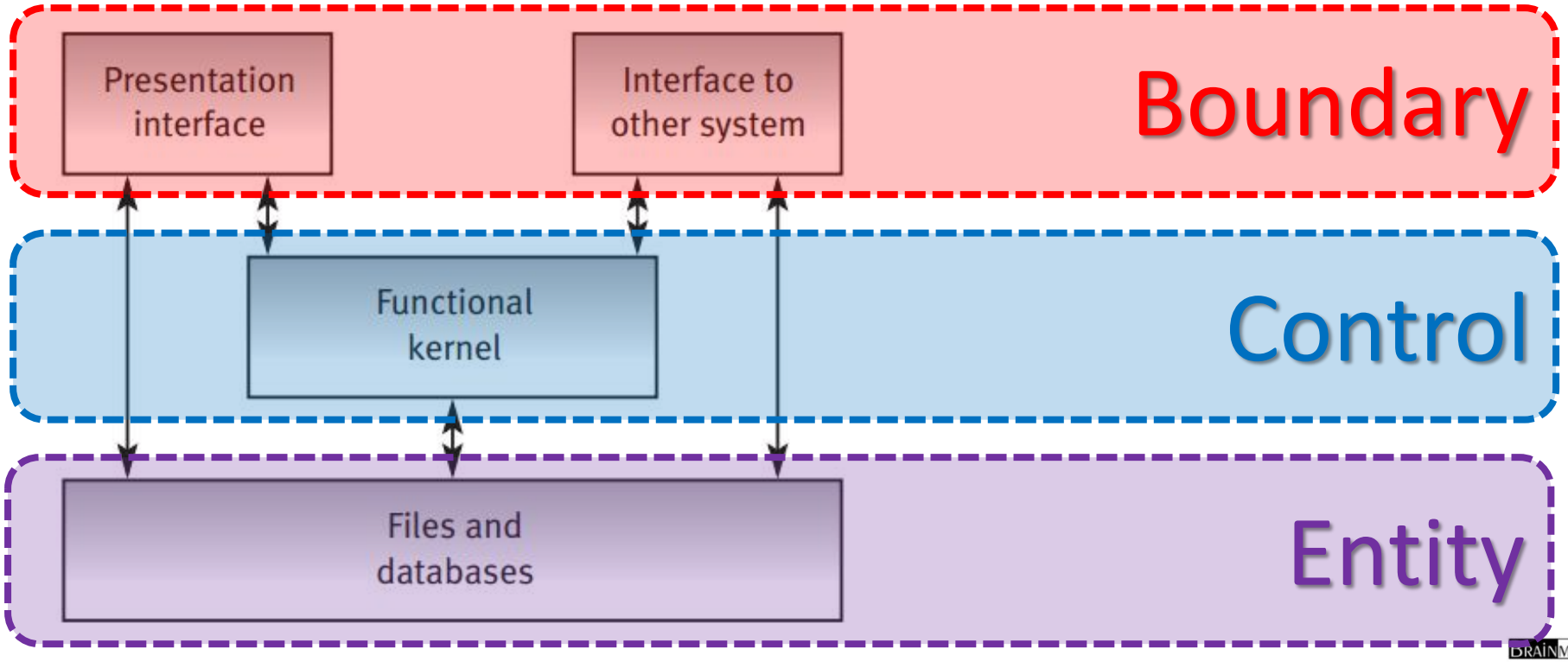




# Denert's Law (1991)

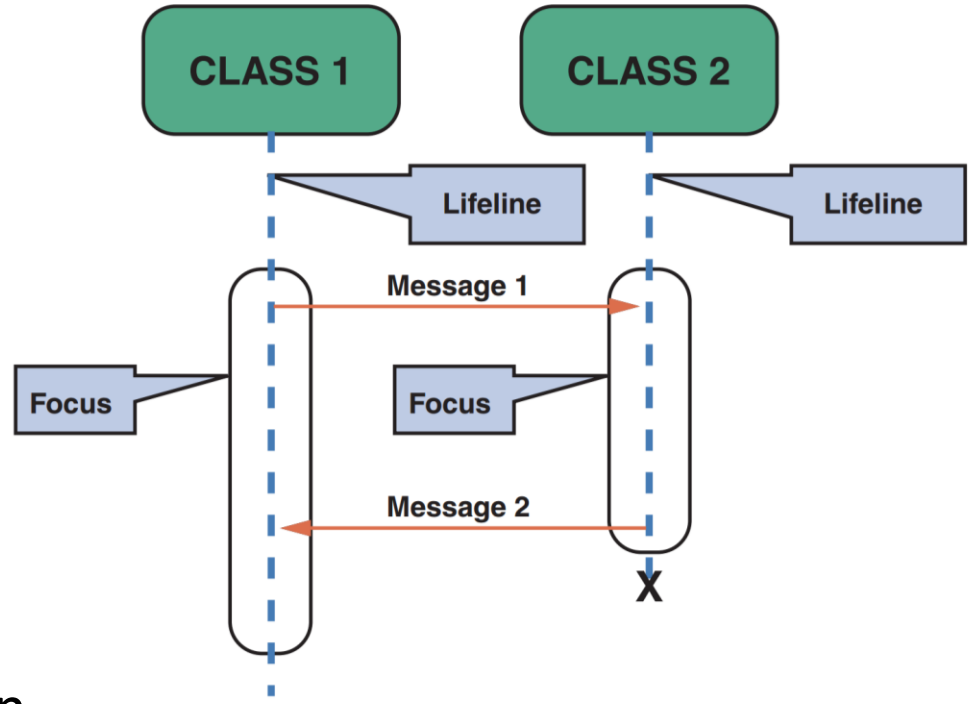
**Separation of concerns** leads to standard architectures

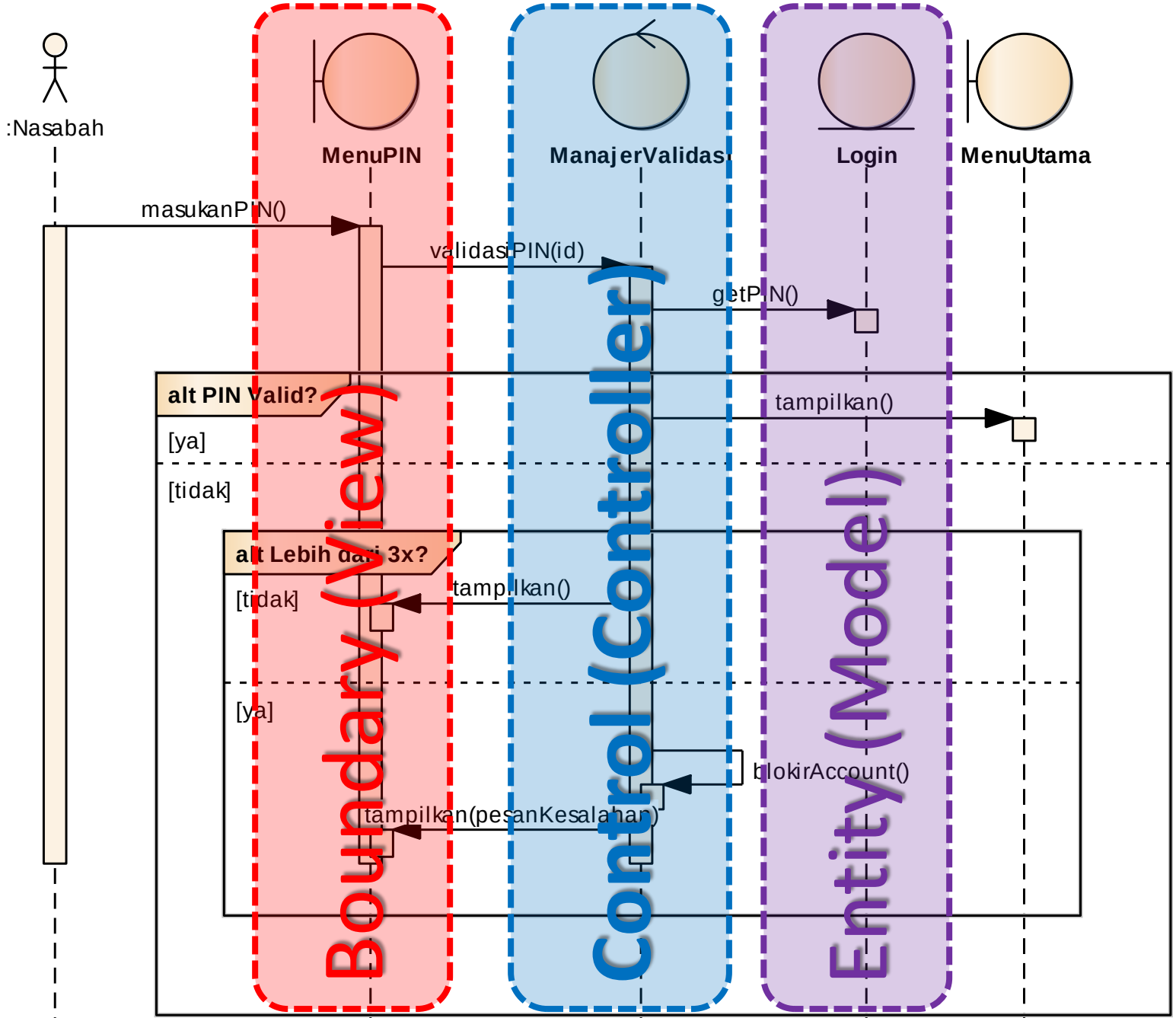
*(Endres, 2003)* *[L9]*

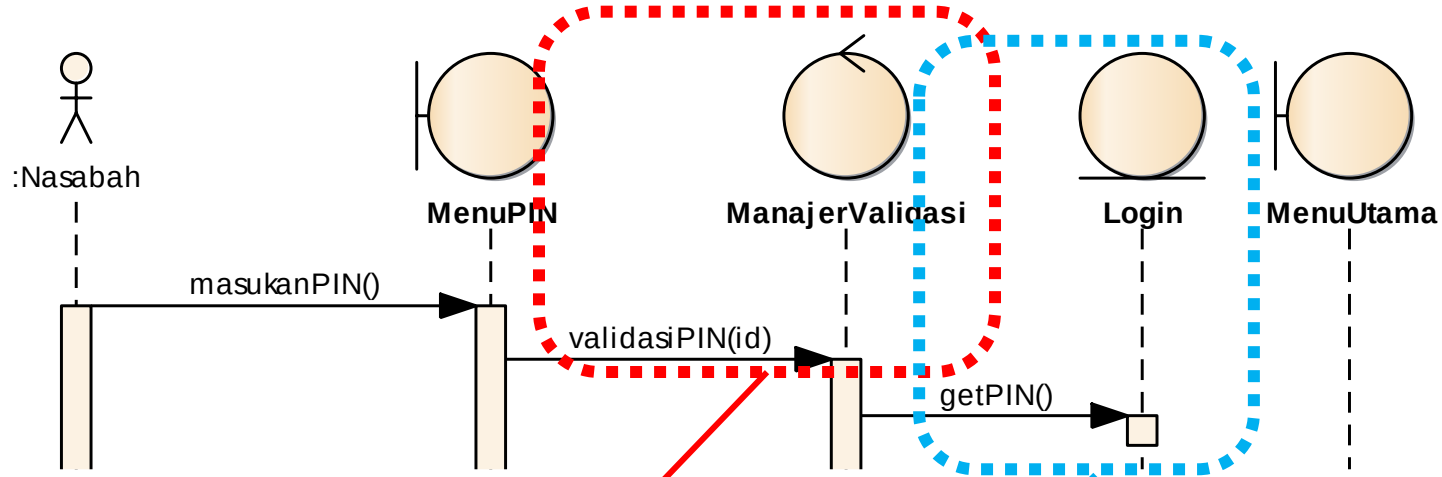


# Sequence Diagram berbasis Arsitektur Boundary – Control – Entity

1. Boundary Class:
   - Class yang berinteraksi dengan aktor langsung (user interface)
   - Form, input, UI, dsb
2. Control Class:
   - Class yang berhubungan dengan pemrosesan, penghitungan, kalkulasi, komputasi, query, dst
3. Entity Class:
   - Class yang berhubungan dengan data, penyimpanan data/file








```
public class ManajerValidasi {

        private Login m_Login;

        public int validasiKartu(){

        }

        public int validasiPIN(){
            int pin = m_Login.getPIN();
            if(pin.equals(pinuser)){
                        MenuUtama.show()};
        }

        public void blokirKartu(){

        }
}
```

```
public class Login {

        public int getPIN(){

        }

}
```

# Sequence Diagram

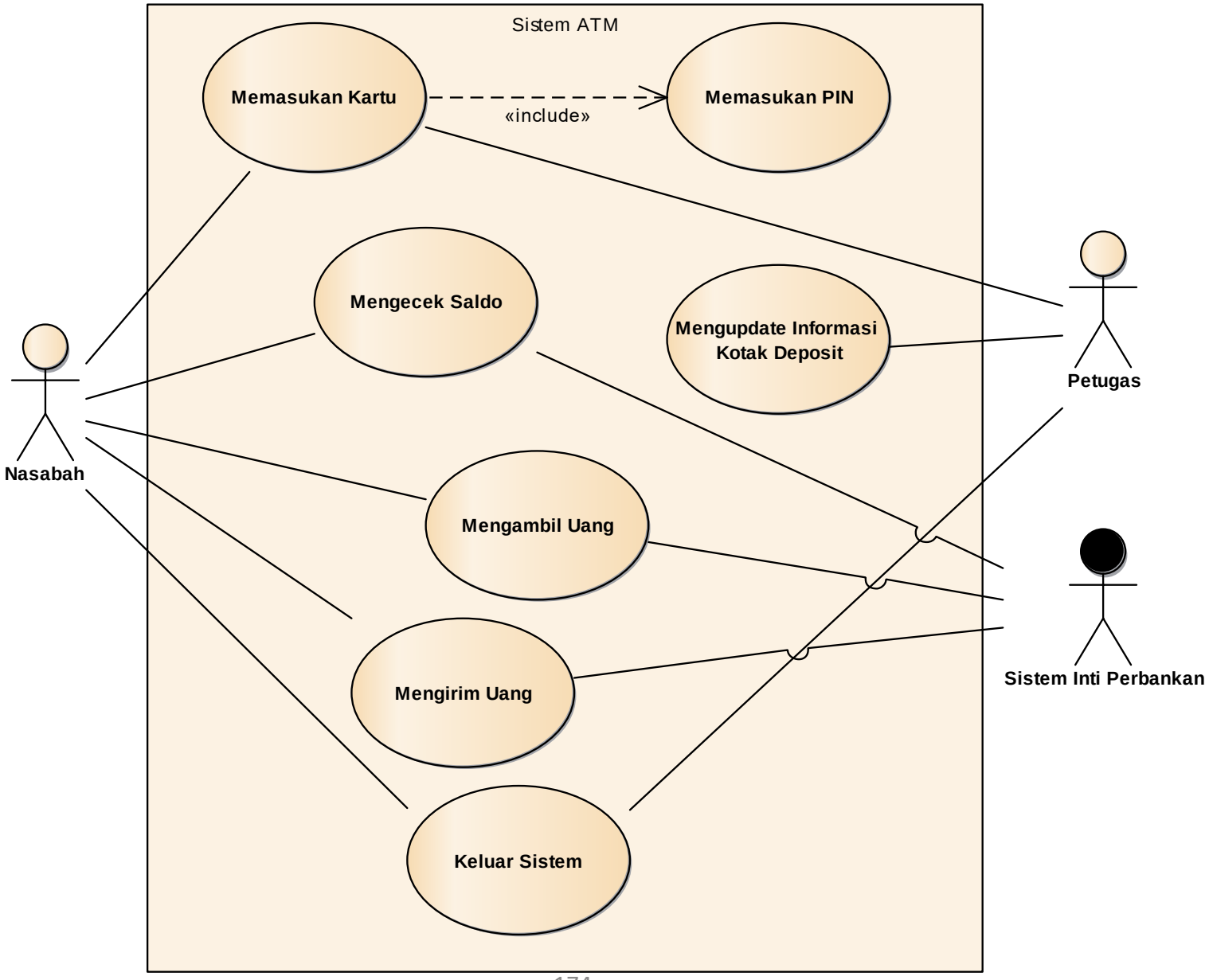
- Sequence Diagram dibuat untuk setiap Use Case yang dibuat

- Dimulai dari menarik Actor yang ada di Use Case Diagram, dilanjutkan dengan membuat sequence detail dari alur proses berjalannya Use Case dengan message yang mengalir didalamnya

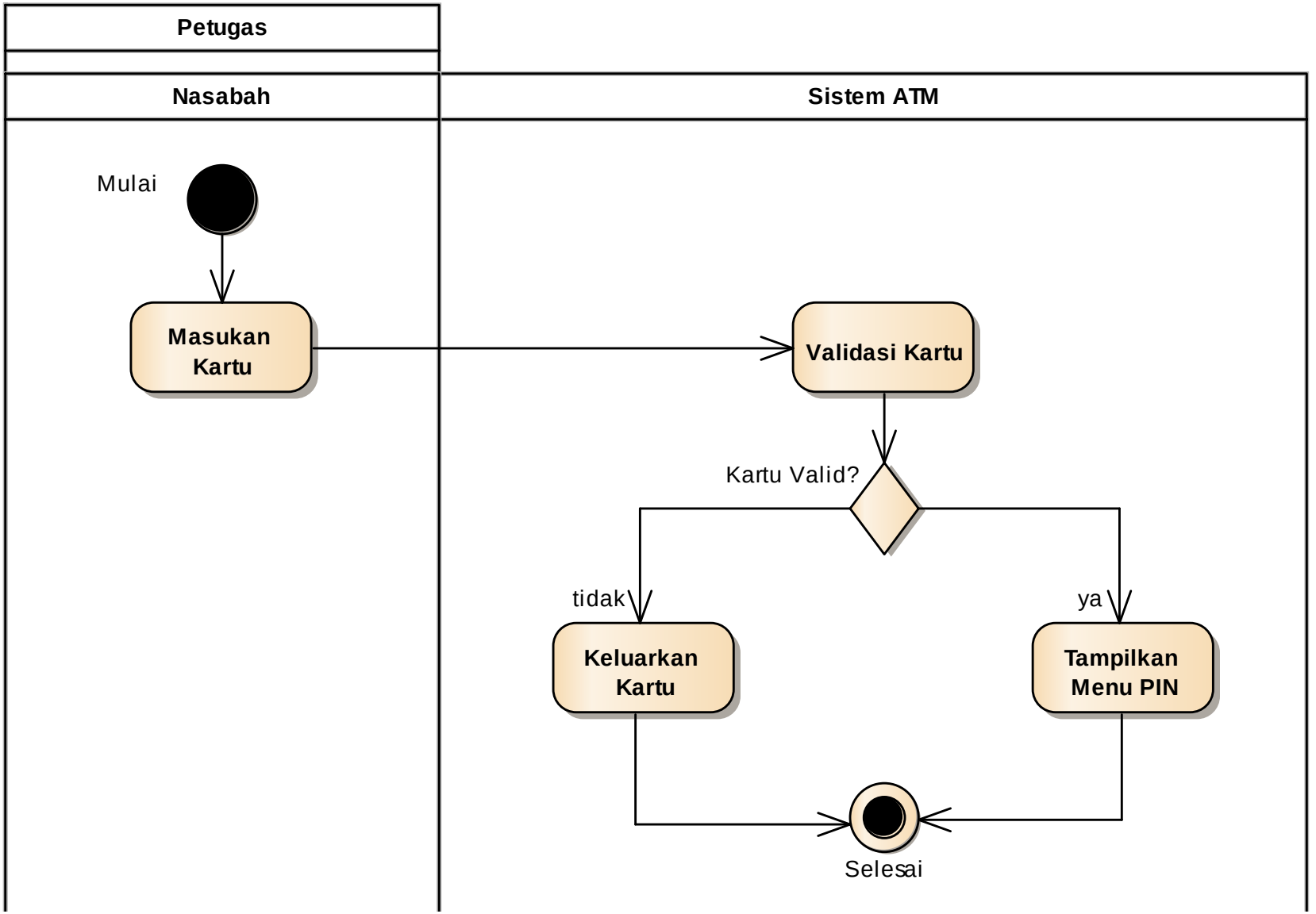
- Catatan: Objek dari Lifeline di Sequence Diagram akan menjadi kandidat Class, karena itulah harus mengikuti arsitektur Boundary – Control – Class

# Studi Kasus: Sequence Diagram Sistem ATM

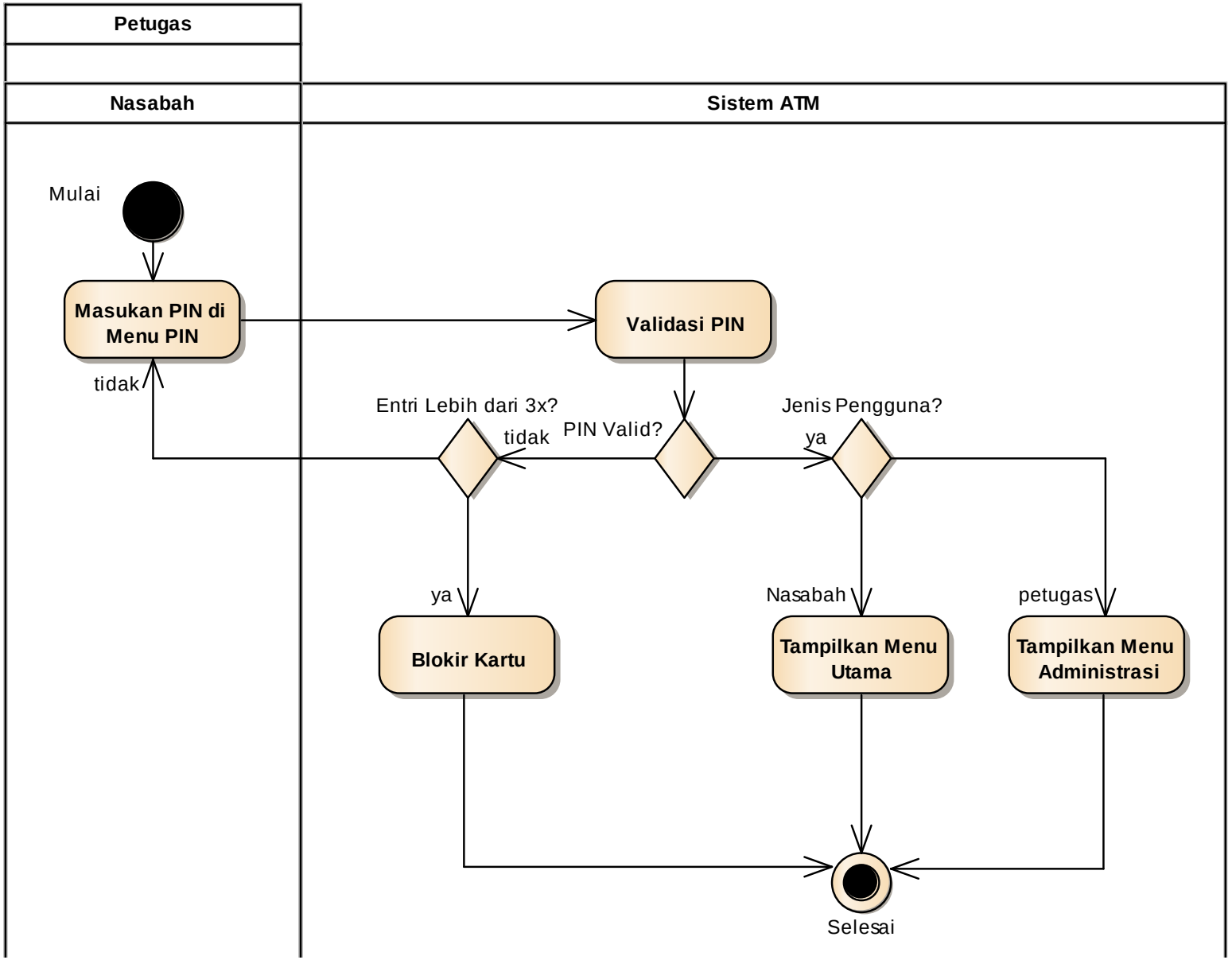
# Use Case Diagram Sistem ATM



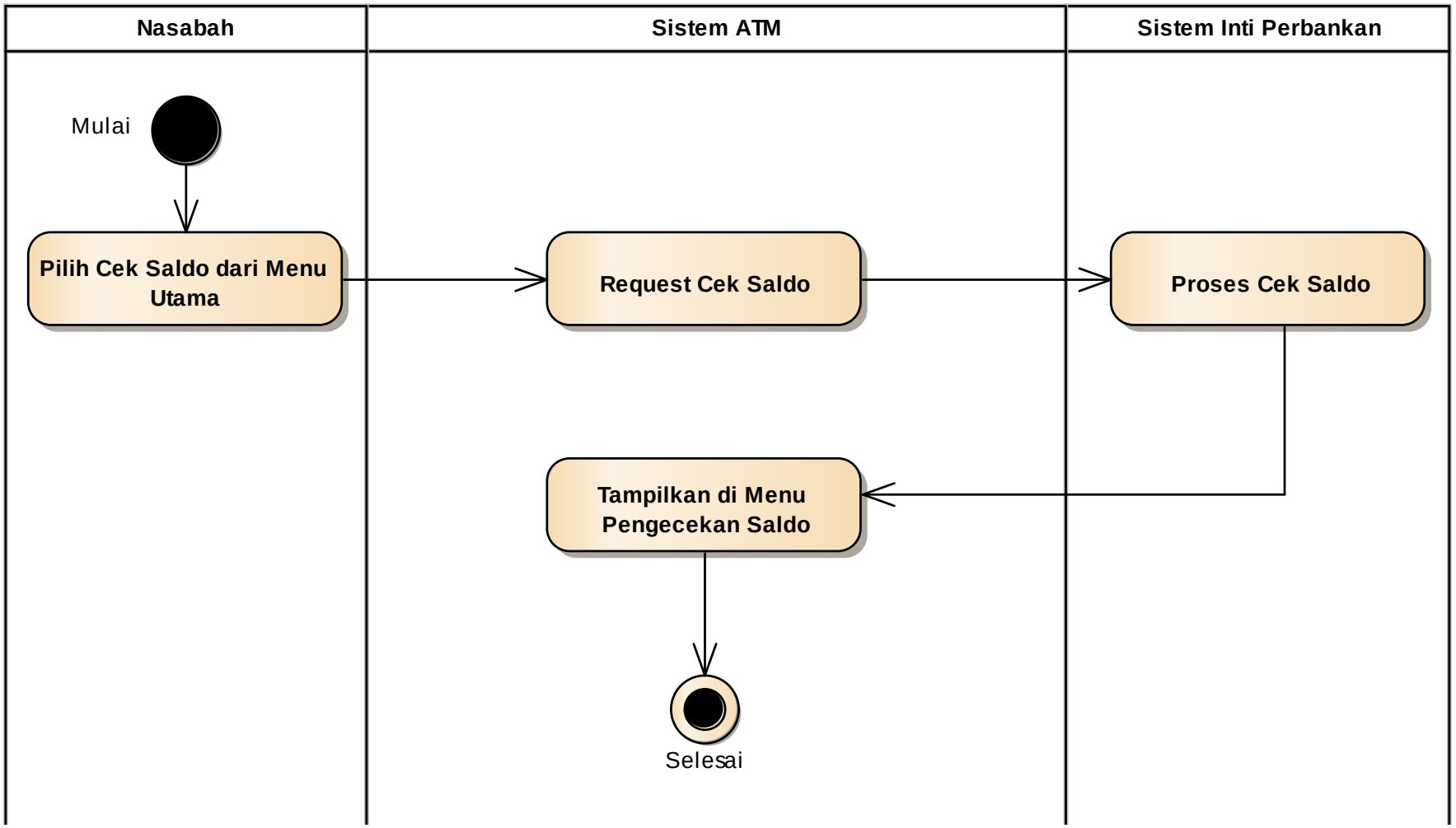
# Activity Diagram: Memasukkan Kartu



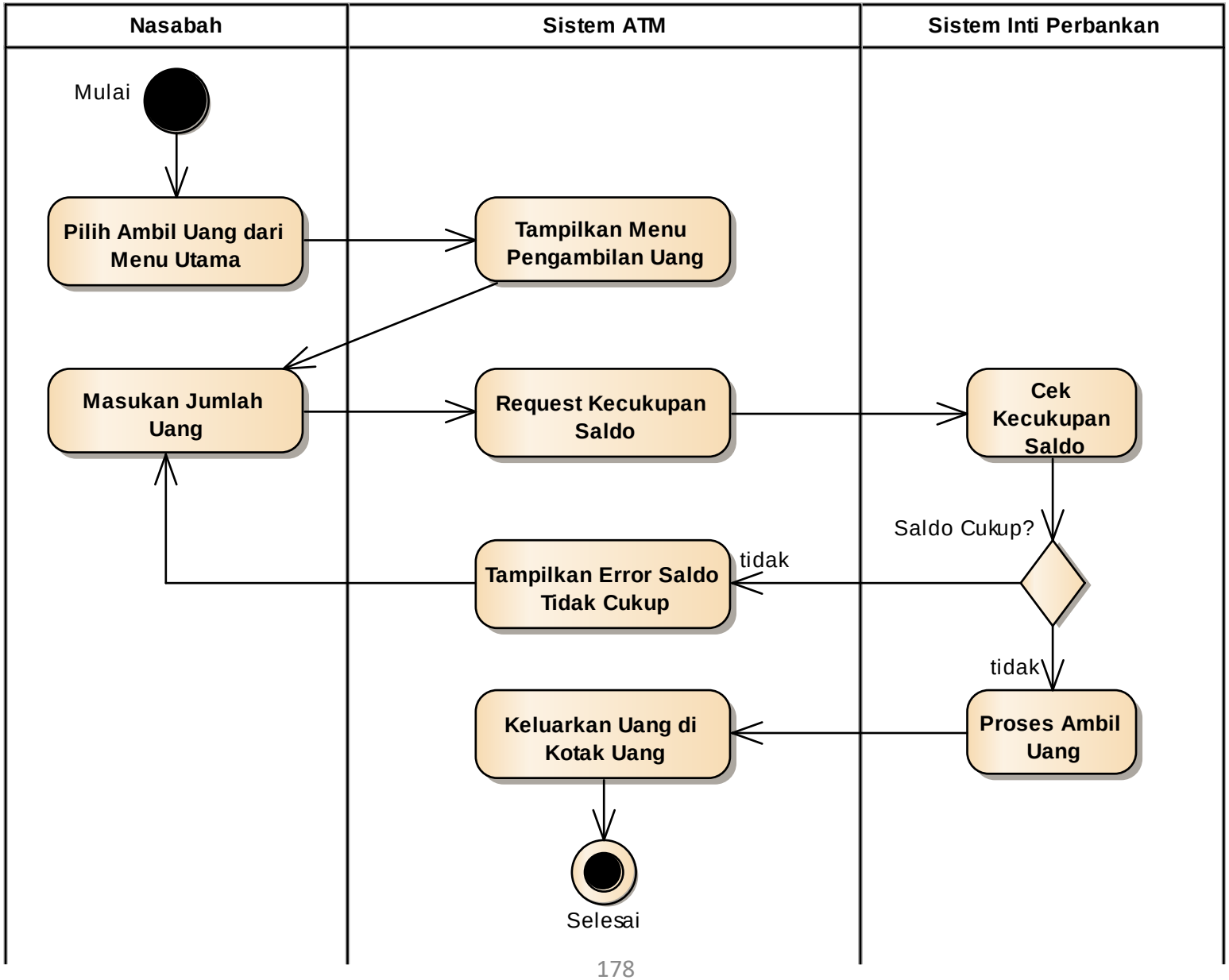
# Activity Diagram: Memasukkan PIN



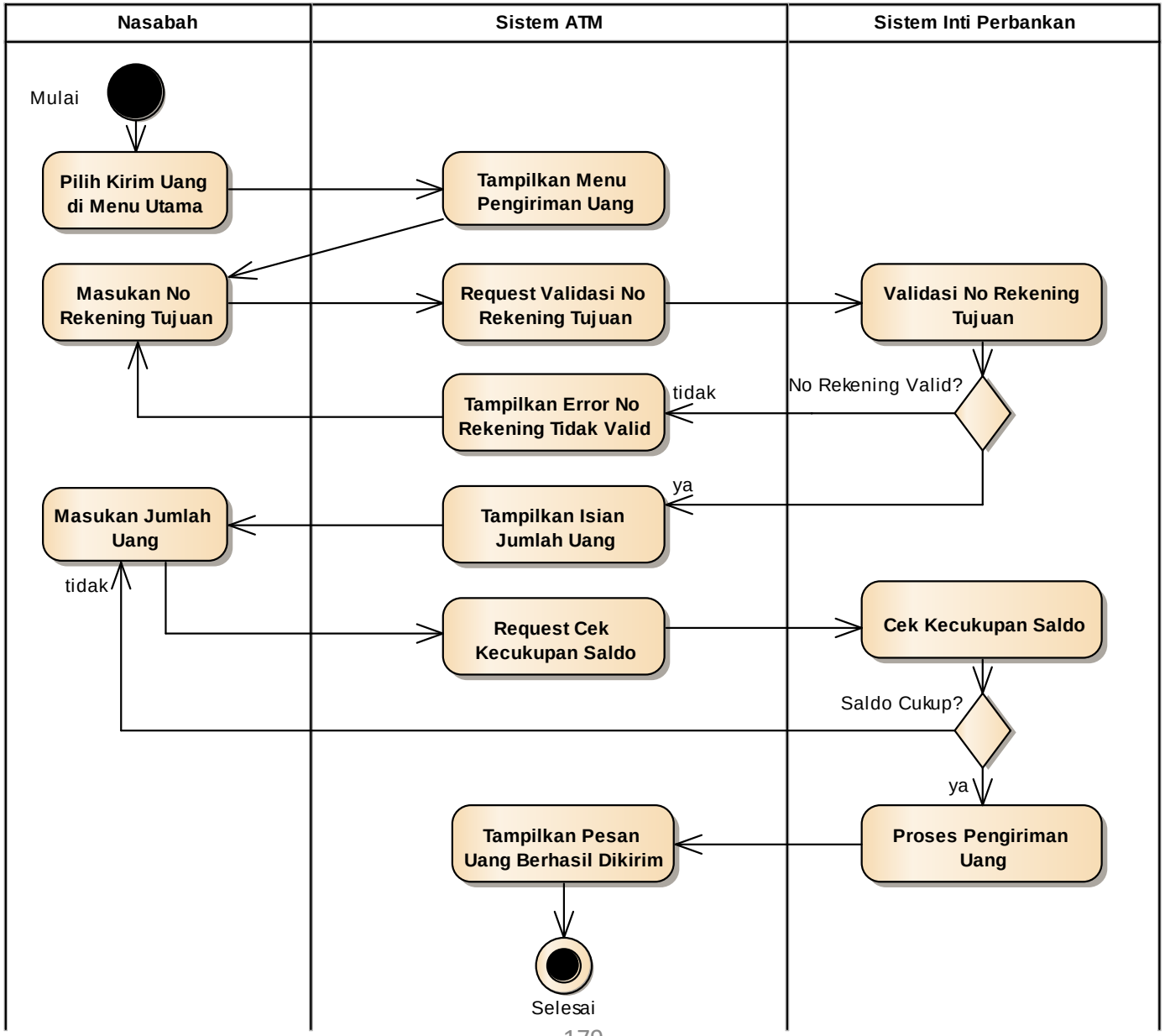
# Activity Diagram: Mengecek Saldo



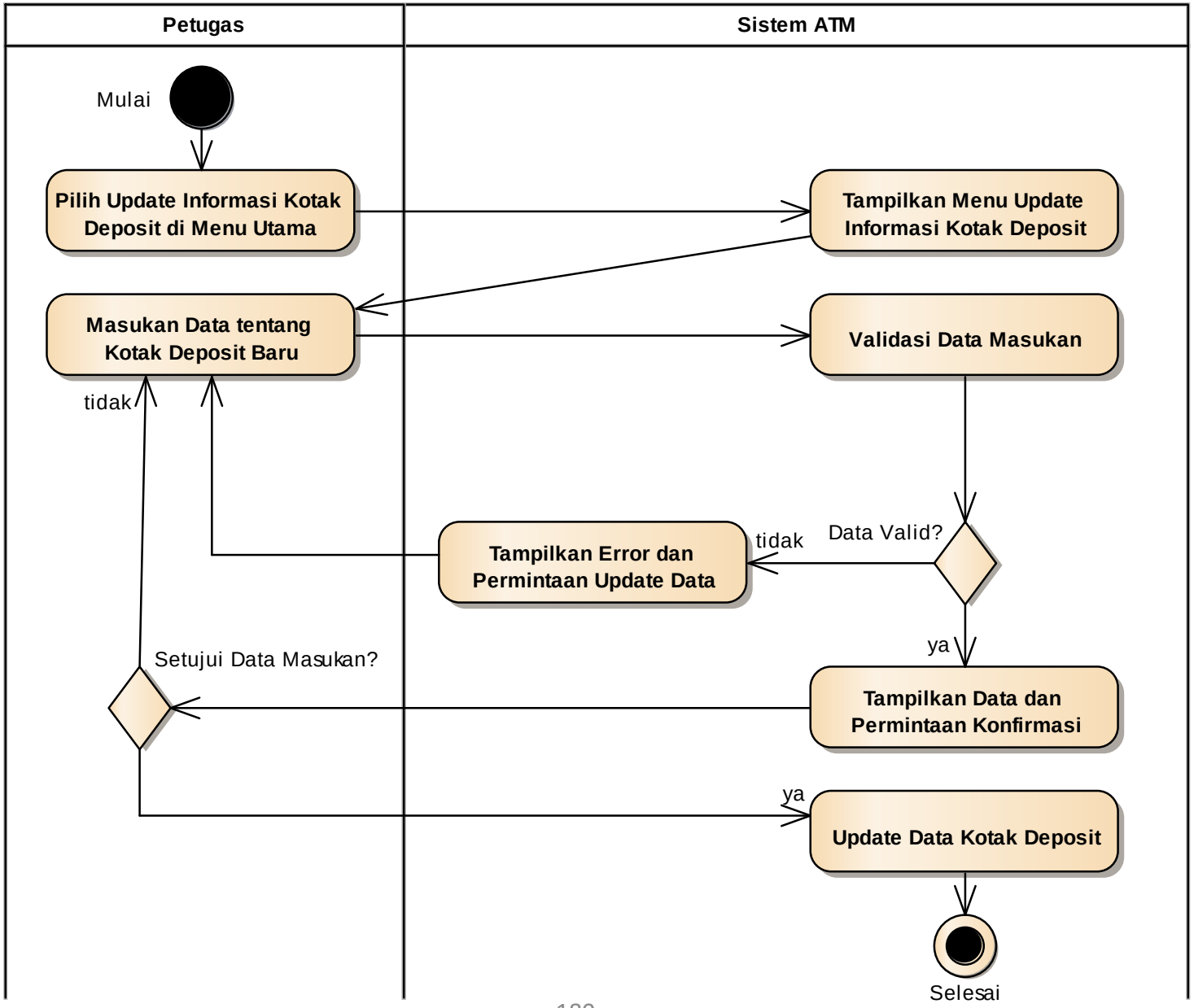
# Activity Diagram: Mengambil Uang



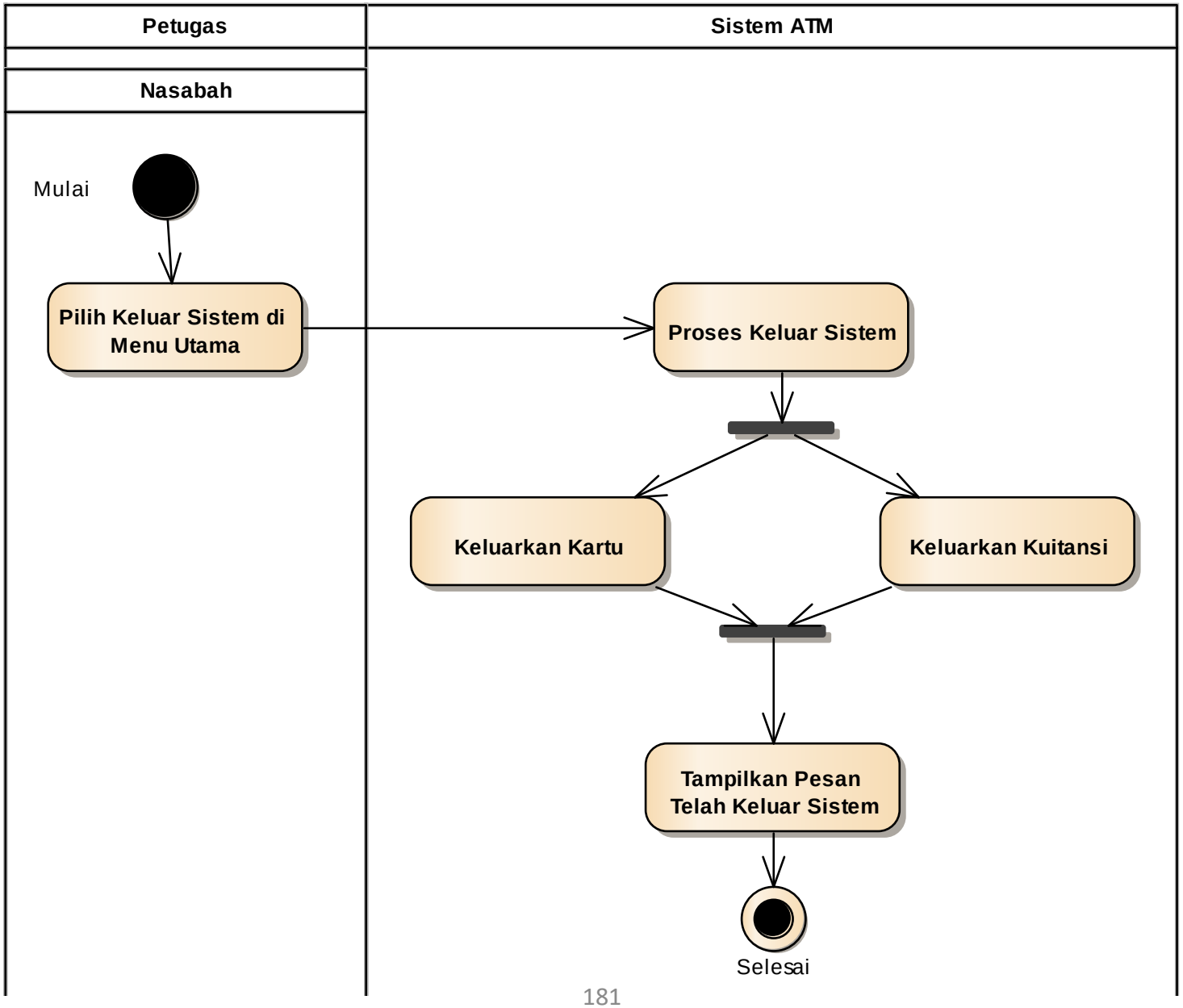
# Activity Diagram: Mengirim Uang



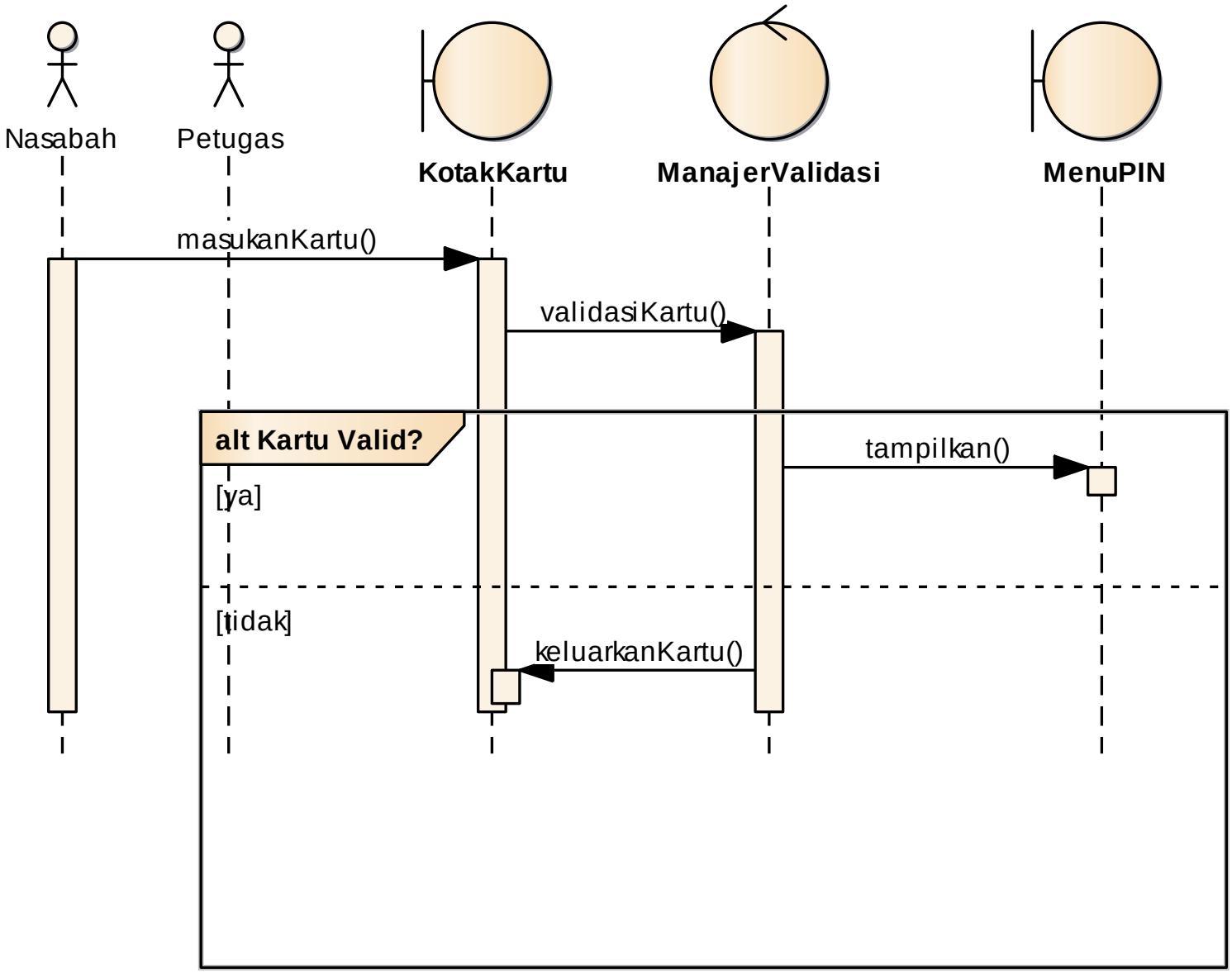
# Activity Diagram: Mengupdate Informasi Kotak Deposit



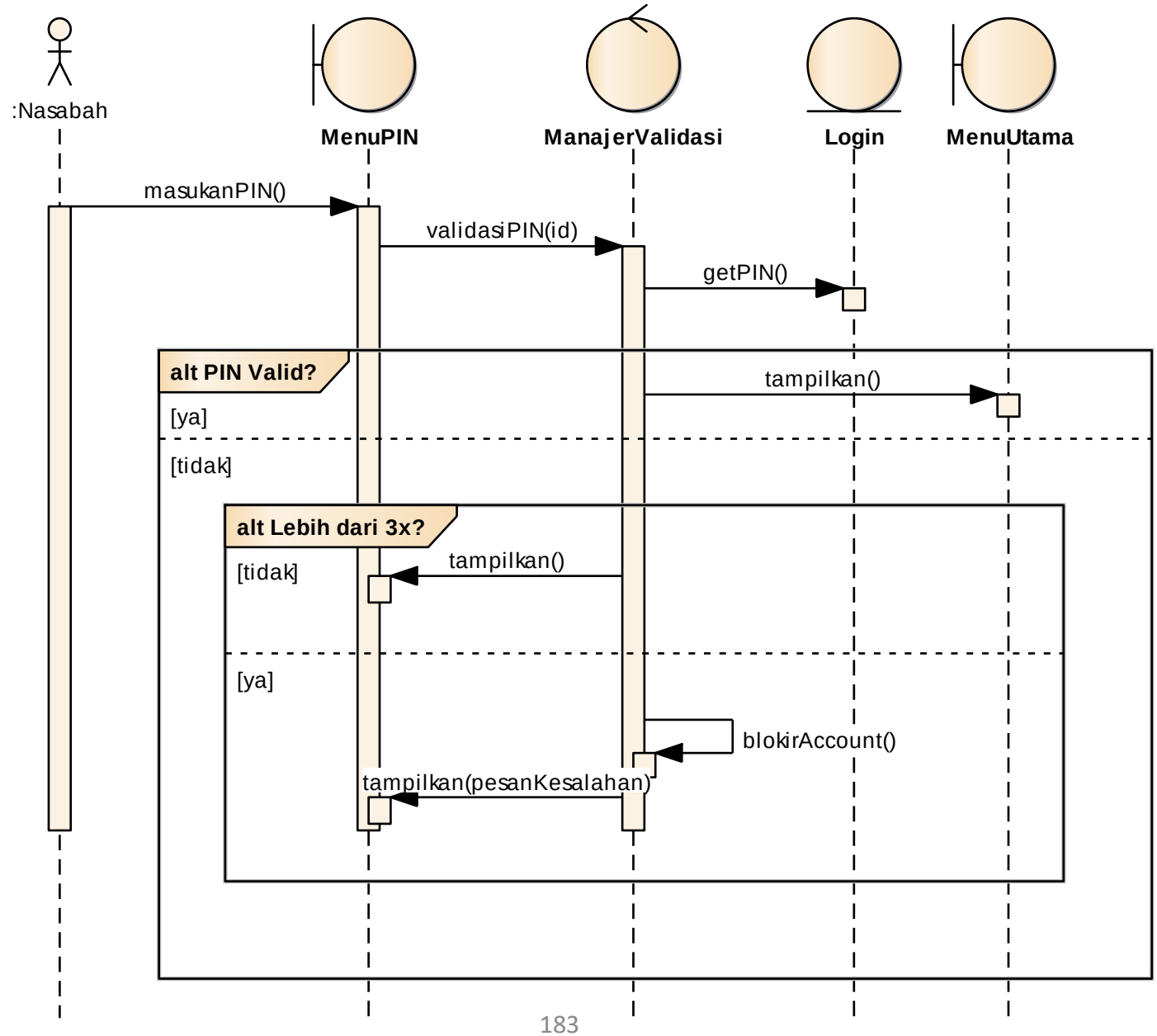
# Activity Diagram: Keluar Sistem



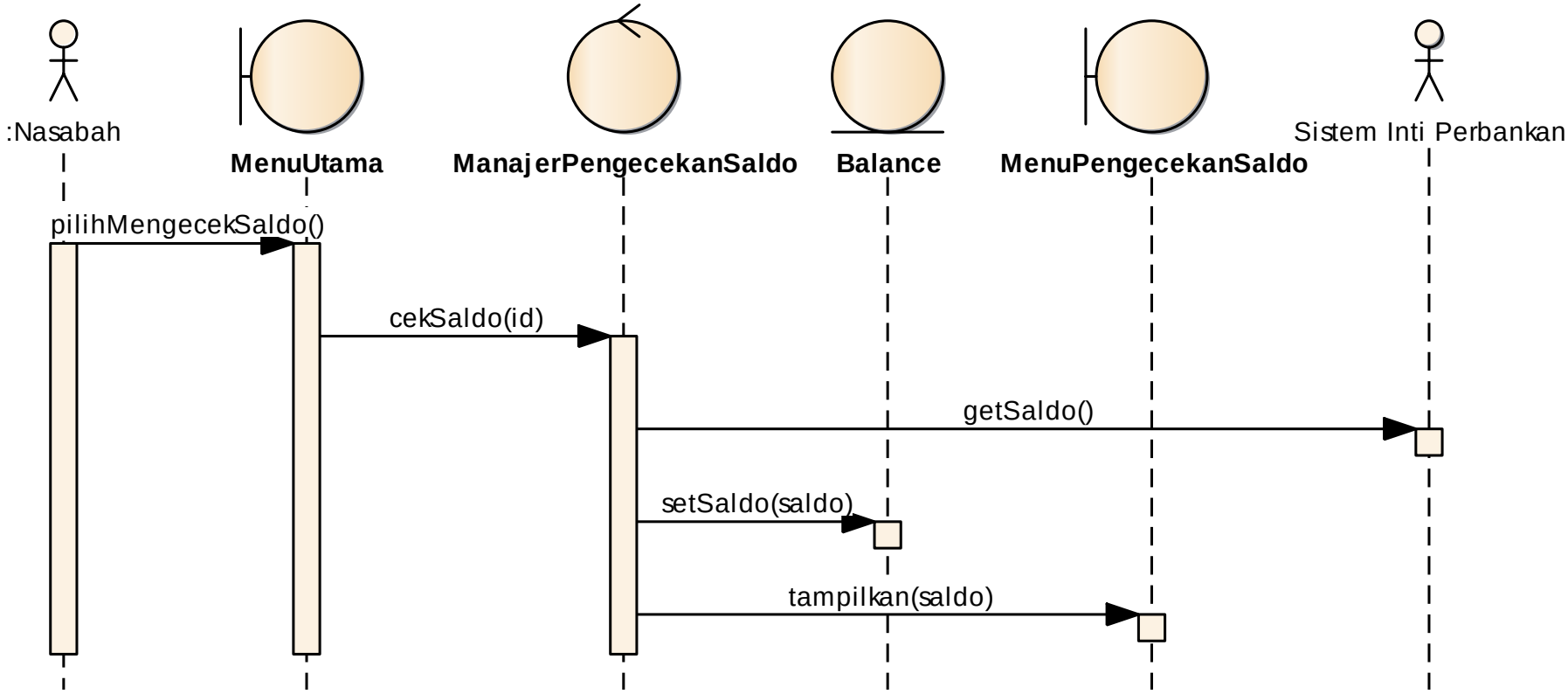
# Sequence Diagram: Memasukkan Kartu



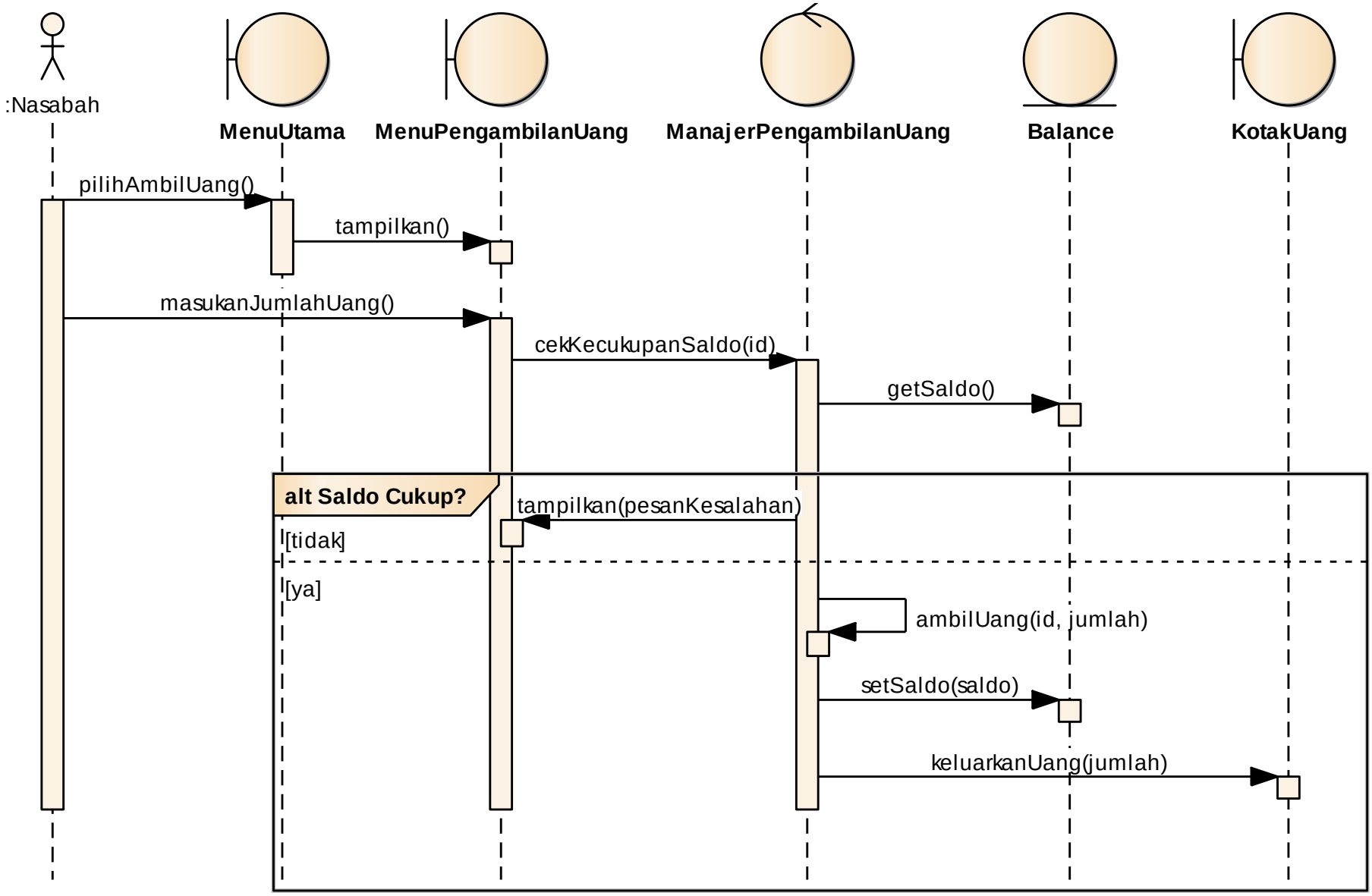
# Sequence Diagram: Memasukkan PIN



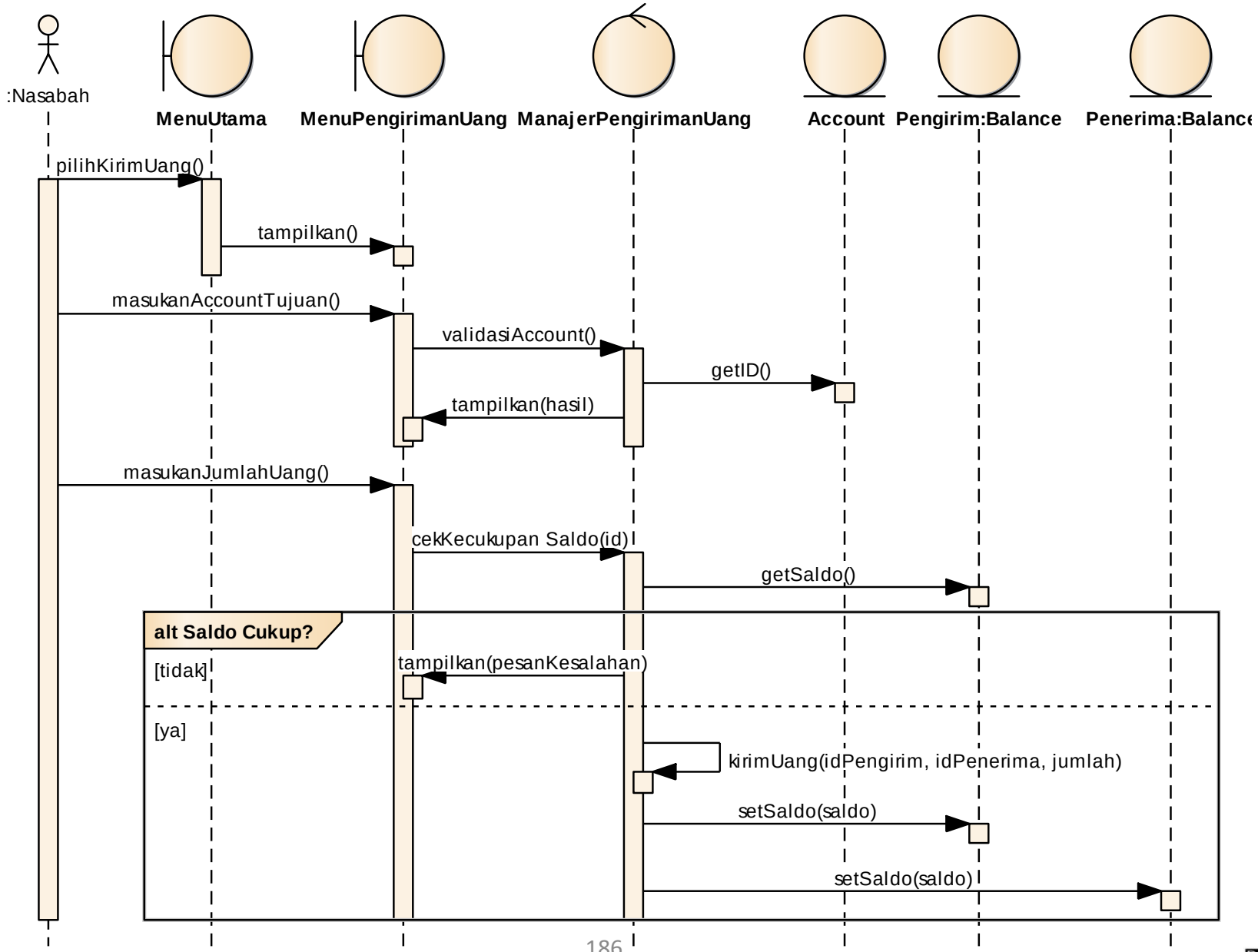
# Sequence Diagram: Mengecek Saldo



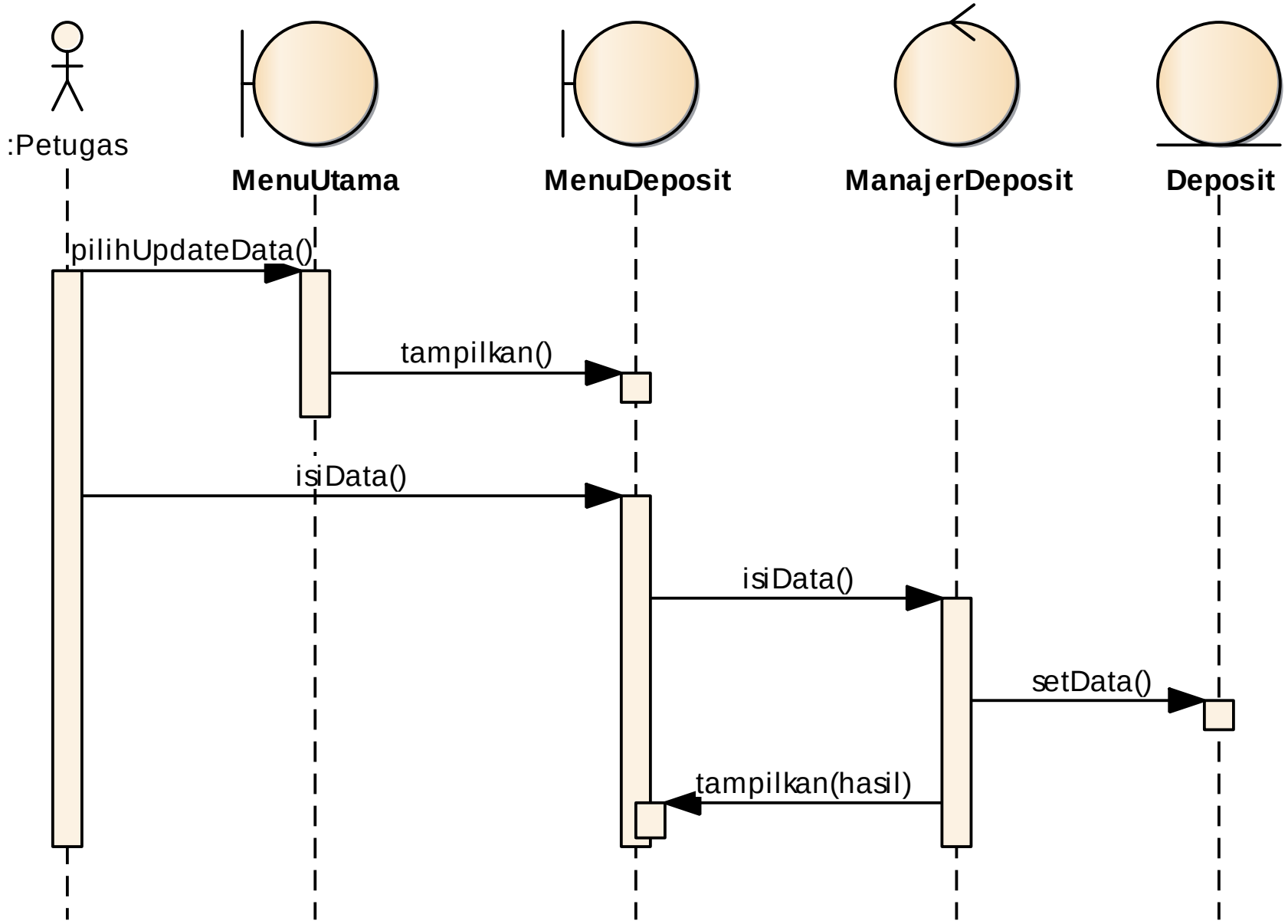
# Sequence Diagram: Mengambil Uang



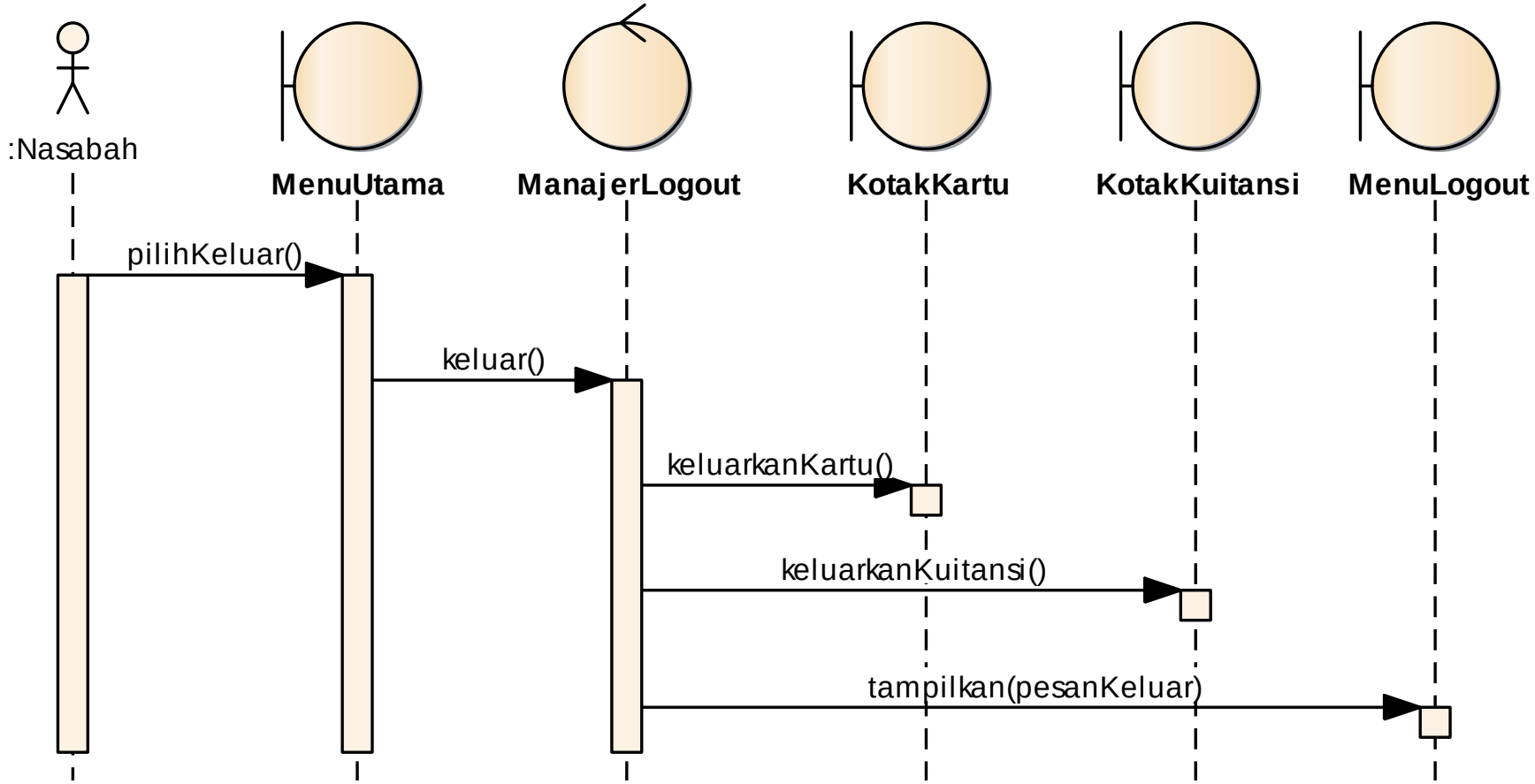
# Sequence Diagram: Mengirim Uang



# Sequence Diagram: Mengupdate Informasi Kotak Deposit

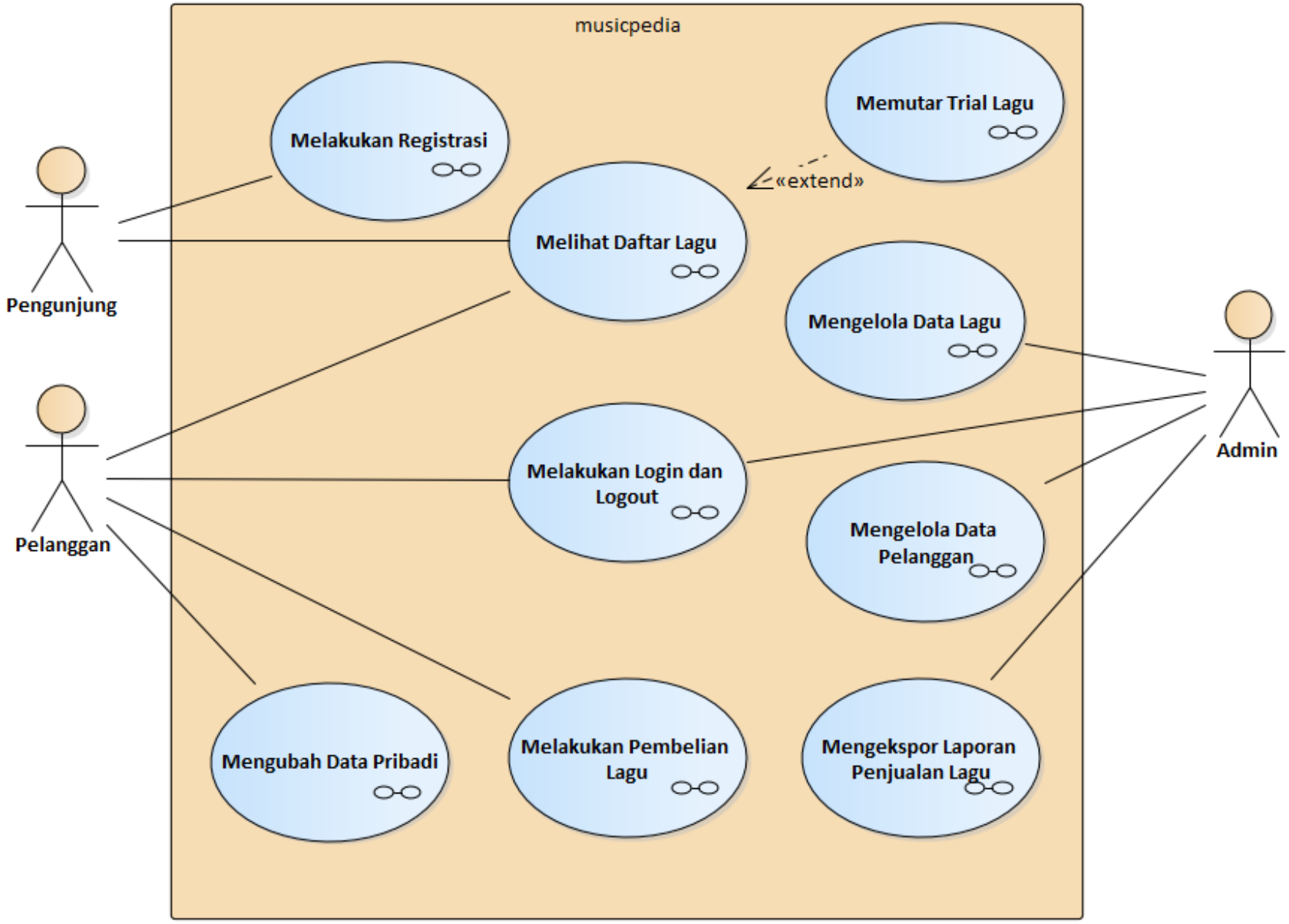


# Sequence Diagram: Keluar Sistem

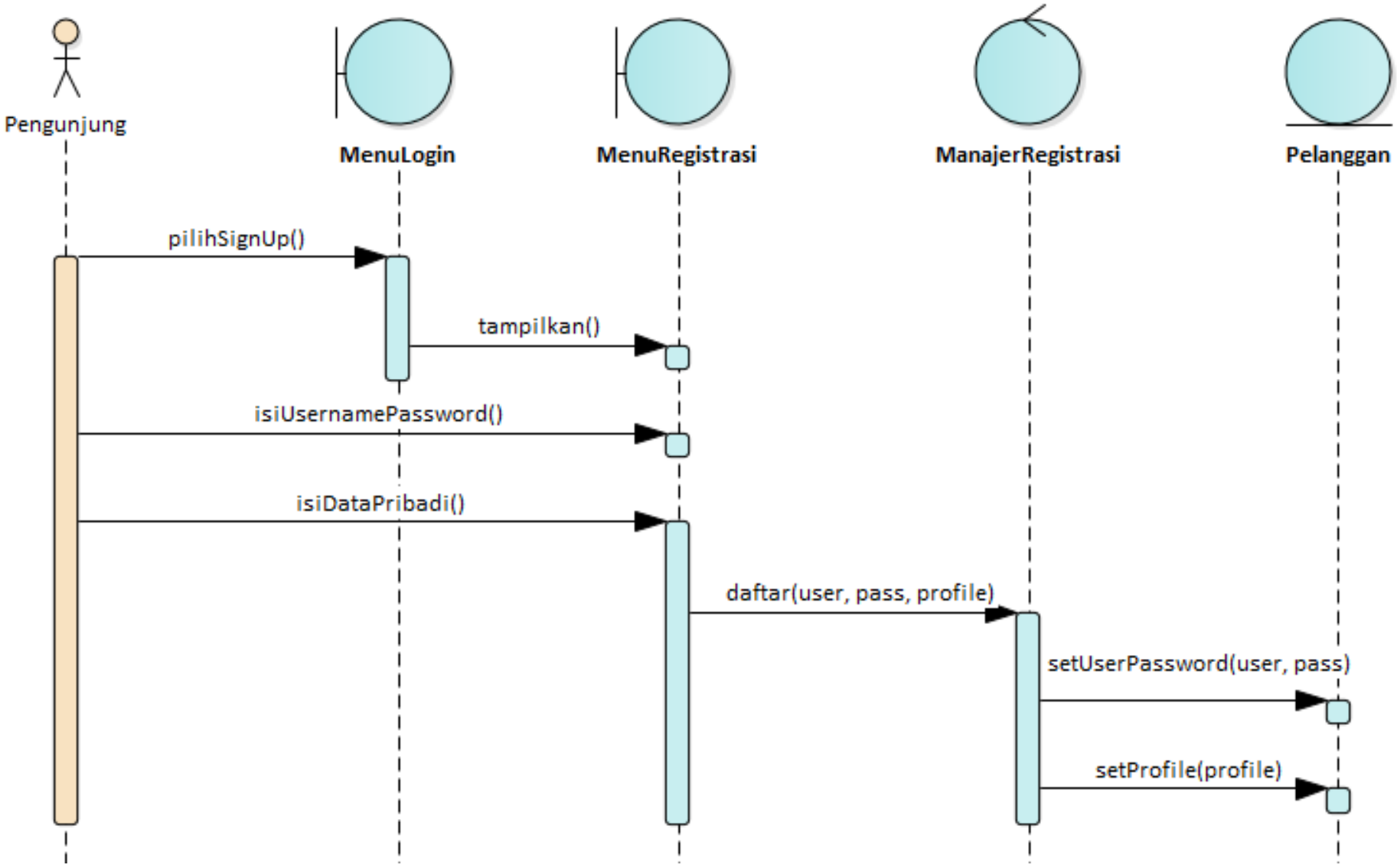


# Studi Kasus: Sequence Diagram MusicPedia

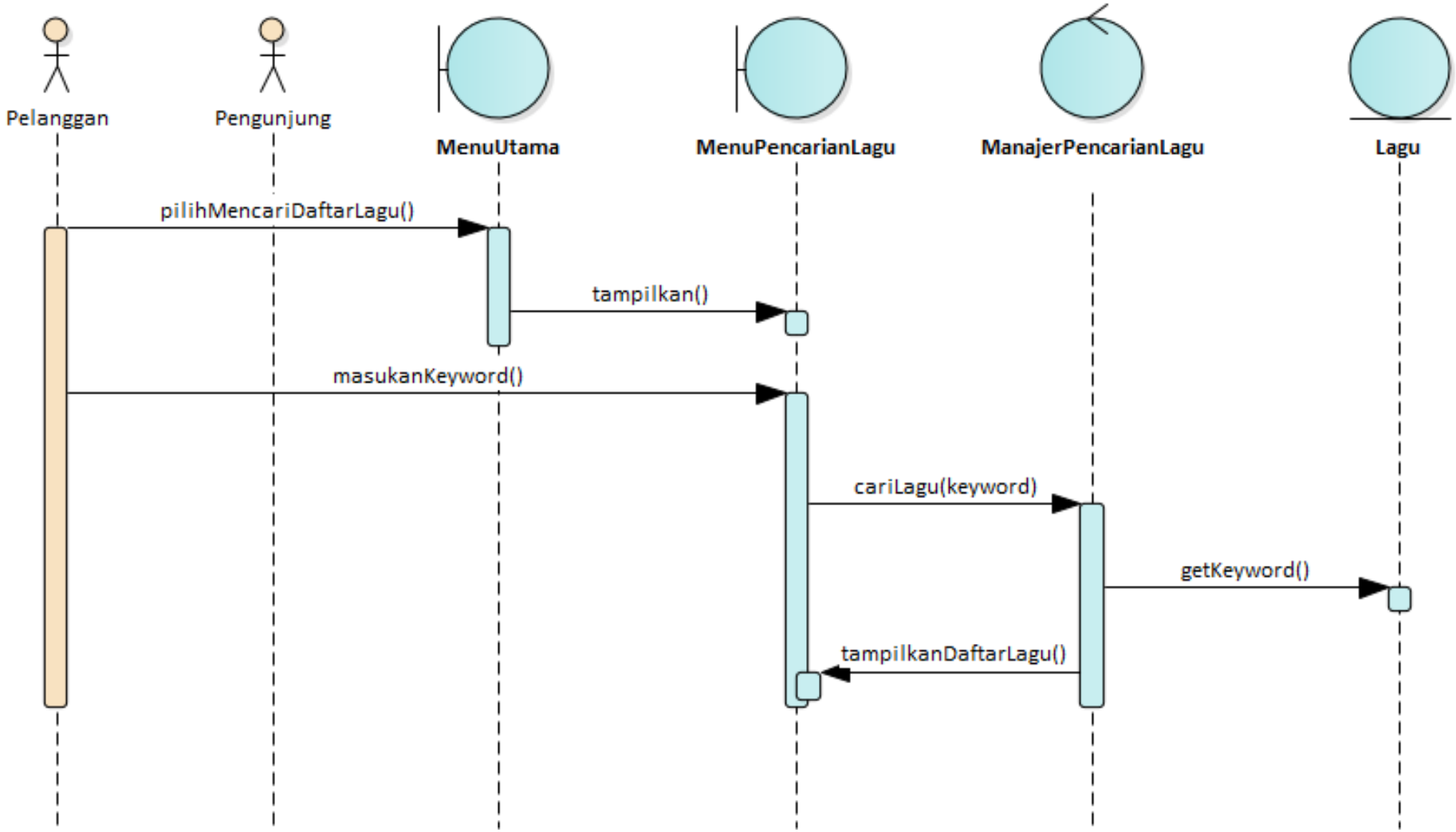
# Use Case Diagram MusicPedia



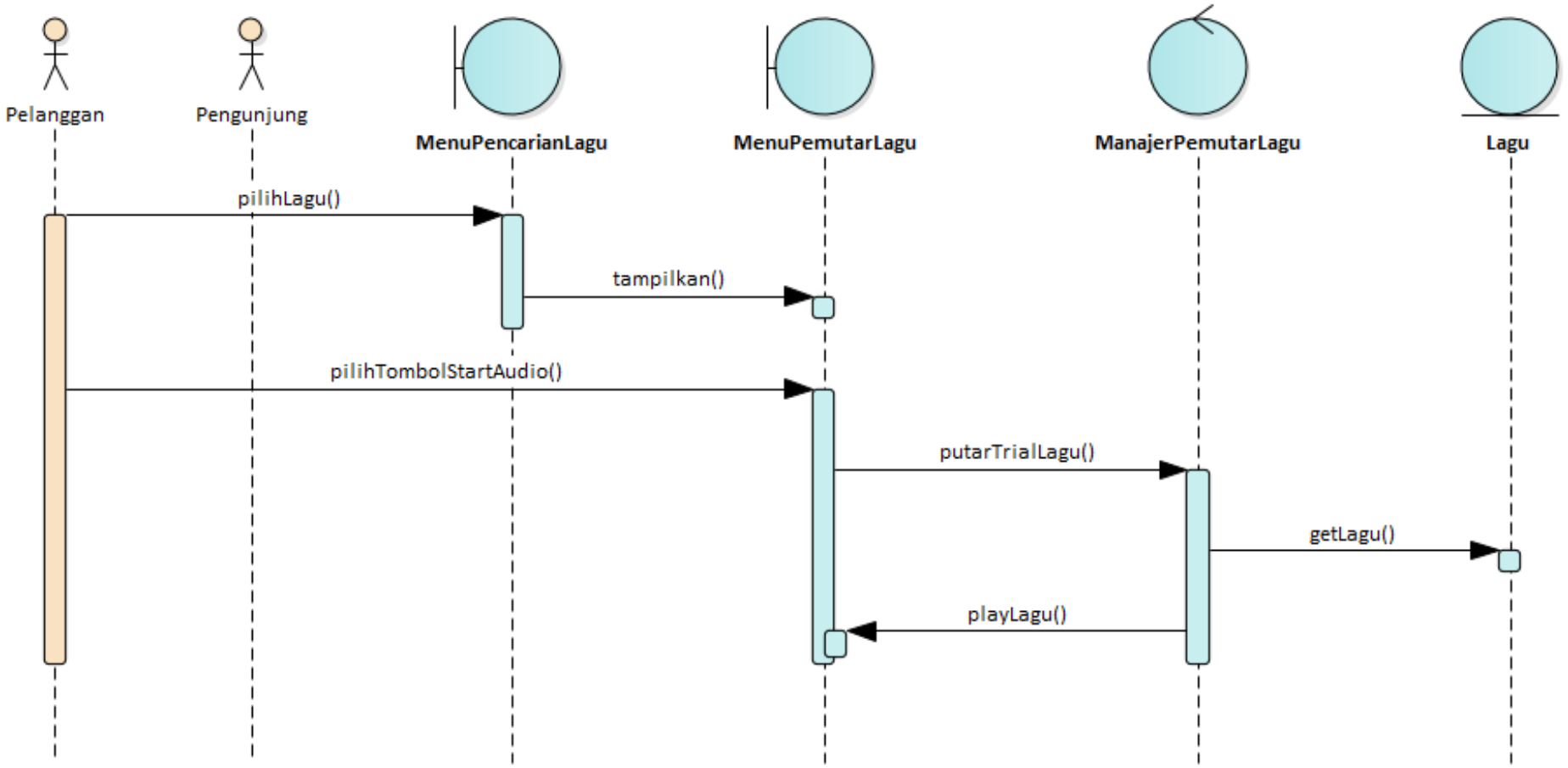
# Sequence Diagram: Melakukan Registrasi



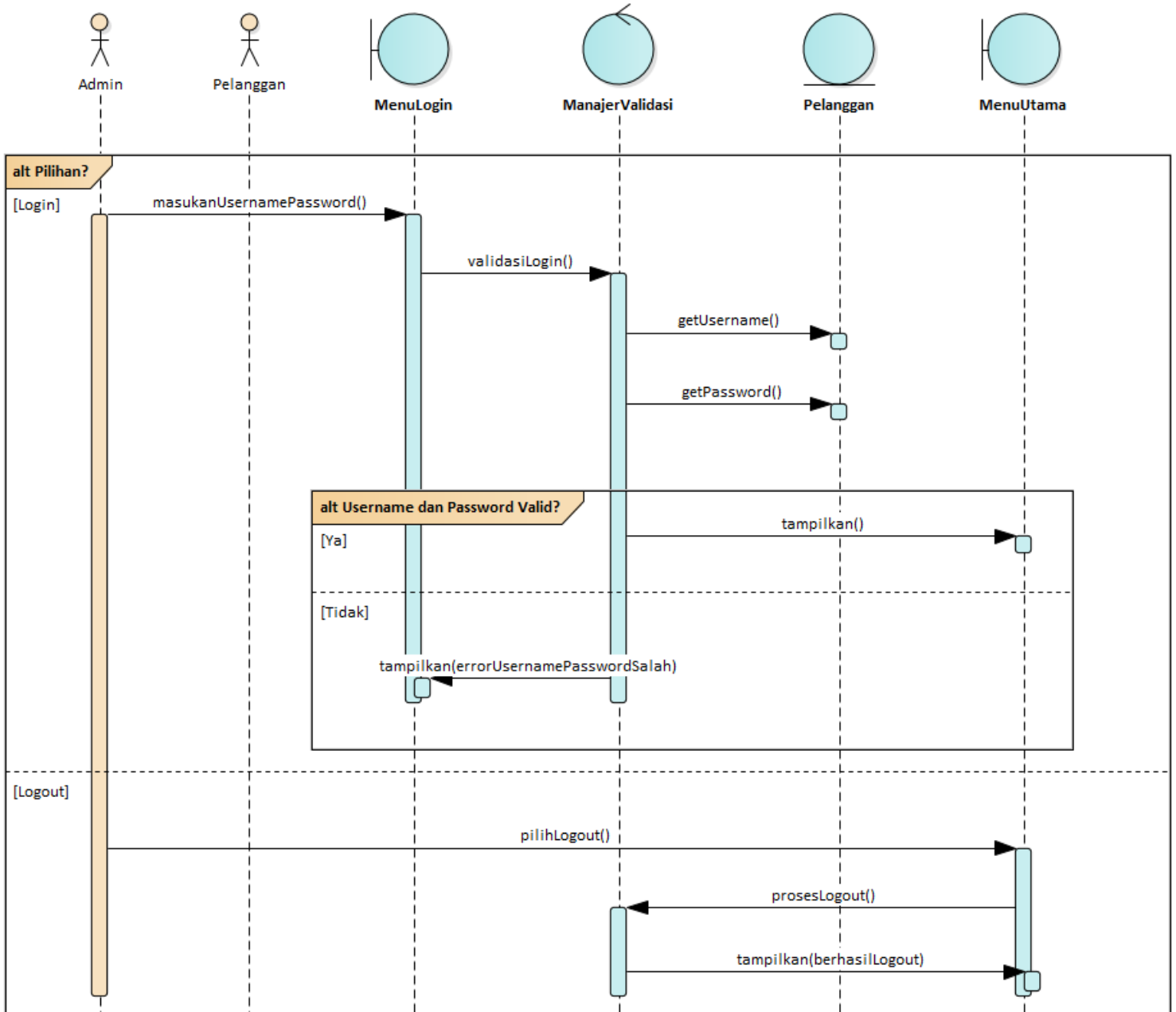
# Sequence Diagram: Melihat Daftar Lagu



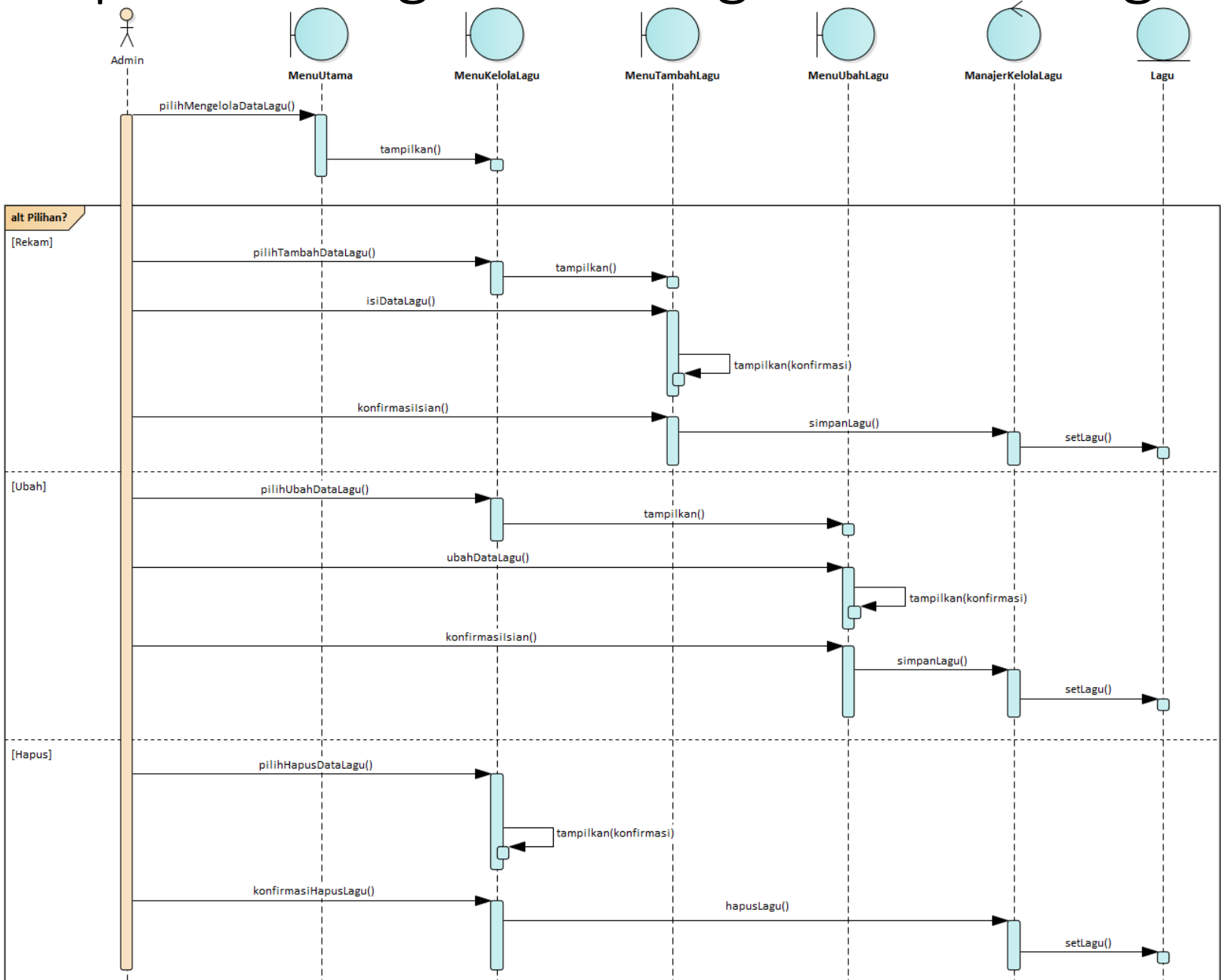
# Sequence Diagram: Memutar Trial Lagu



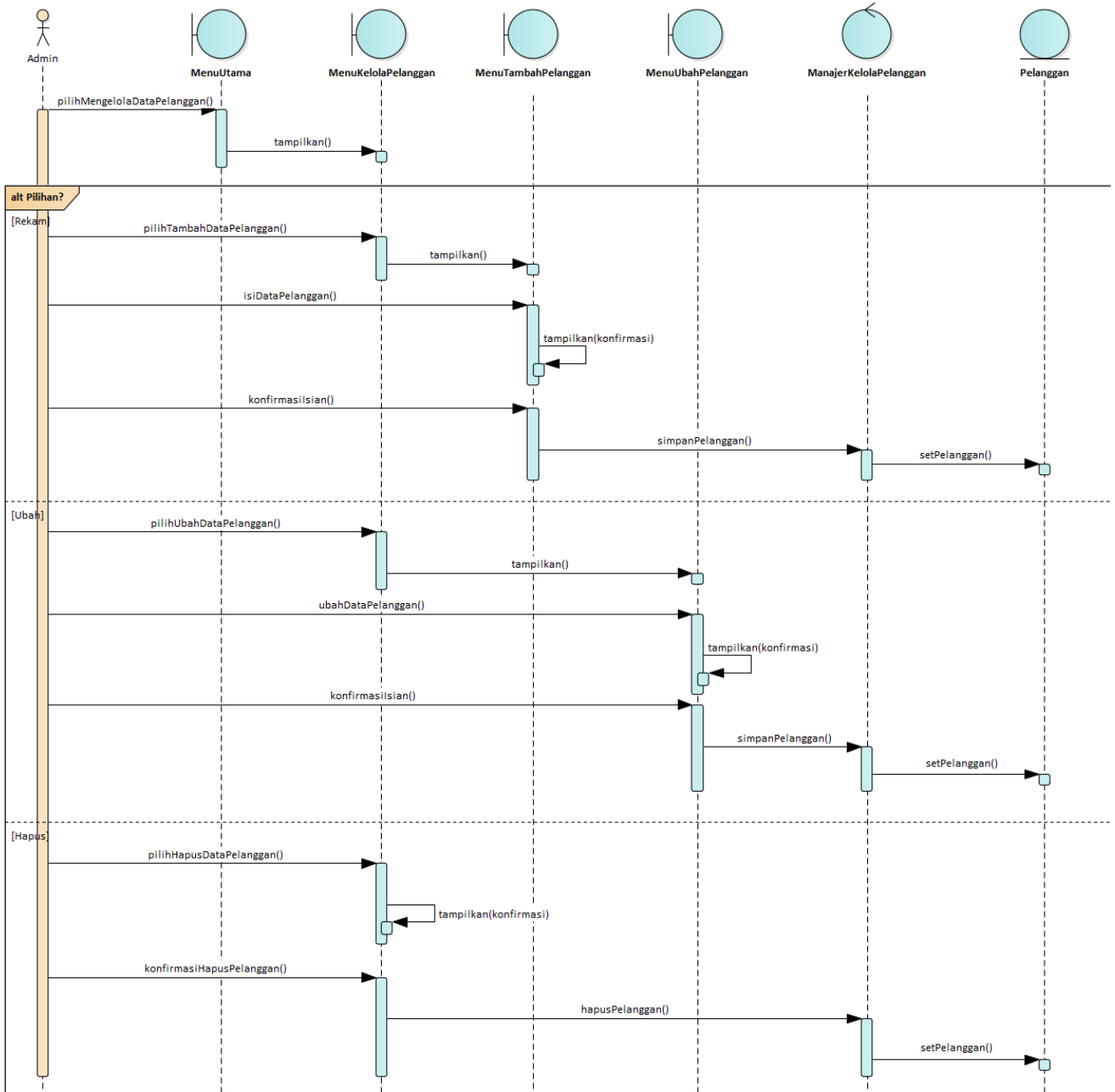
# Sequence Diagram: Melakukan Login dan Logout



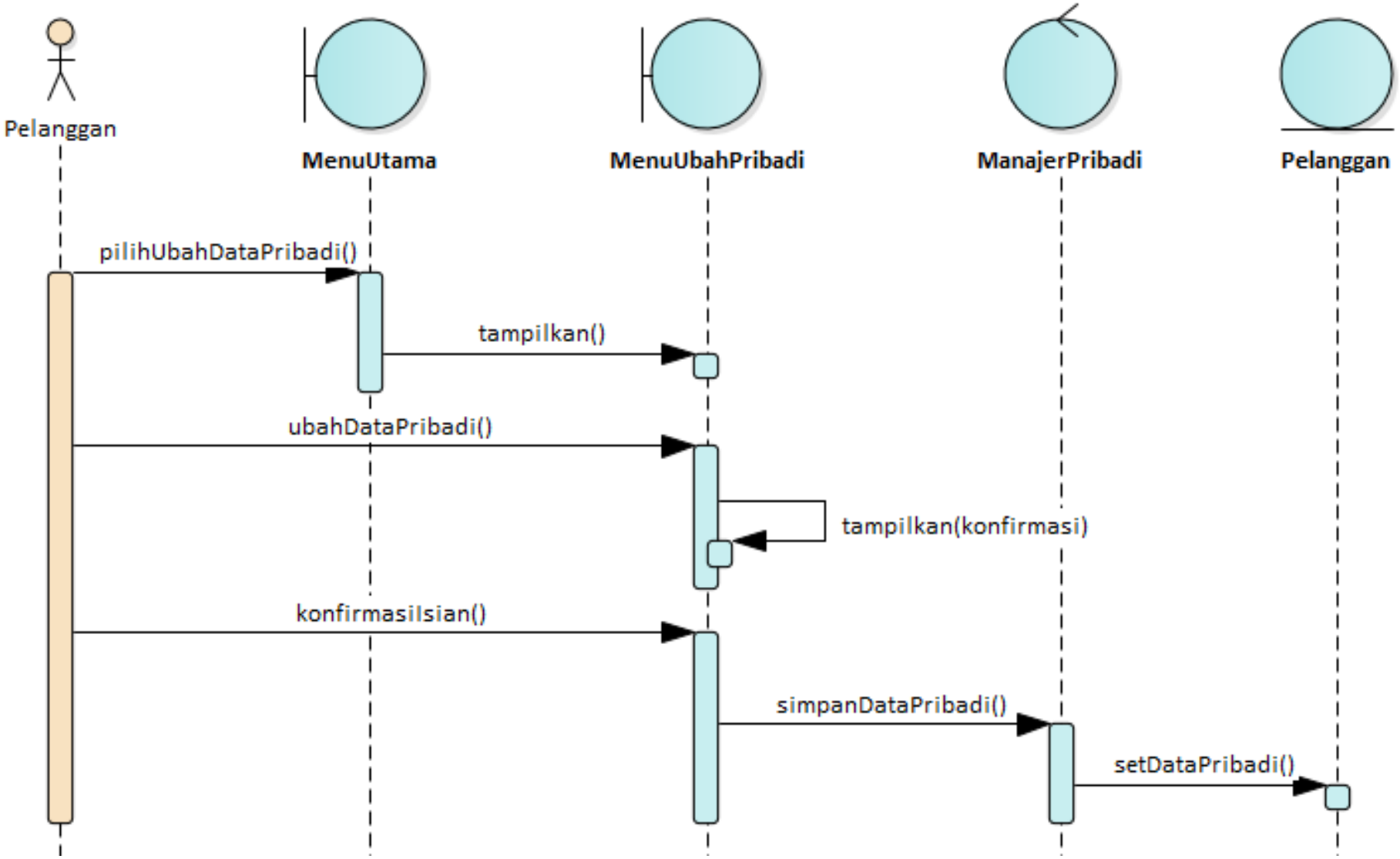
# Sequence Diagram: Mengelola Data Lagu



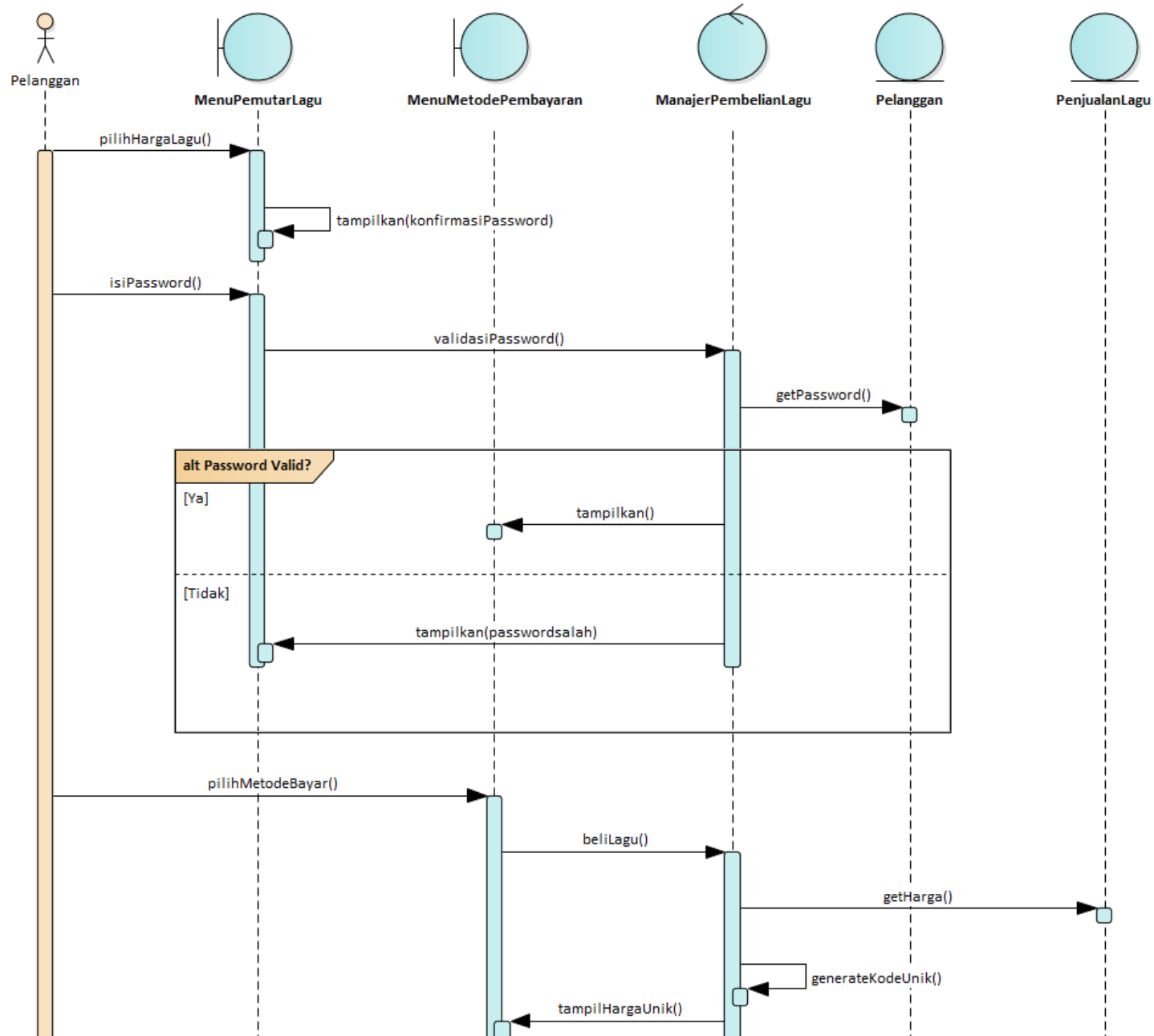
# Sequence Diagram: Mengelola Data Pelanggan



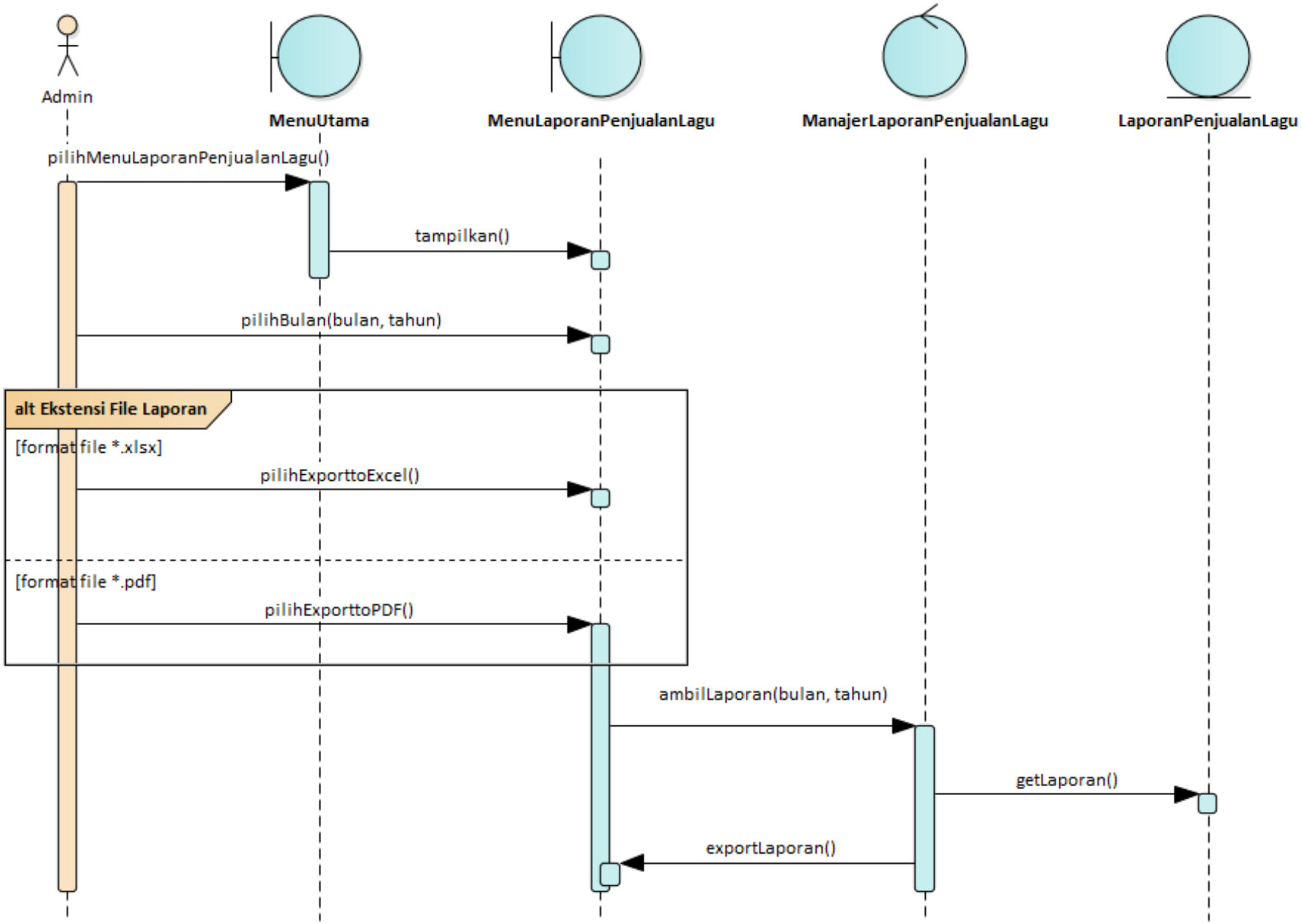
# Sequence Diagram: Mengubah Data Pribadi



# Sequence Diagram: Melakukan Pembelian Lagu



# Sequence Diagram: Mengekspor Laporan Penjualan

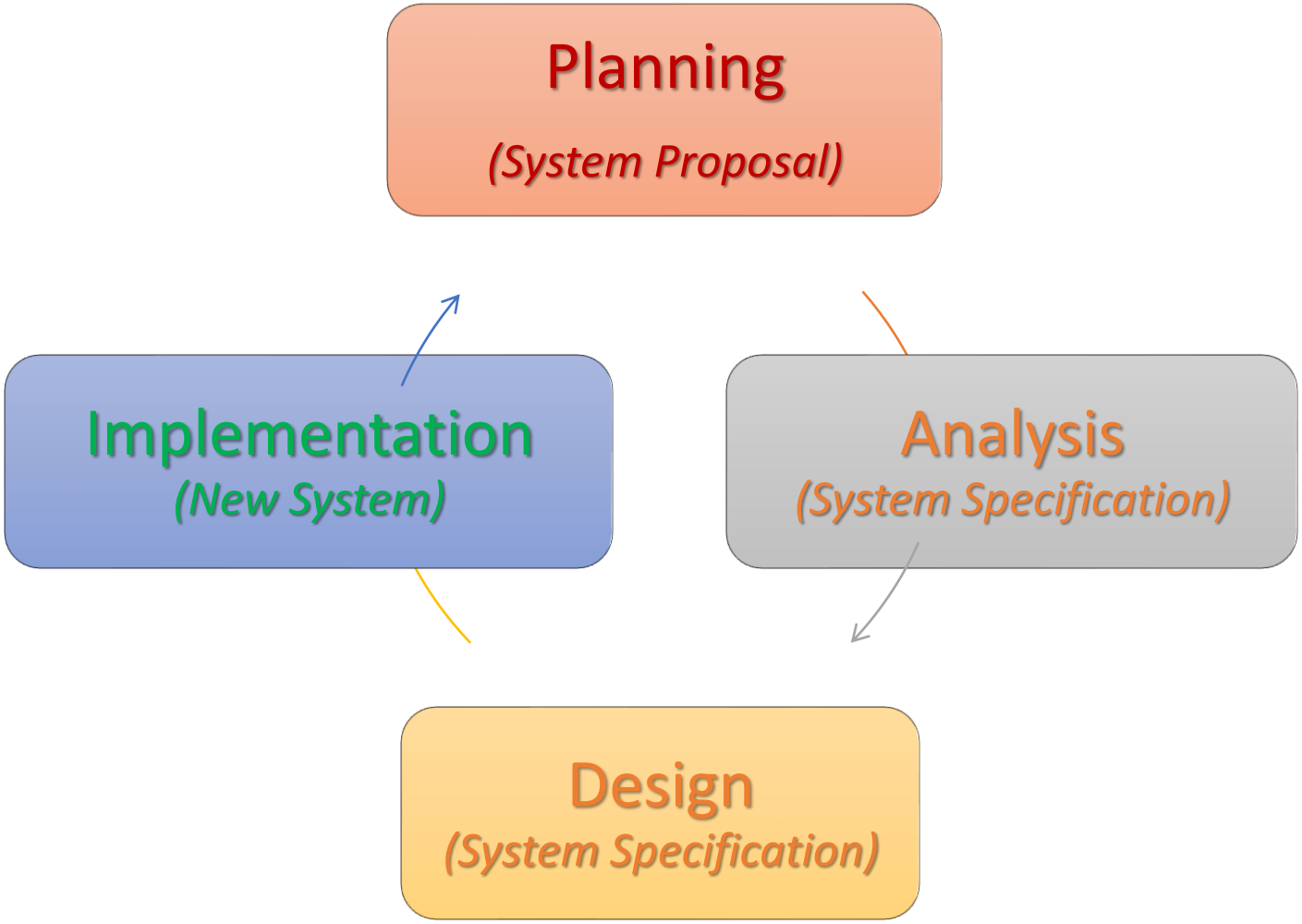


# Catatan Pola Kesalahan

- Gunakan pola Subject-Verb-Object (S-V-O) untuk use case diagram (Actor – Use Case) dan Activity Diagram (Partition – Action)
- Use Case Diagram adalah apa yang dilakukan Actor di sistem, bukan apa yang dilakukan oleh sistem
- Pada Sequence Diagram perhatikan transaksi yang harusnya datangnya dari actor, dan bukan dari object lain
- Naming untuk object dan class adalah kata benda (noun), untuk message di Sequence Diagram (method) adalah kata kerja
- Object lifeline dan message di Sequence Diagram dan nama class di Class Diagram, tidak boleh menggunakan spasi, karena akan jadi Class dan Method di kode program
- Actor (manusia) akan mengirim message hanya ke Boundary Class, tidak ke Control atau Entity Class. Sedangkan Actor (System) akan mengirim message ke Control
- Entity Class bukan consumer, jadi tidak pernah mengirim message ke Boundary atau Control Class
- Boundary class akan menjadi UI Design, entity class akan menjadi Data Model
- Class diagram tidak menunjukkan alur, tapi menunjukkan struktur dan komposisi dari sistem yg kita bangun

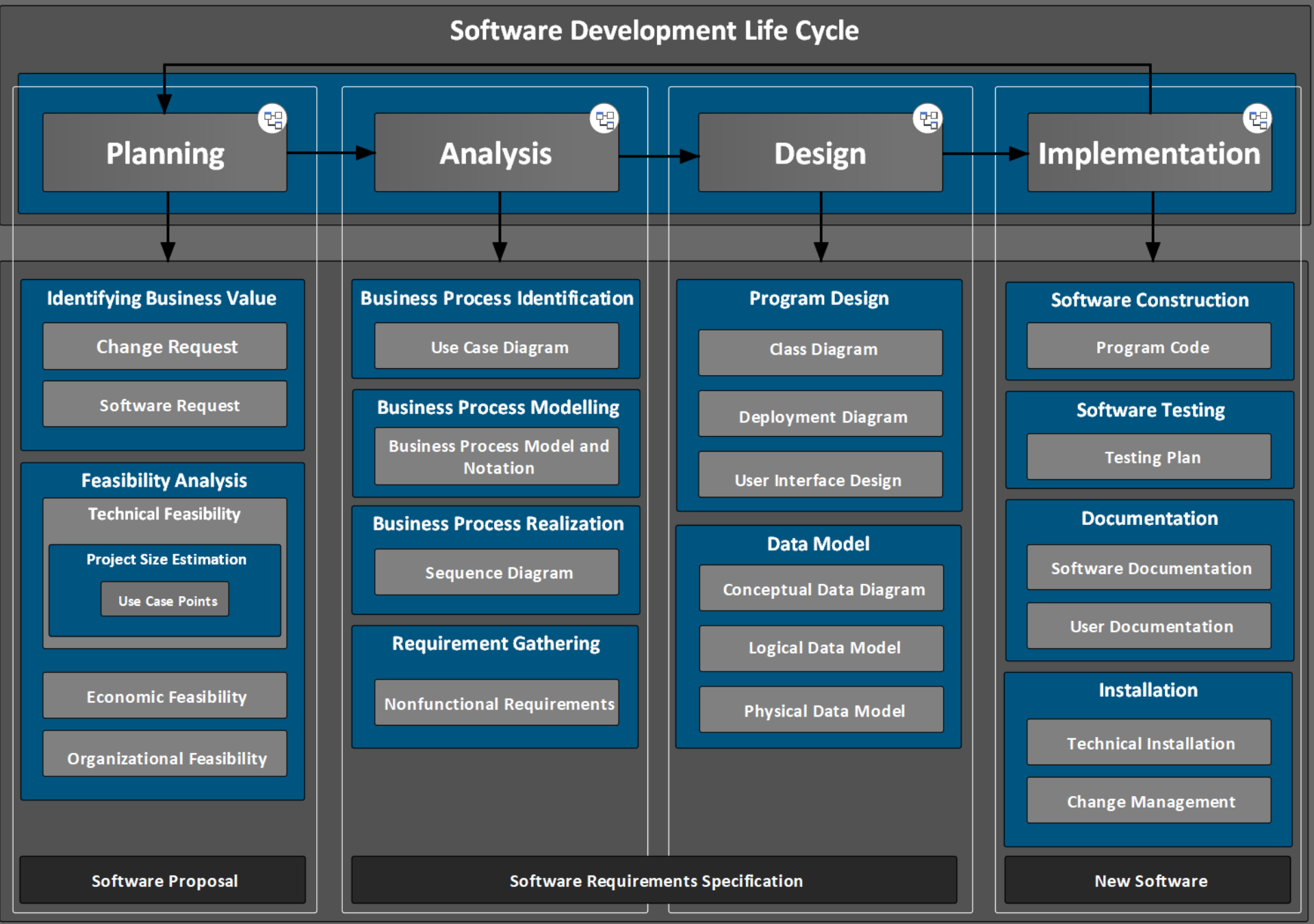
# 3. Systems Design

# Siklus Pengembangan Software




*(Tilley, 2012)* *(Dennis, 2016)* *(Valacich, 2017)*

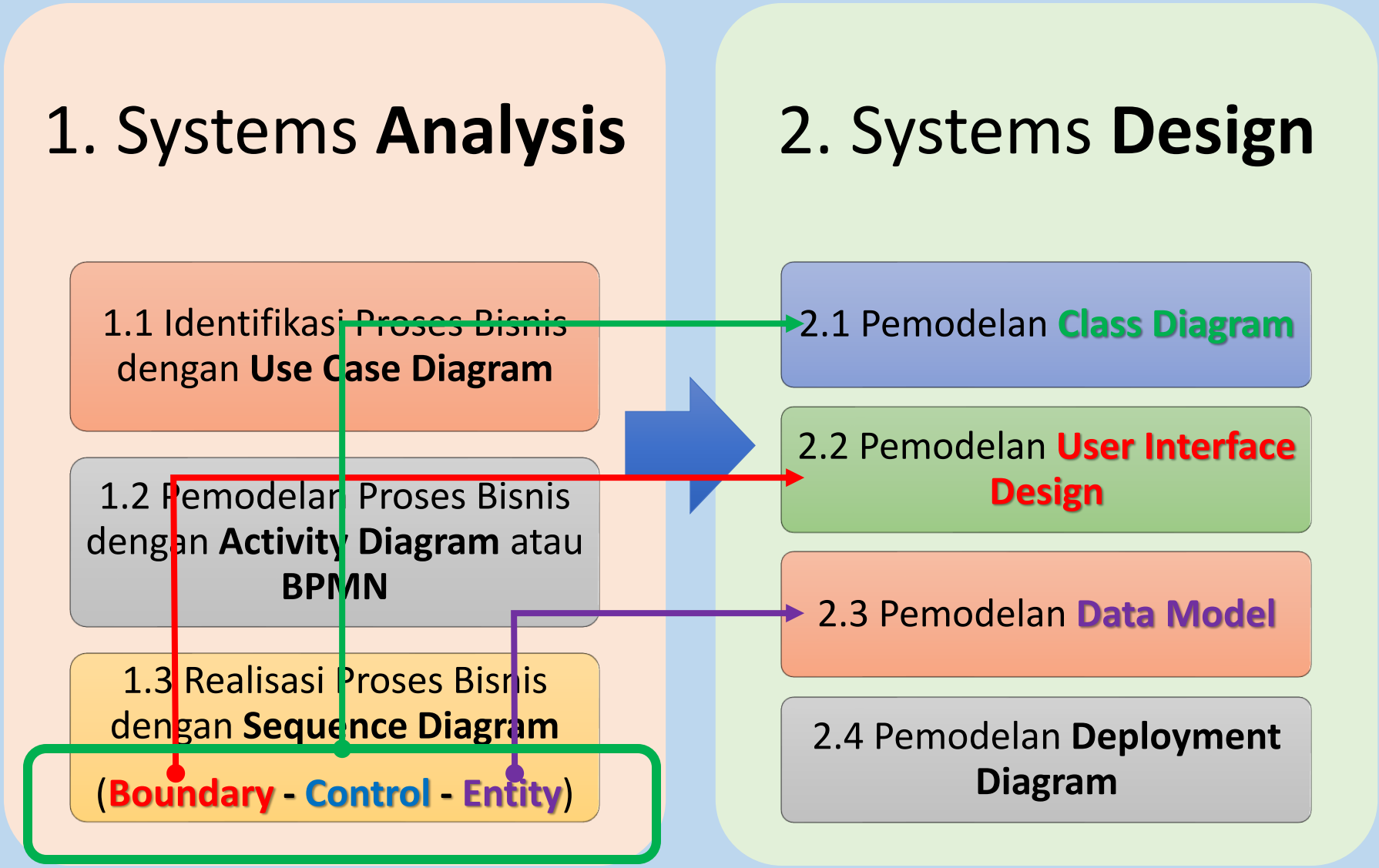
# Application Development Governance



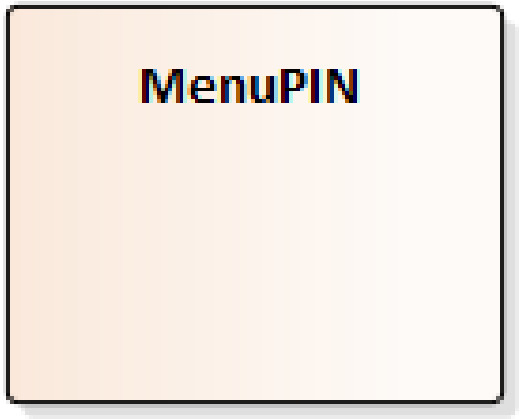
# 3.1 Perancangan Class Diagram

# UML based Software Analysis and Design

*(Wahono, 2009)*



# Class

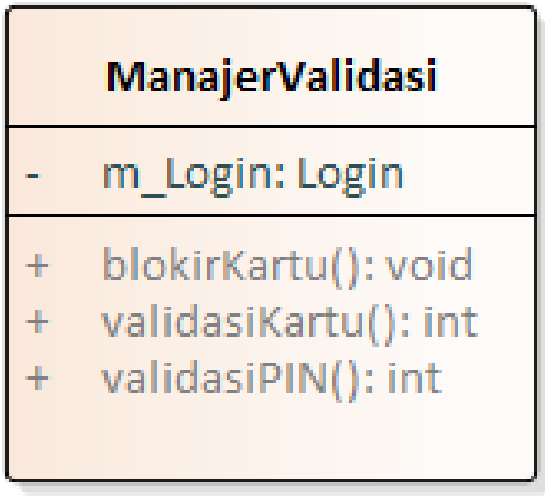



```
public class MenuPIN{

    ........

}
```

# Attributes

- Visibility of attributes
  - **+ Public**: not hidden from any object
  - **# Protected**: hidden from all but immediate subclasses
  - **– Private**: hidden from all other classes
- Default is private

# Class with Attribute and Method

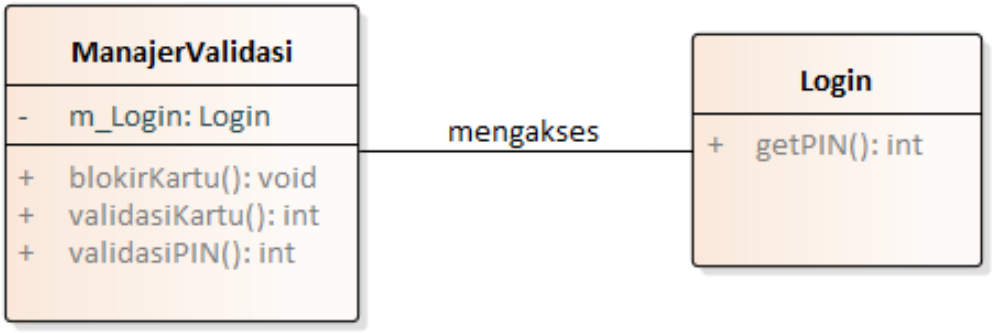


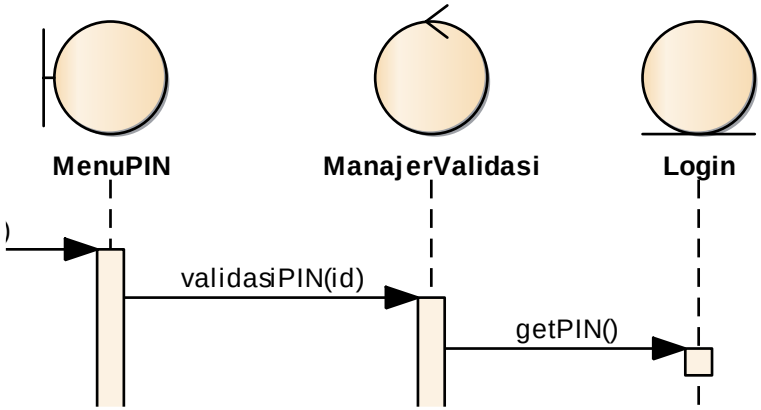

```
public class ManajerValidasi {

        private Login m_Login;

        public int validasiKartu(){
            return 0;
         }

         public int validasiPIN(){
            return 0;
         }

          public void blokirKartu(){

          }
   }
```

# Relationships

1. Generalization
   - "Is-A" relationship
   - Enables inheritance of attributes & operations
   - Subclasses and superclasses
2. Aggregation
   - "Has-A" relationship
   - Relates parts to wholes
   - Uses decomposition
3. Composition
   - "Has-A" relationship
   - Similar but stronger than Aggregation
4. Association
   - Relationships that don't fit "Is-A" or "Has-A"
   - Often a weaker form of "Has-A"
   - Miscellaneous relationships between classes

# Class Association

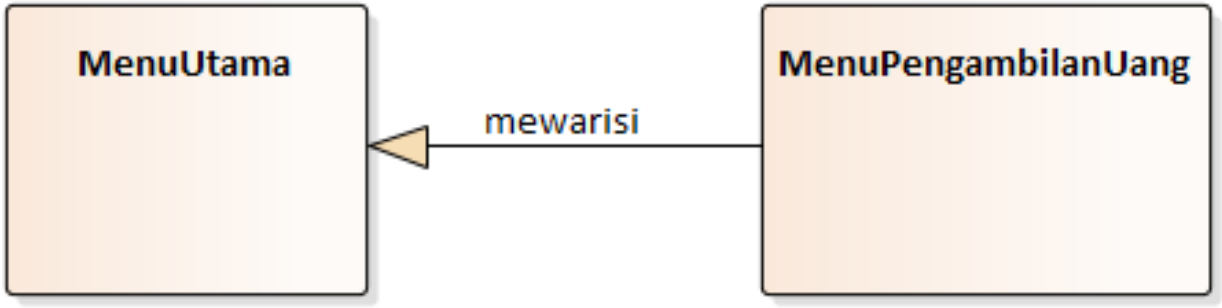



```
public class ManajerValidasi {

        private Login m_Login;

        public int validasiKartu(){
              return 0;
         }

         public int validasiPIN(){
              int pin = m_login.getPIN();
              return 0;
         }

          public void blokirKartu(){

          }
}
```

```
public class Login {


        public int getPIN(){
              return 0;
         }

}
```

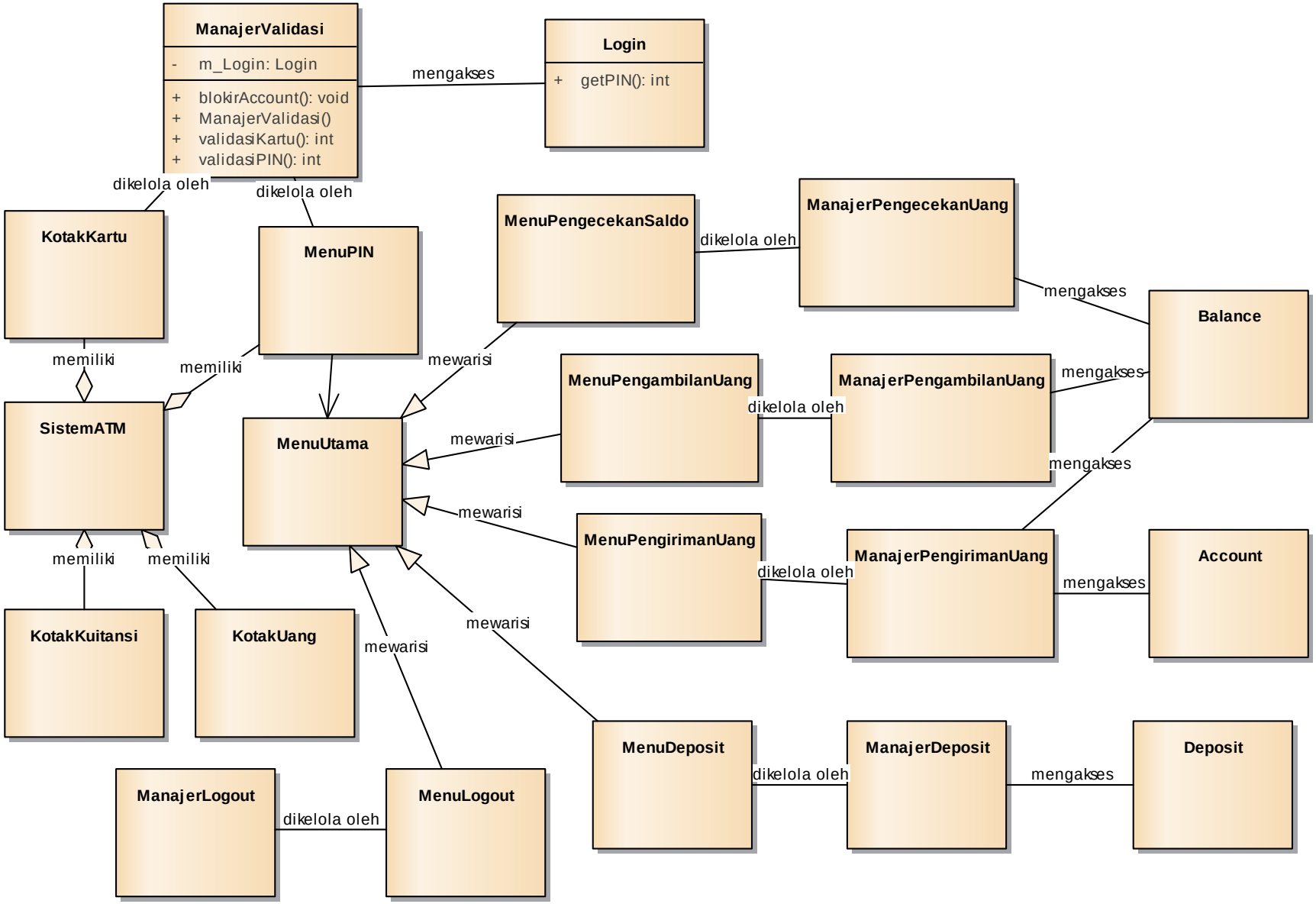
# Class Inheritance



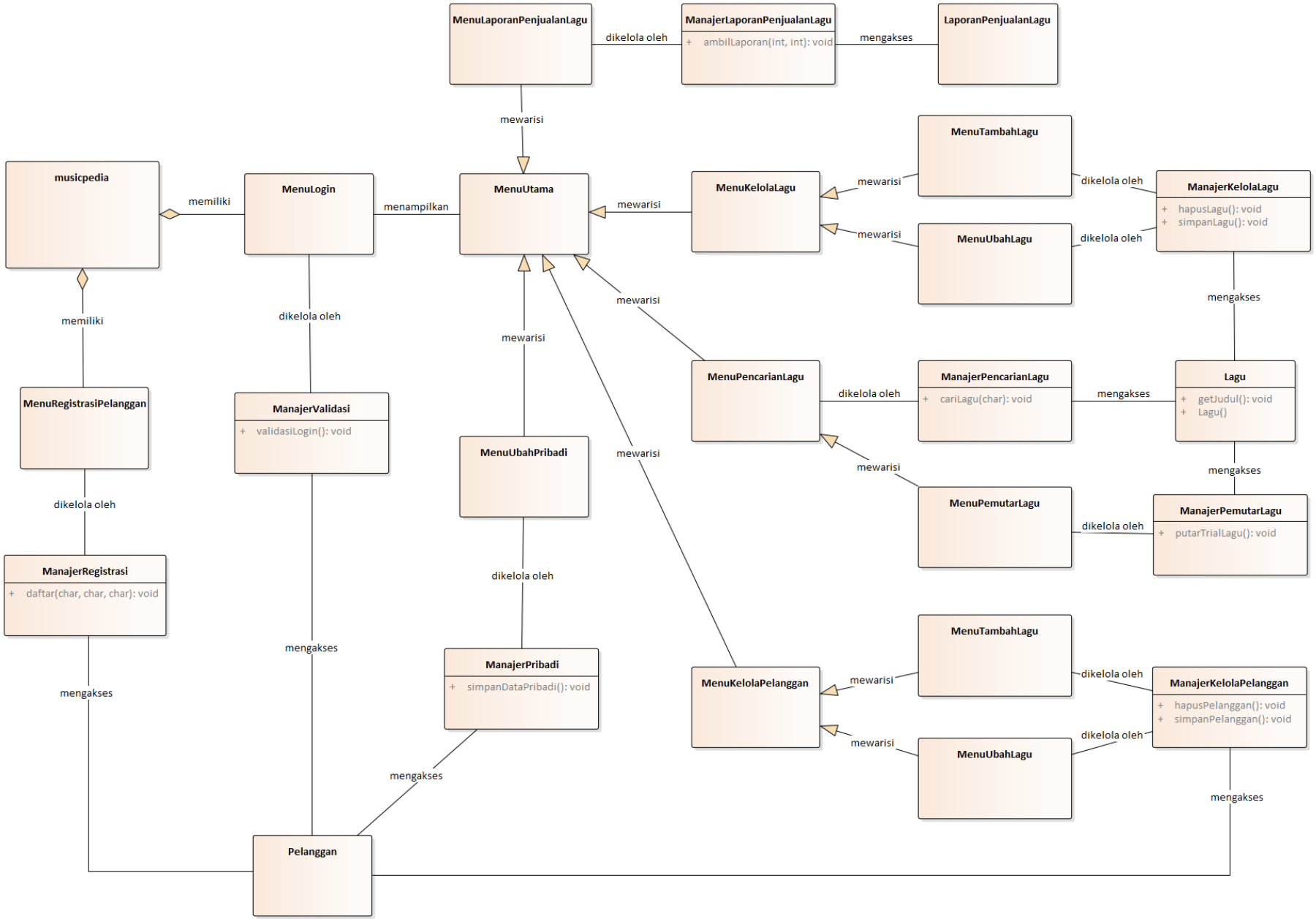

```
public class MenuUtama {

    ......

}
```

```
public class MenuPengambilanUang
        extends MenuUtama {

    .....

}
```

# Class Diagram: Sistem ATM



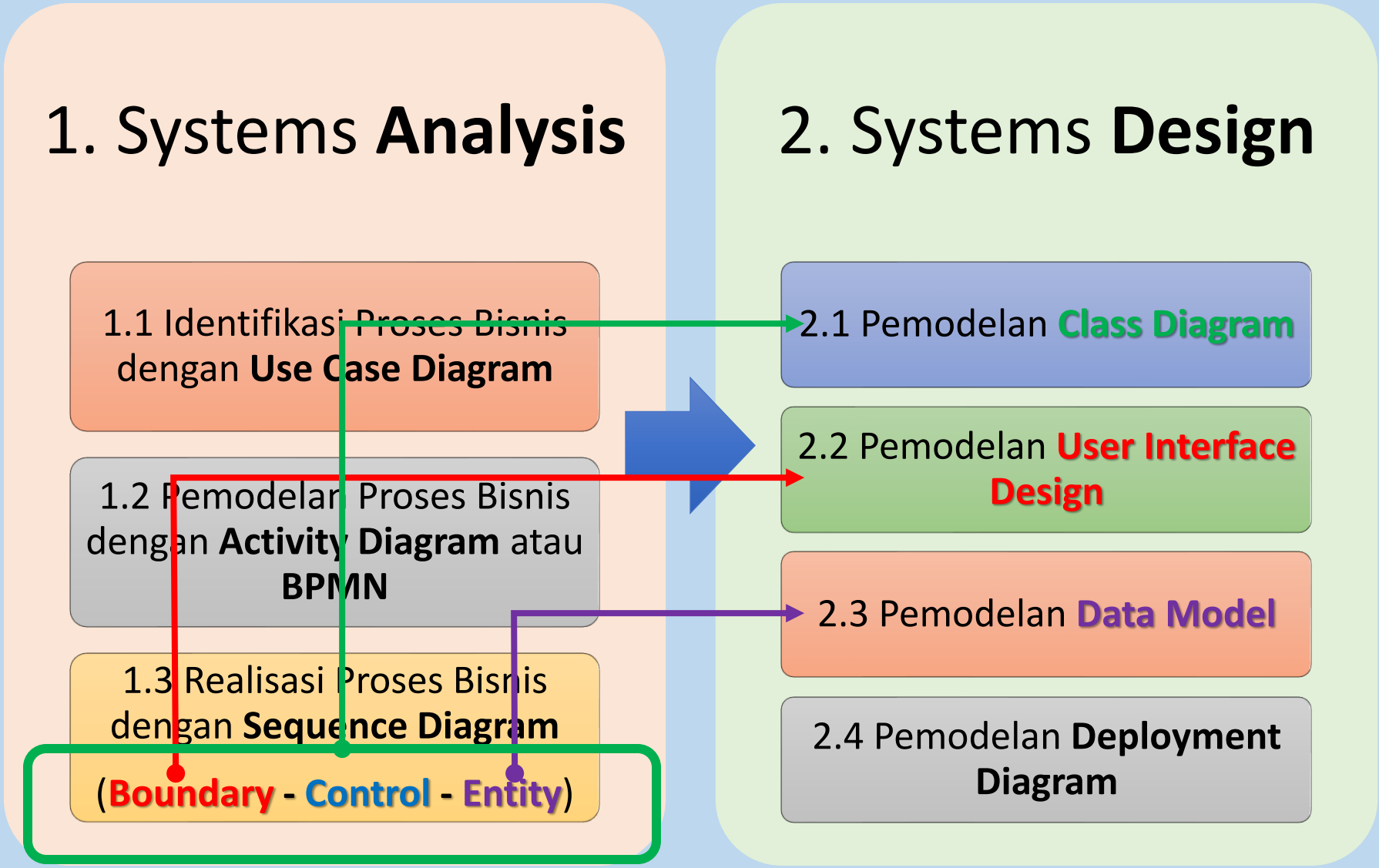
# Class Diagram: MusicPedia



# 3.2 Perancangan User Interface Design

# UML based Software Analysis and Design

*(Wahono, 2009)*

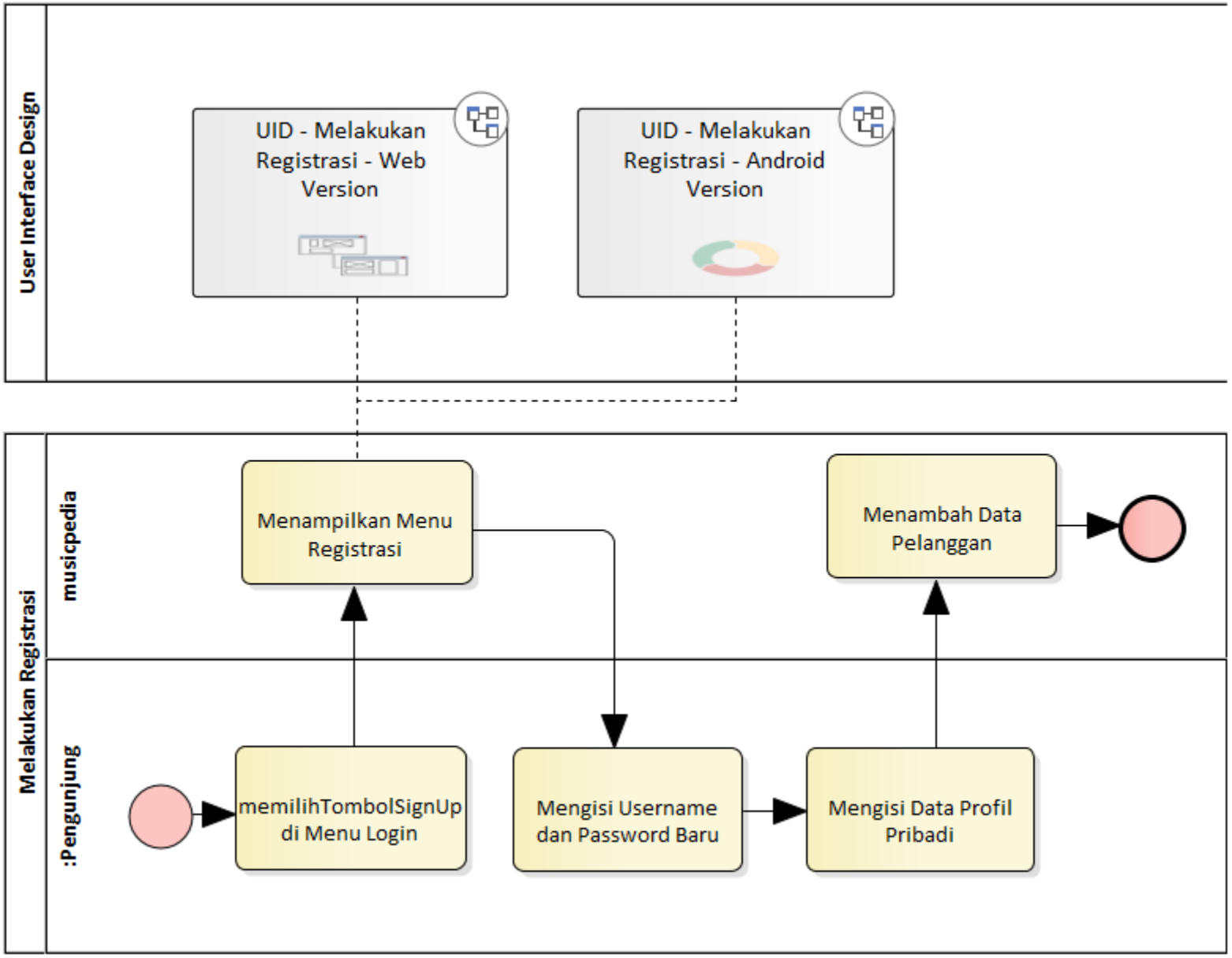


# Philosophy and Principles

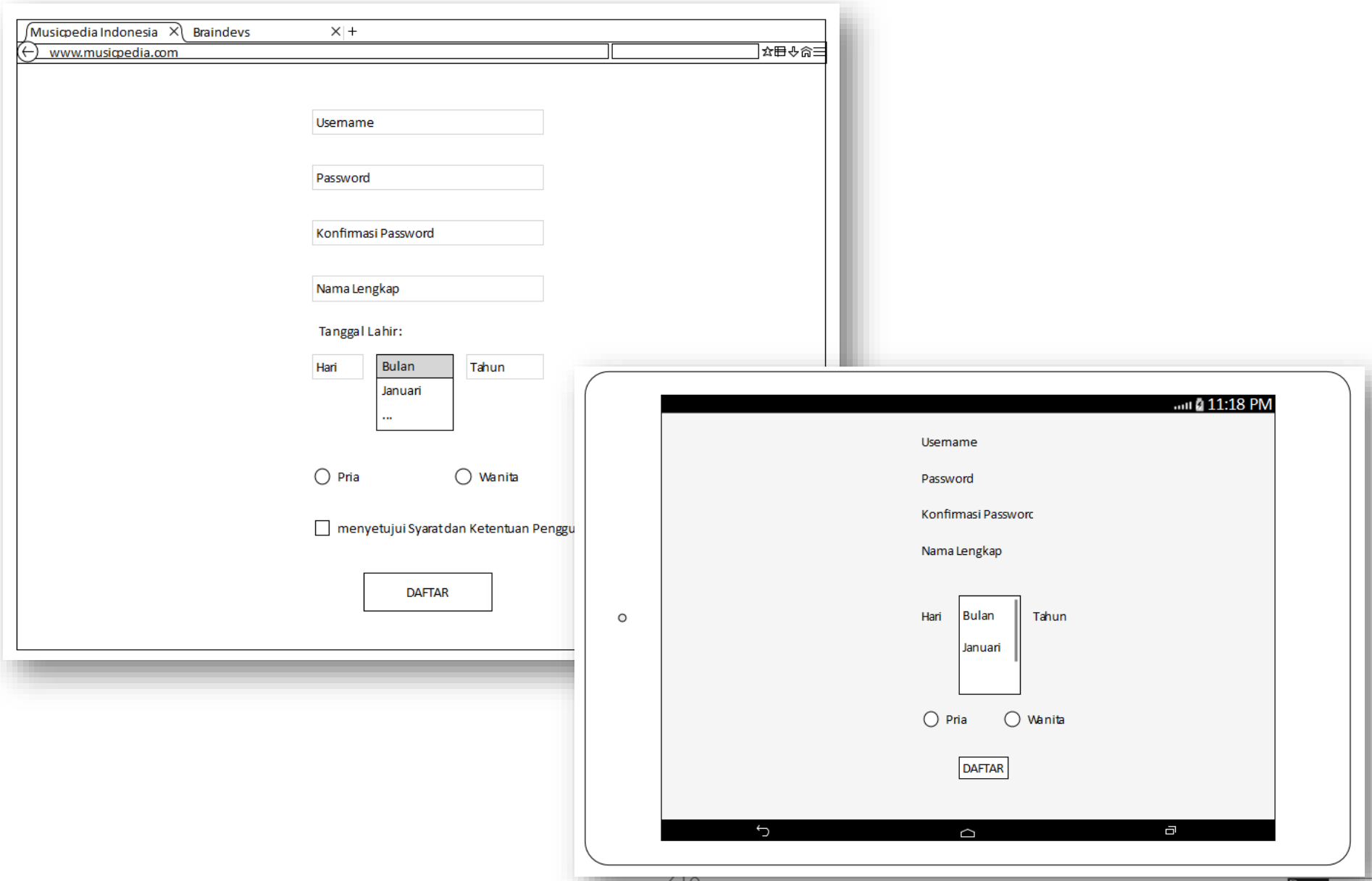
- Interface design is an art
- Balance between
  - Making the interface useful and
  - Presenting too much information

- Principles for User Interface Design:
  1. Layout: Consistent use of screen area
  2. Content awareness: Users know where they are
  3. Aesthetics: White space vs. functionality
  4. User experience: Ease of use vs. learning curve
  5. Consistency: User can predict what will happen for each action
  6. Minimal user effort: Simple to use, three click rule

# BPMN Melakukan Registrasi



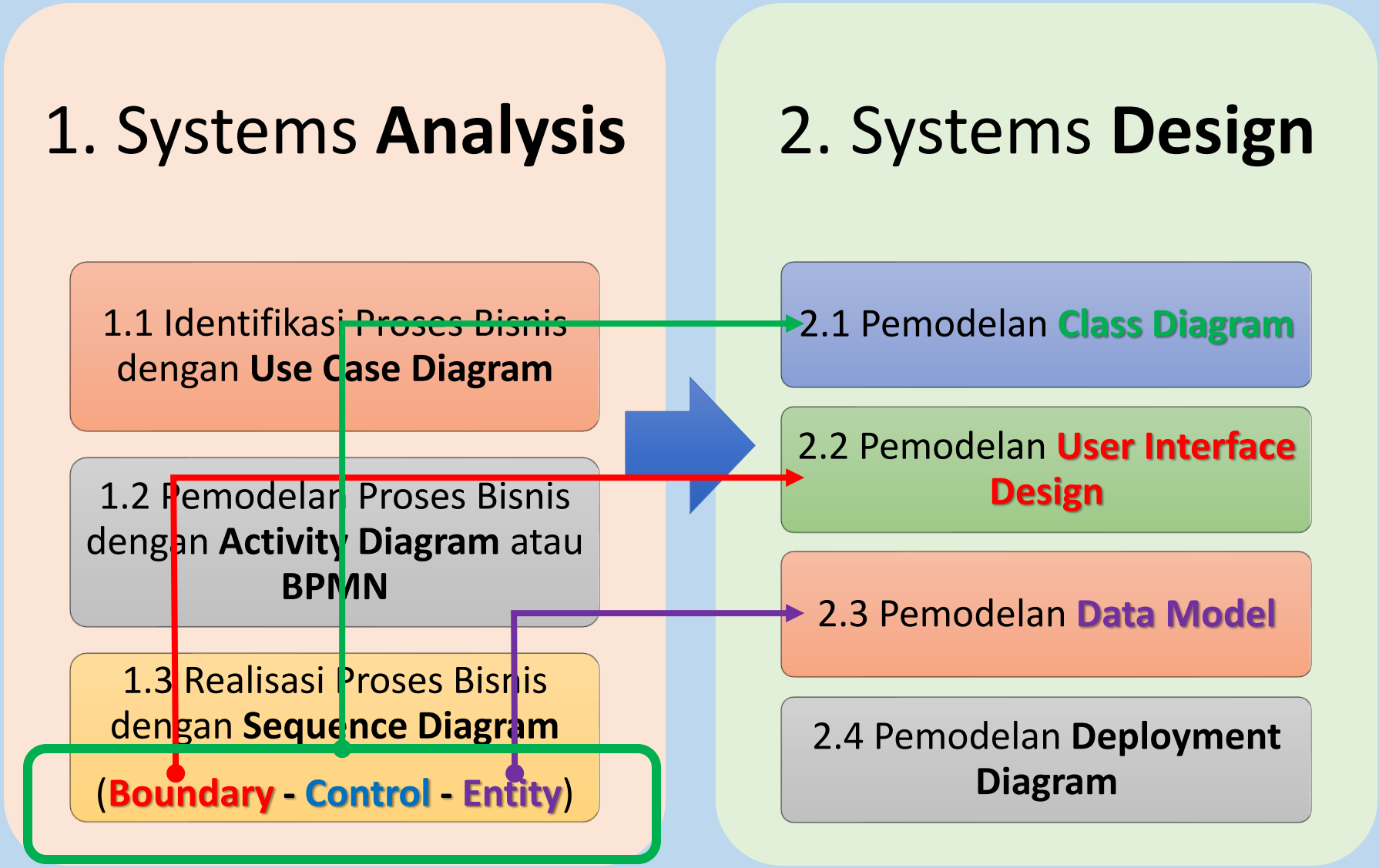
# User Interface Design Melakukan Registrasi (versi Web dan versi Android)



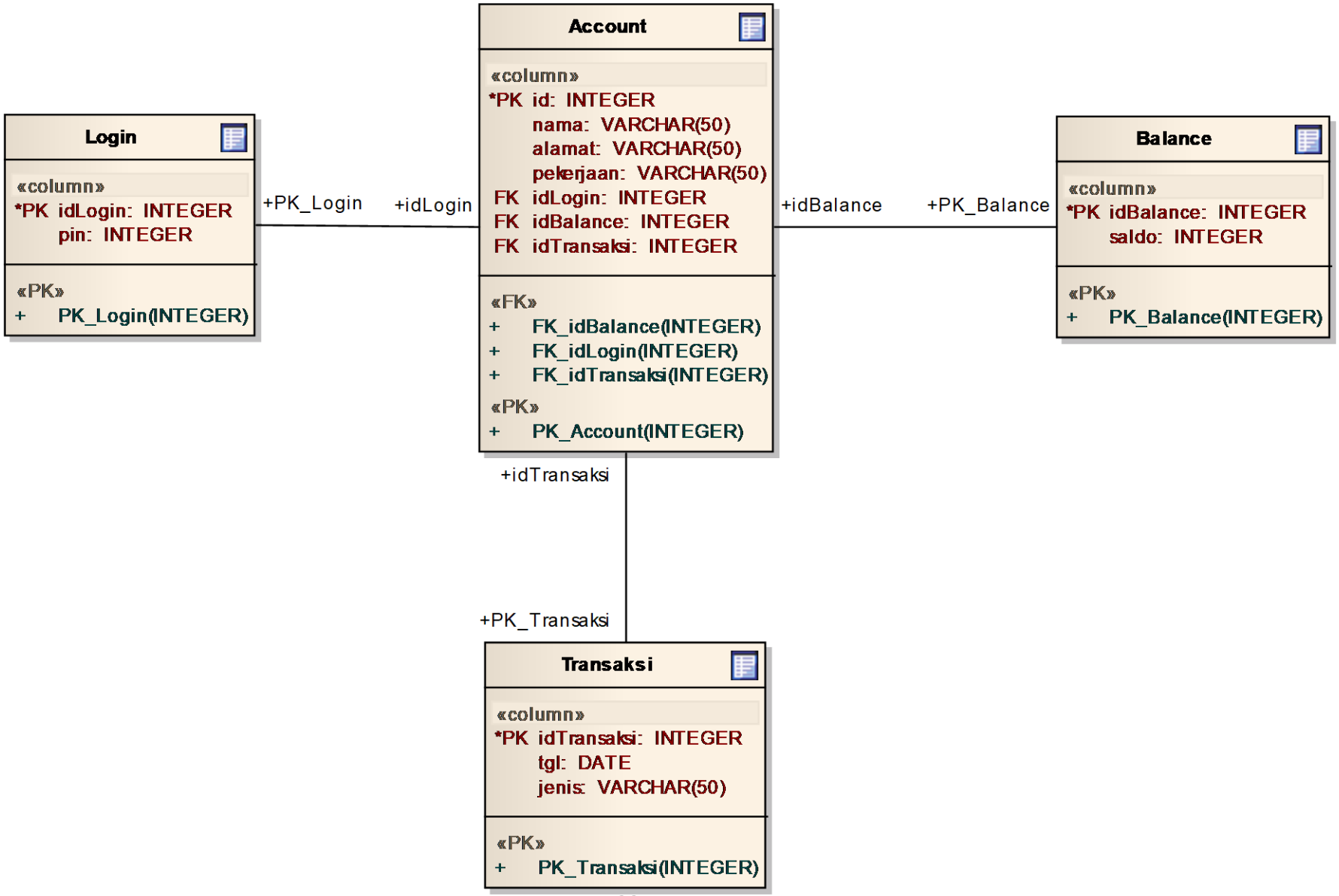
# 3.3 Perancangan Data Model

# UML based Software Analysis and Design

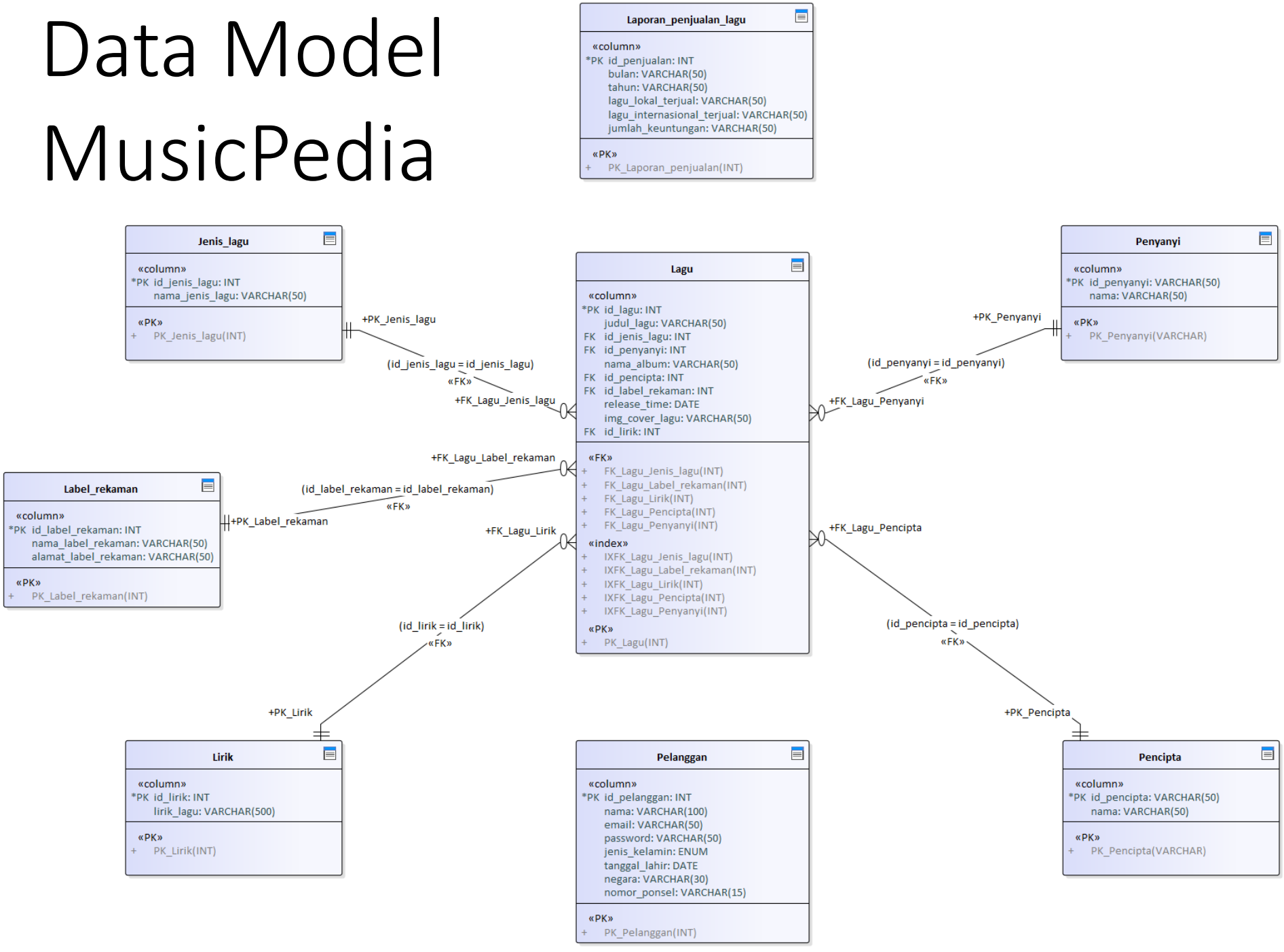
*(Wahono, 2009)*



# Data Model Sistem ATM



# Data Model MusicPedia



# 3.4 Perancangan Deployment Diagram

# Deployment Diagram

- Servers
  - Mainframes, Minis, Micros
- Clients
  - Input/Output HW used by users
  - Terminals, PCs, special purpose HW
- Network
  - HW and SW to connect clients to servers
- Nodes
  - Any piece of hardware in the model
  - A computational resource
  - Labeled by its name
  - Stereotype to label the type of node

Node

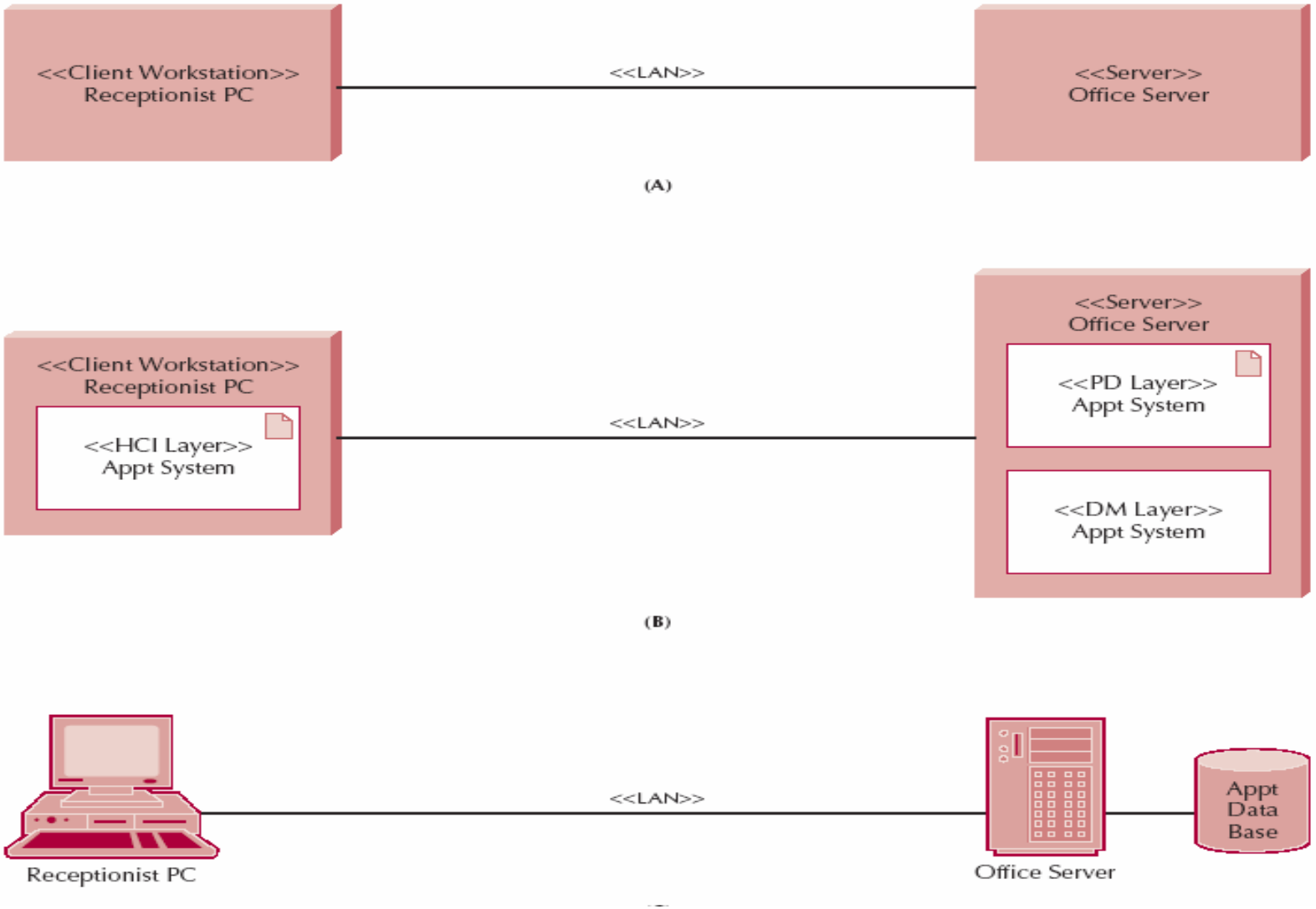

Artifact


Node with Deployment Artifact

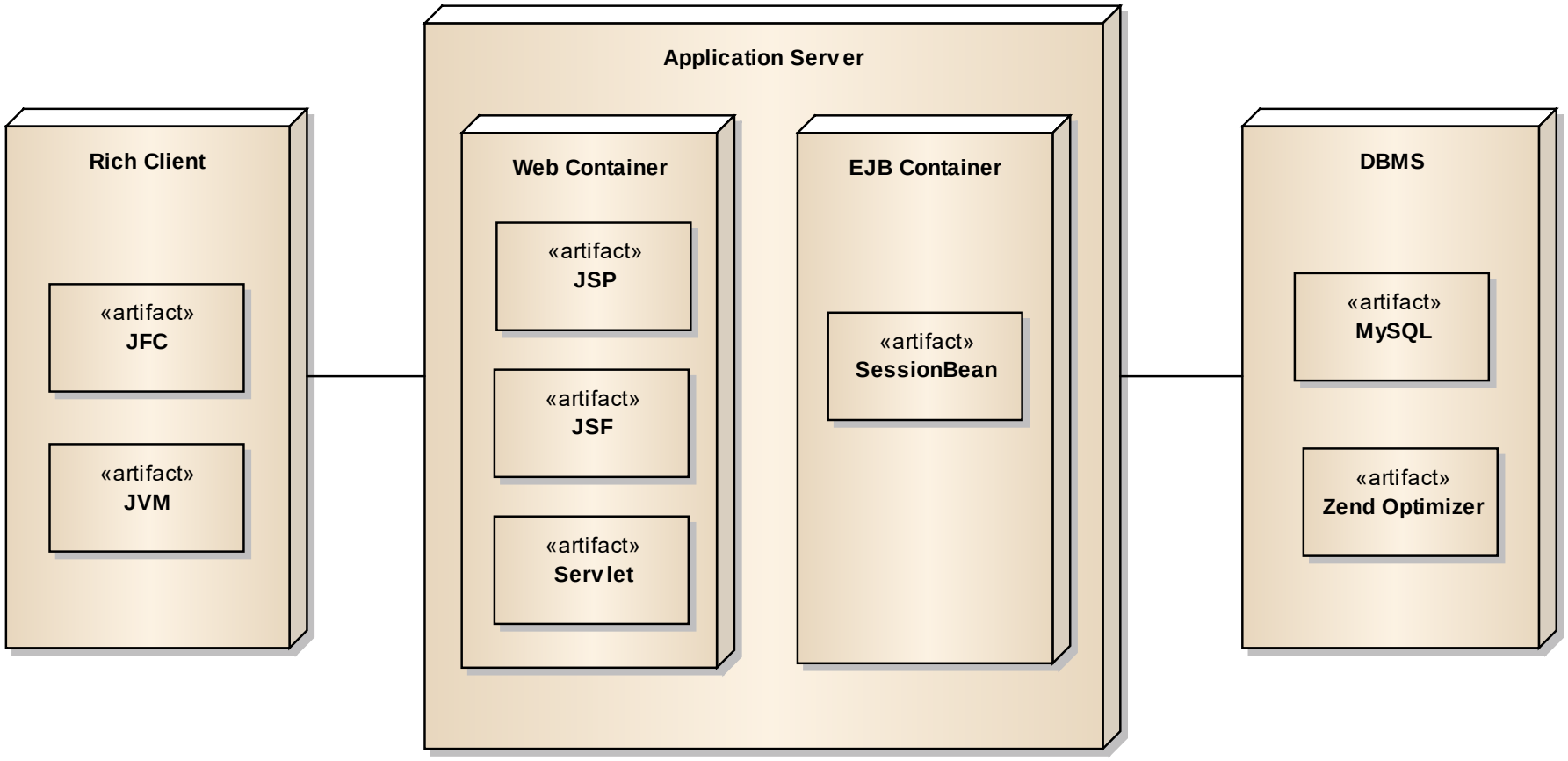

Communication Path


- Artifacts
  - Piece of the information system, such as software or a database table
- Node with Deployed Artifact
  - Shows artifact placed on a physical node
  - Good for showing distribution data or software
- Communication paths
  - Links between nodes of the network

# Diagram Examples

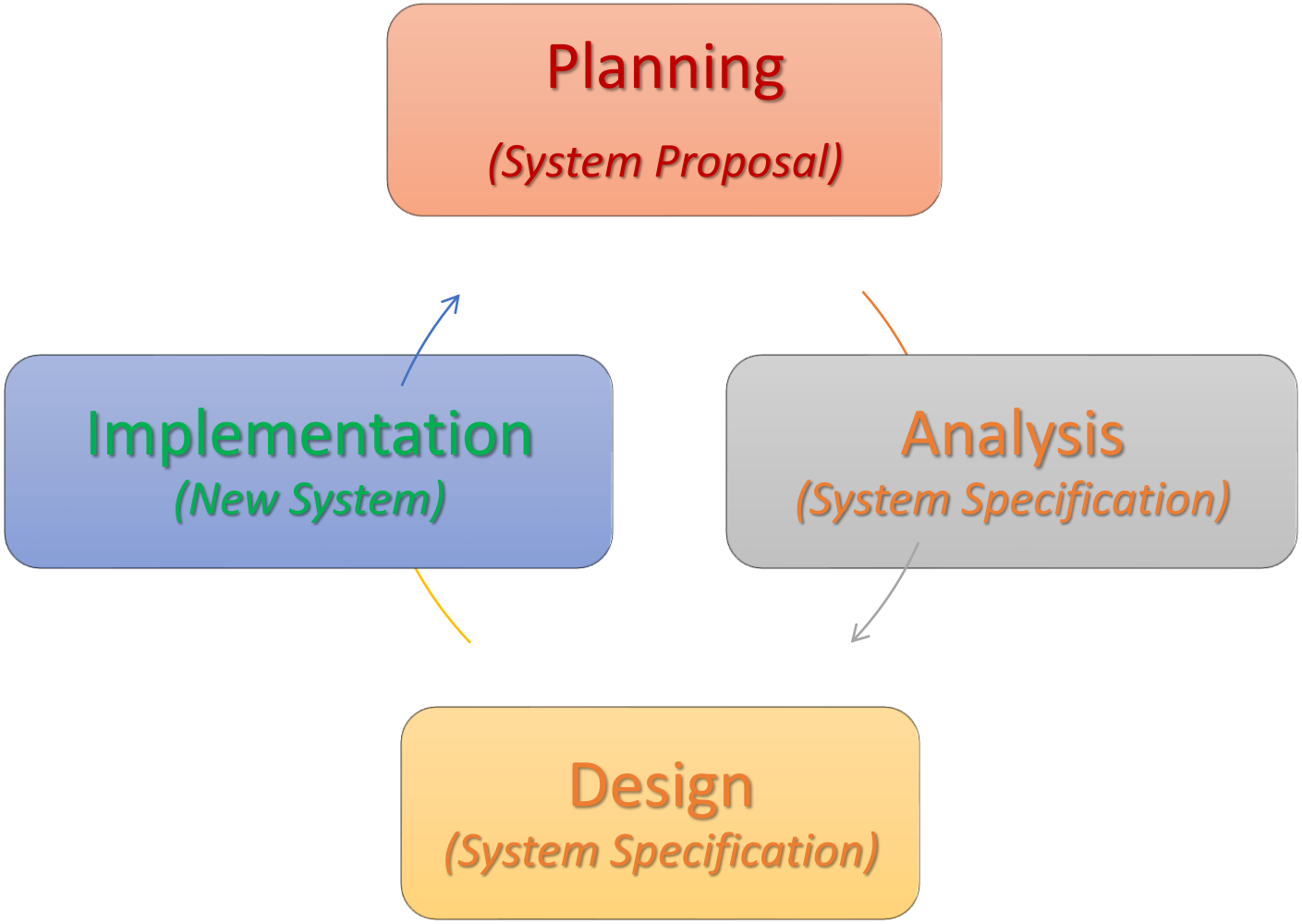


# Deployment Diagram (3 Tier)



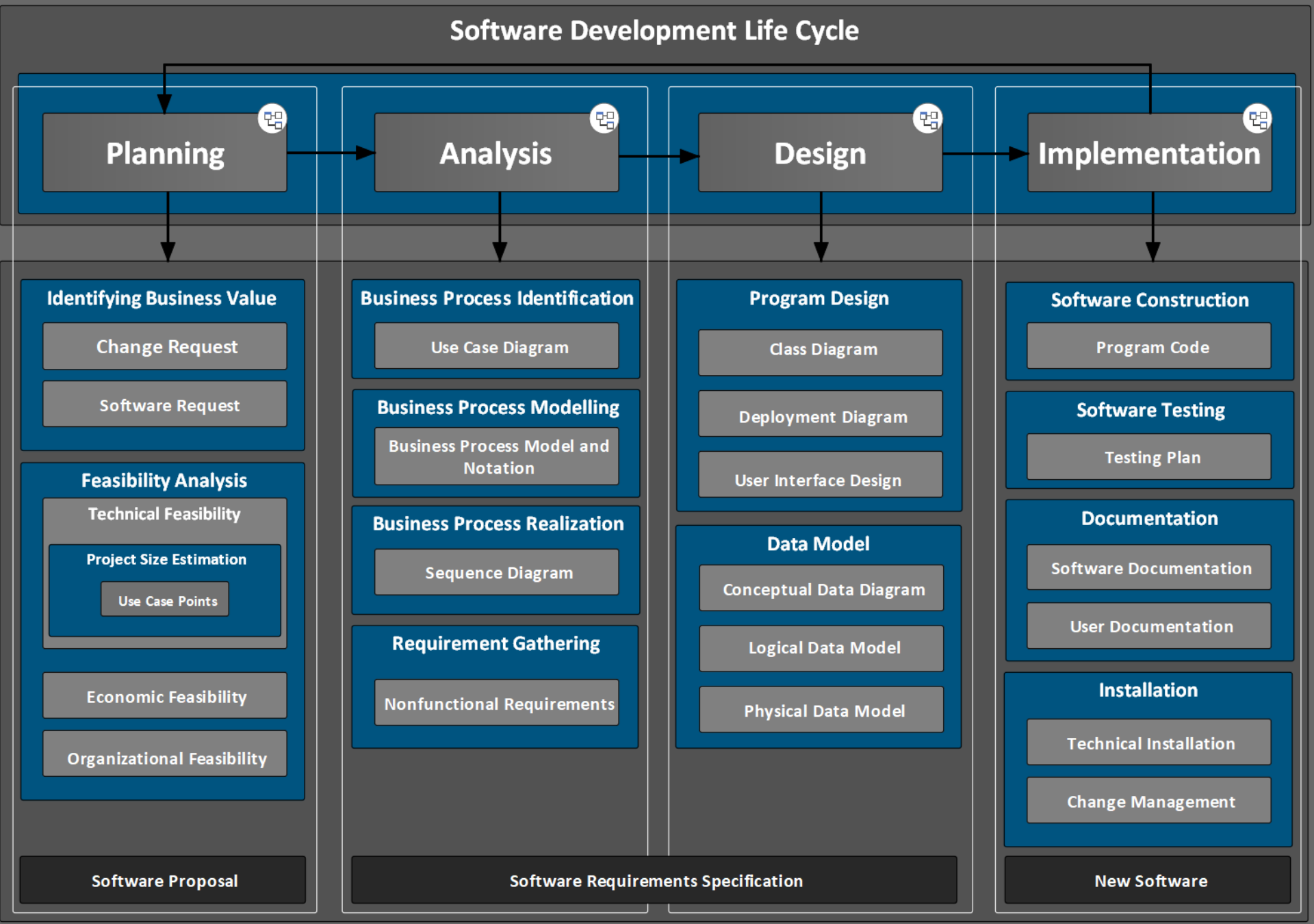
# Rangkuman Studi Kasus Sistem ATM

# Siklus Pengembangan Software

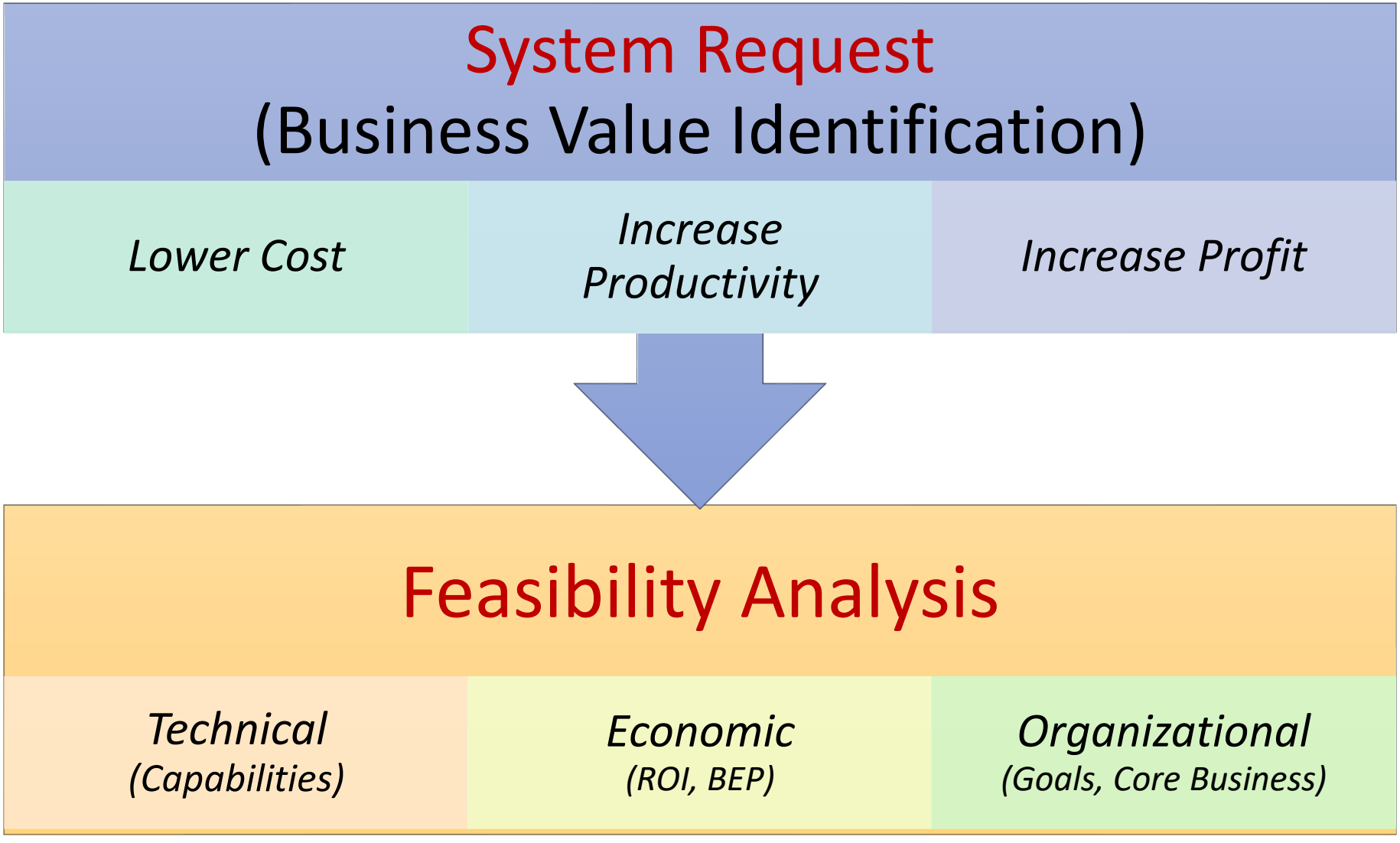



*(Tilley, 2012)* *(Dennis, 2016)* *(Valacich, 2017)*

# Application Development Governance



# Planning



# System Request: Sistem Penjualan Musik Online

**Project Sponsor:** Margaret Mooney, Vice President of Marketing

**Business Needs:** Project ini dibangun untuk:

1. Mendapatkan pelanggan baru lewat Internet
2. Memberikan layanan pendukung dengan menggunakan internet

**Business Requirements:**

Sistem yang mendukung penjualan musik secara online. Fitur-fitur yang harus ada:

1. Fitur Pencarian Produk
2. Fitur Pencarian Toko yang Menyediakan Stok Produk
3. Fitur Pemesanan Produk Melalui Toko yang Menyediakan
4. Fitur Pembayaran dengan Berbagai Pilihan Pembayaran

**Business Value:**

*Intangible Value:*

- Meningkatkan kenyamanan dan kepuasan pelanggan
- Meningkatkan brand recognition tentang perusahaan di dunia Internet

*Tangible Value:*

1. Meningkatkan penjualan dari pelanggan baru lewat Internet:
   - Rp 400 juta peningkatan penjualan dari pelanggan baru dan Rp 600 juta dari pelanggan lama
2. Mengurangi biaya operasional untuk menangani komplain dari pelanggan
   - Rp 100 juta pengurangan tahunan biaya telepon untuk menangani pelanggan

# Studi Kelayakan Sistem Penjualan Musik Online

Margaret Mooney dan Alec Adams membuat studi kelayakan untuk pengembangan Sistem Penjualan Musik Online

## Kelayakan Teknis

Sistem penjualan musik online layak secara teknis, meskipun memiliki beberapa risiko.

Risiko Berhubungan dengan **Kefamilieran dengan Aplikasi**: Resiko Tinggi

- Divisi Marketing tidak memiliki pengalaman menggunakan sistem penjualan online
- Divisi IT memiliki pemahaman yang baik tentang sistem penjualan offline, akan tetapi tidak berpengalaman mengembangkan sistem penjualan musik online

Risiko Berhubungan dengan **Kefamilieran dengan Teknologi**: Resiko Sedang

- Divisi IT tidak menguasai masalah infrastruktur dan ISP, tetapi akan menyewa konsultan
- Divisi IT cukup familier dengan framework dan IDE yang akan digunakan
- Divisi Marketing tidak memiliki pengalaman menggunakan teknologi Web

Risiko berhubungan dengan **Ukuran Project**: Risiko Rendah

- Perusahaan memiliki total 30 orang pengembang
- Project dikerjakan oleh 5 orang pengembang dengan estimasi waktu 6 bulan

**Kompatibilitas** dengan sistem dan infrastruktur yang ada: Risiko Rendah

- Sistem pemesanan yang ada sekarang menggunakan *open standard*, jadi sangat kompatibel dengan sistem penjualan berbasis web yang akan dibangun

## Kelayakan Ekonomi

Cost benefit analysis telah dilakukan. Sistem Penjualan musik online memiliki peluang yang baik untuk bisa meningkatkan pendapatan perusahaan.

- Return on Investment (ROI) setelah 3 tahun: **31%**
- Break-even point (BEP): **2.25 tahun**
- Total keuntungan setelah 3 tahun: **Rp. 503.559.986,-**

Keuntungan Intangible

- Meningkatkan kepuasaan pelanggan
- Meningkatkan branding perusahaan

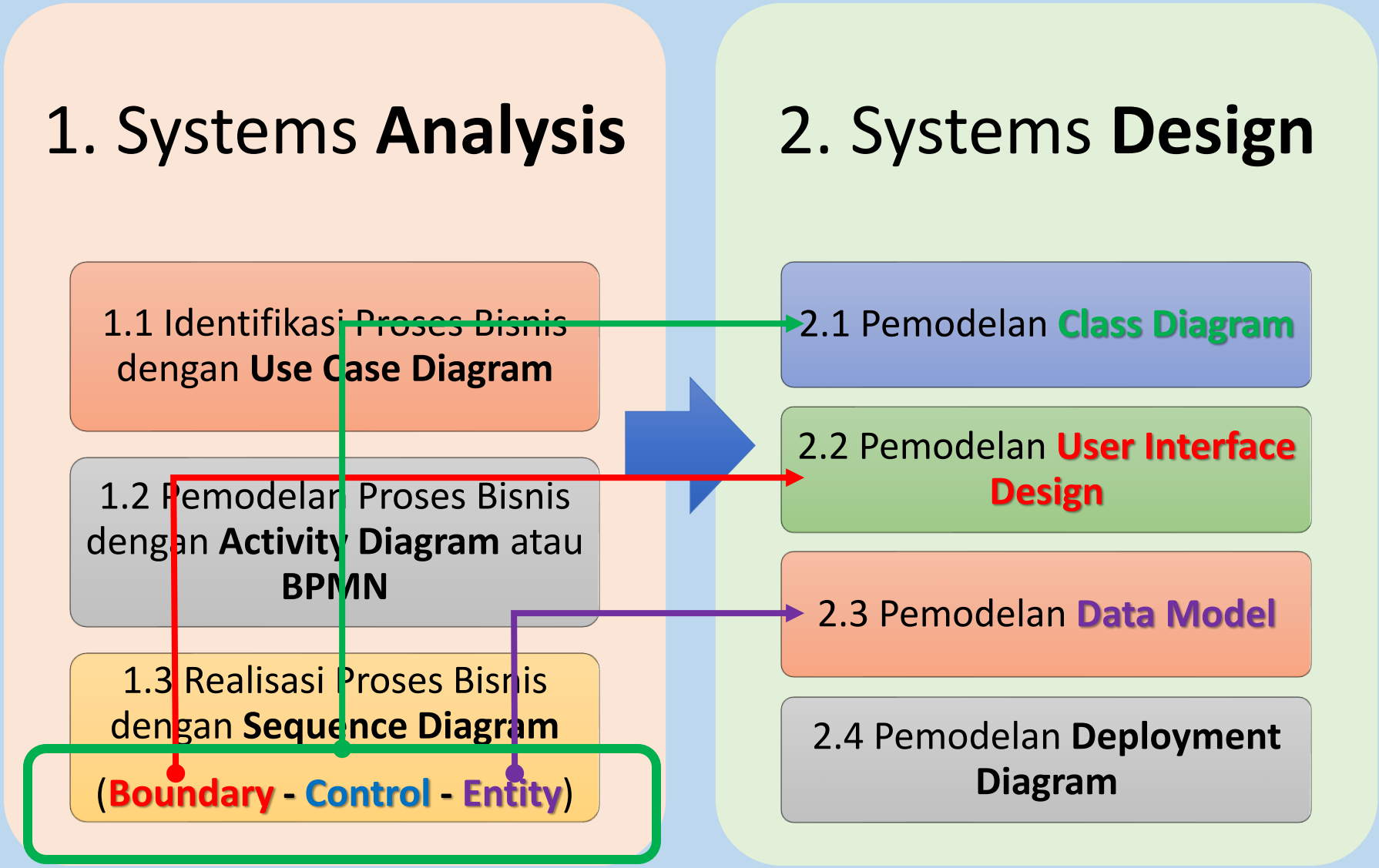
## Kelayakan Organisasi

- Secara organisasi, resikonya rendah. Tujuan dari pengembangan sistem penjualan musik online adalah meningkatkan penjualan perusahaan. Dan ini selaras dengan KPI marketing yang ke arah peningkatan kuantitas penjualan
- Project champion dari pengembangan sistem penjualan musik online ini adalah Margaret Mooney, Vice President of Marketing

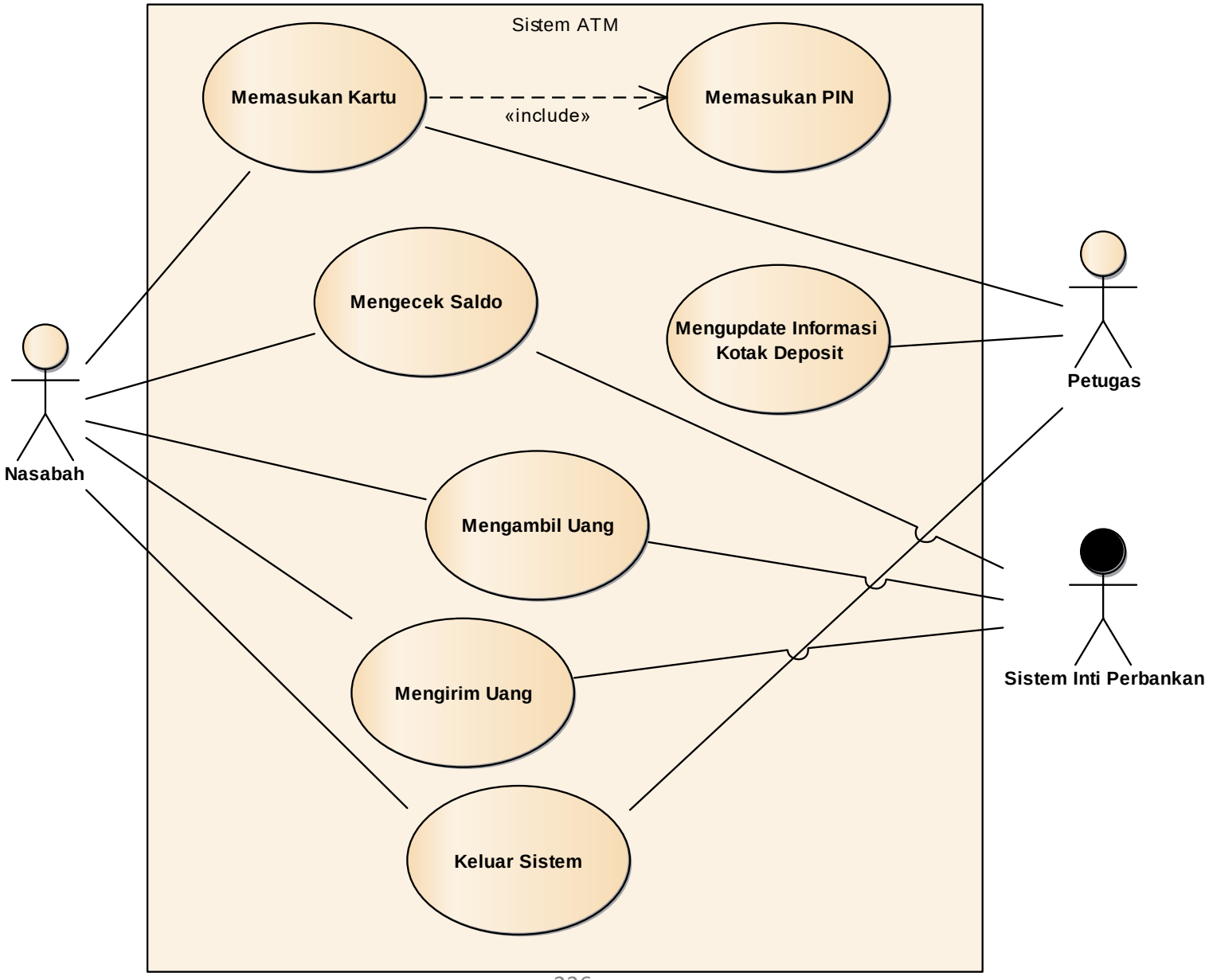
| | 2019 | 2020 | 2021 |
|---|---|---|---|
| Peningkatan penjualan dari pelanggan baru | 0 | 400,000,000 | 500,000,000 |
| Peningkatan penjualan dari pelanggan lama | 0 | 600,000,000 | 700,000,000 |
| Pengurangan biaya operasional dan telepon | 0 | 100,000,000 | 100,000,000 |
| **Total Benefits:** | **0** | **1,100,000,000** | **1,300,000,000** |
| **PV of Benefits:** | **0** | **978,996,084** | **1,091,505,068** |
| **PV of All Benefits:** | **0** | **978,996,084** | **2,070,501,152** |
| Honor Tim (Analysis, Design and Implementation) | 360,000,000 | 0 | 0 |
| Honor Konsultan | 90,000,000 | 0 | 0 |
| **Total Development Costs:** | **450,000,000** | **0** | **0** |
| Honor Pengelola Web | 60,000,000 | 70,000,000 | 80,000,000 |
| Biaya Lisensi Software | 50,000,000 | 60,000,000 | 70,000,000 |
| Hardware upgrades | 100,000,000 | 100,000,000 | 100,000,000 |
| Biaya Komunikasi | 20,000,000 | 30,000,000 | 40,000,000 |
| Biaya Marketing | 100,000,000 | 200,000,000 | 300,000,000 |
| **Total Operational Costs:** | **330,000,000** | **460,000,000** | **590,000,000** |
| **Total Costs:** | **780,000,000** | **460,000,000** | **590,000,000** |
| **PV of Costs:** | **735,849,057** | **409,398,362** | **495,375,377** |
| **PV of all Costs:** | **735,849,057** | **1,145,247,419** | **1,640,622,796** |
| **Total Project Costs Less Benefits:** | **-780,000,000** | **640,000,000** | **710,000,000** |
| **Yearly NPV:** | **-735,849,057** | **569,597,722** | **669,811,321** |
| **Cumulative NPV:** | **-735,849,057** | **-166,251,335** | **503,559,986** |
| **Return on Investment (ROI) di Tahun 3: 30.70%** | **-100.00%** | **-0.145166304** | **0.306932213** |
| **Break-even Point (BEP): 2.25 tahun** | 234 | | **2.248206218** |

# UML based Software Analysis and Design

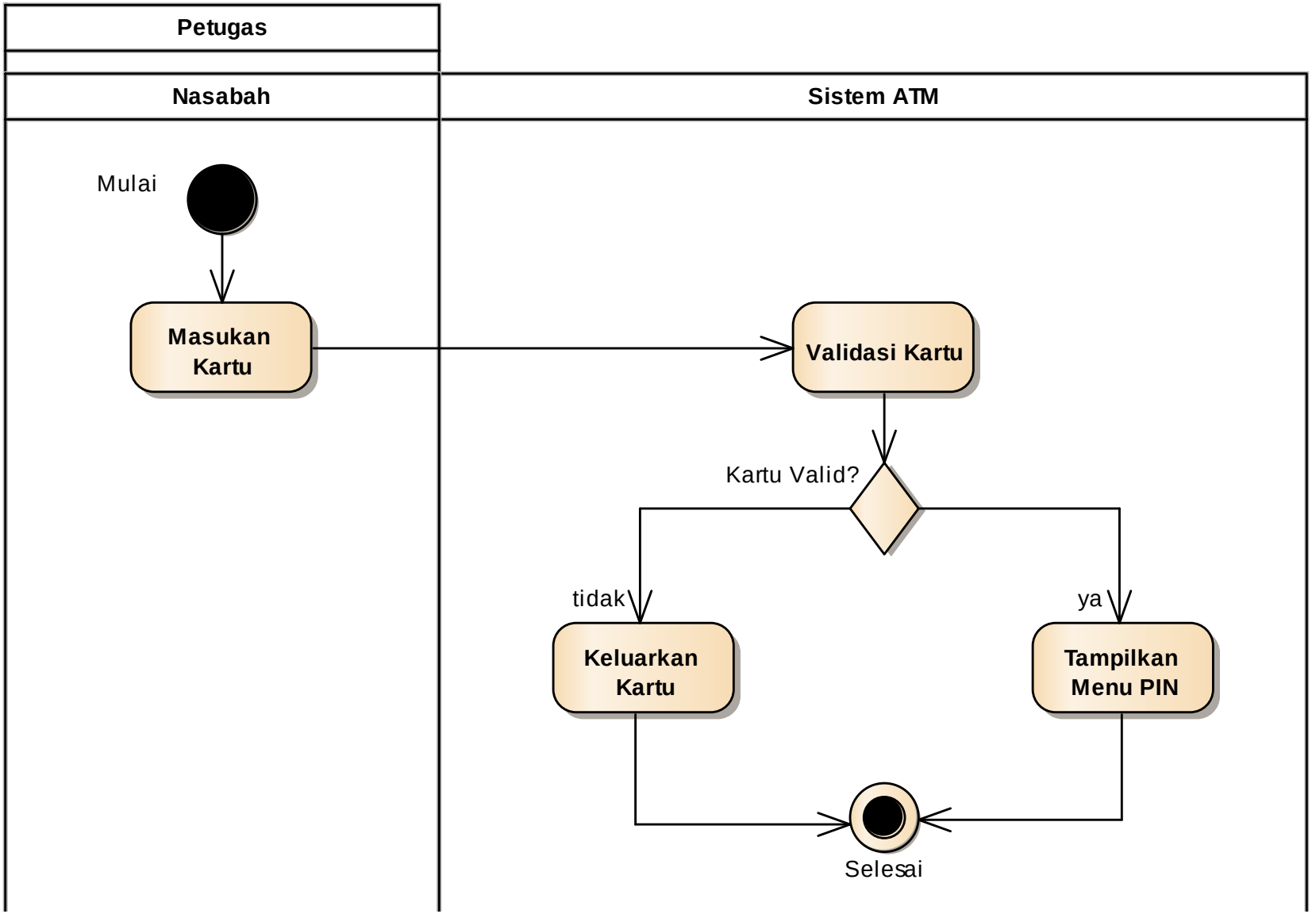
*(Wahono, 2009)*



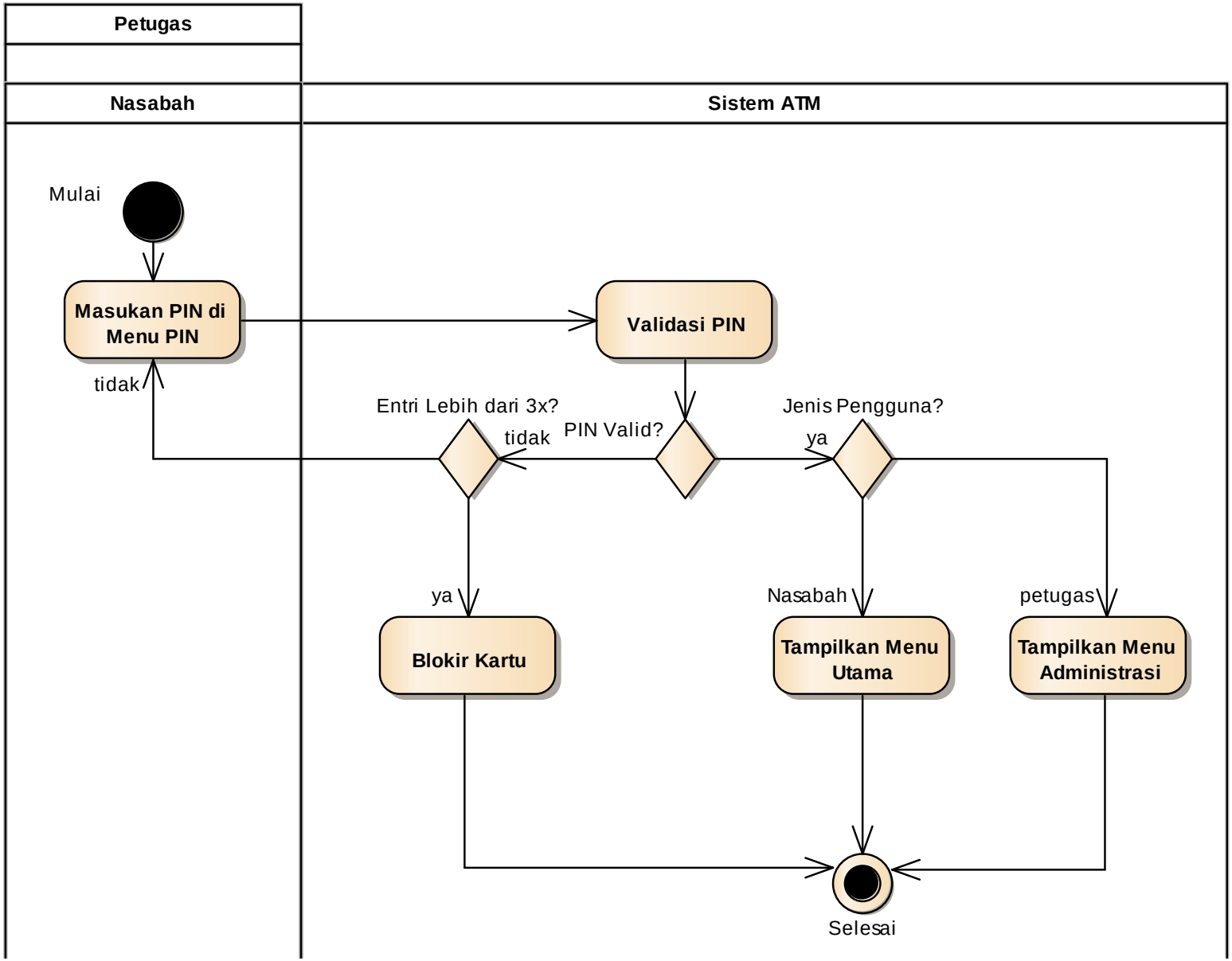
# Use Case Diagram



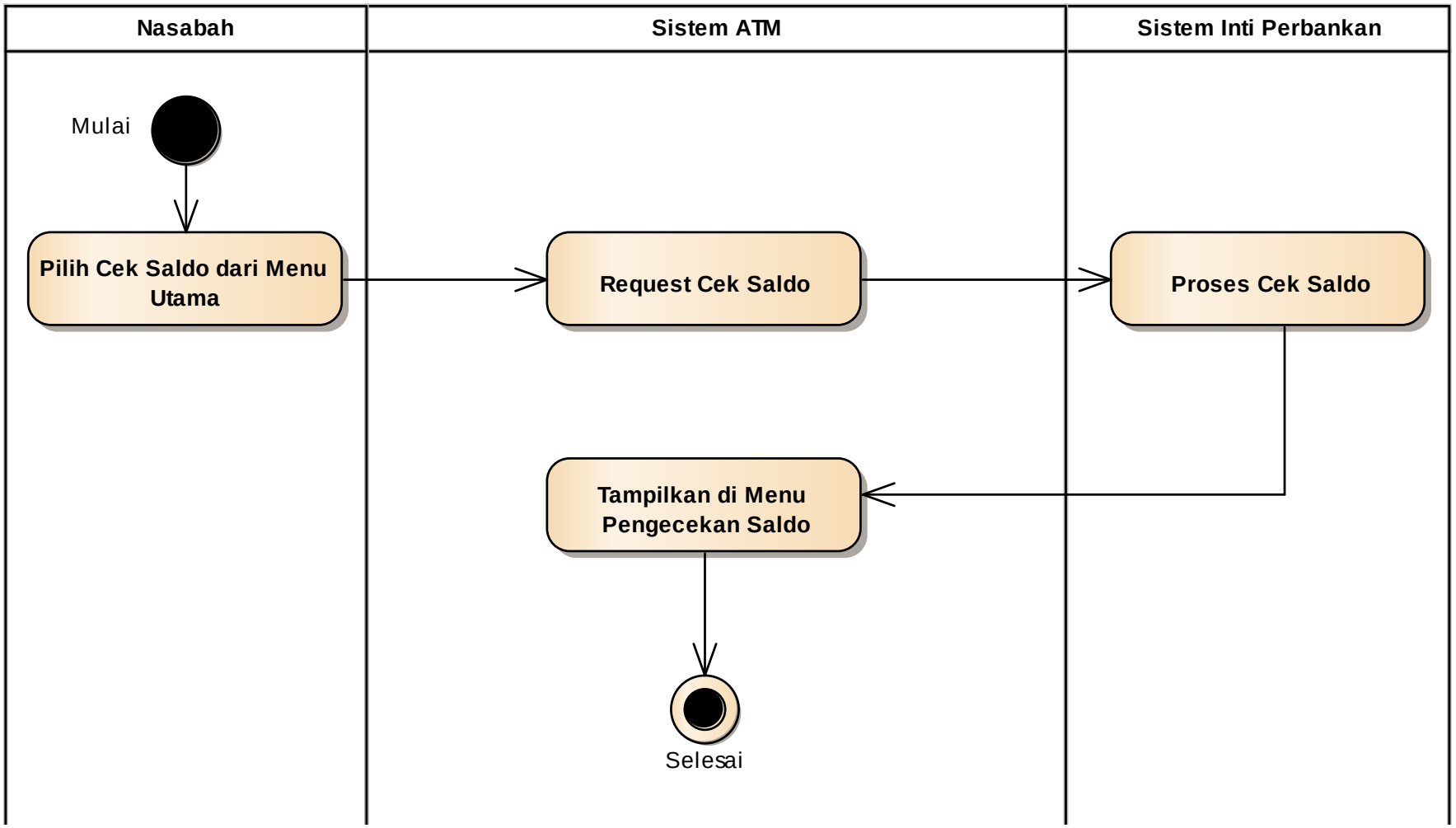
# Activity Diagram: Memasukkan Kartu



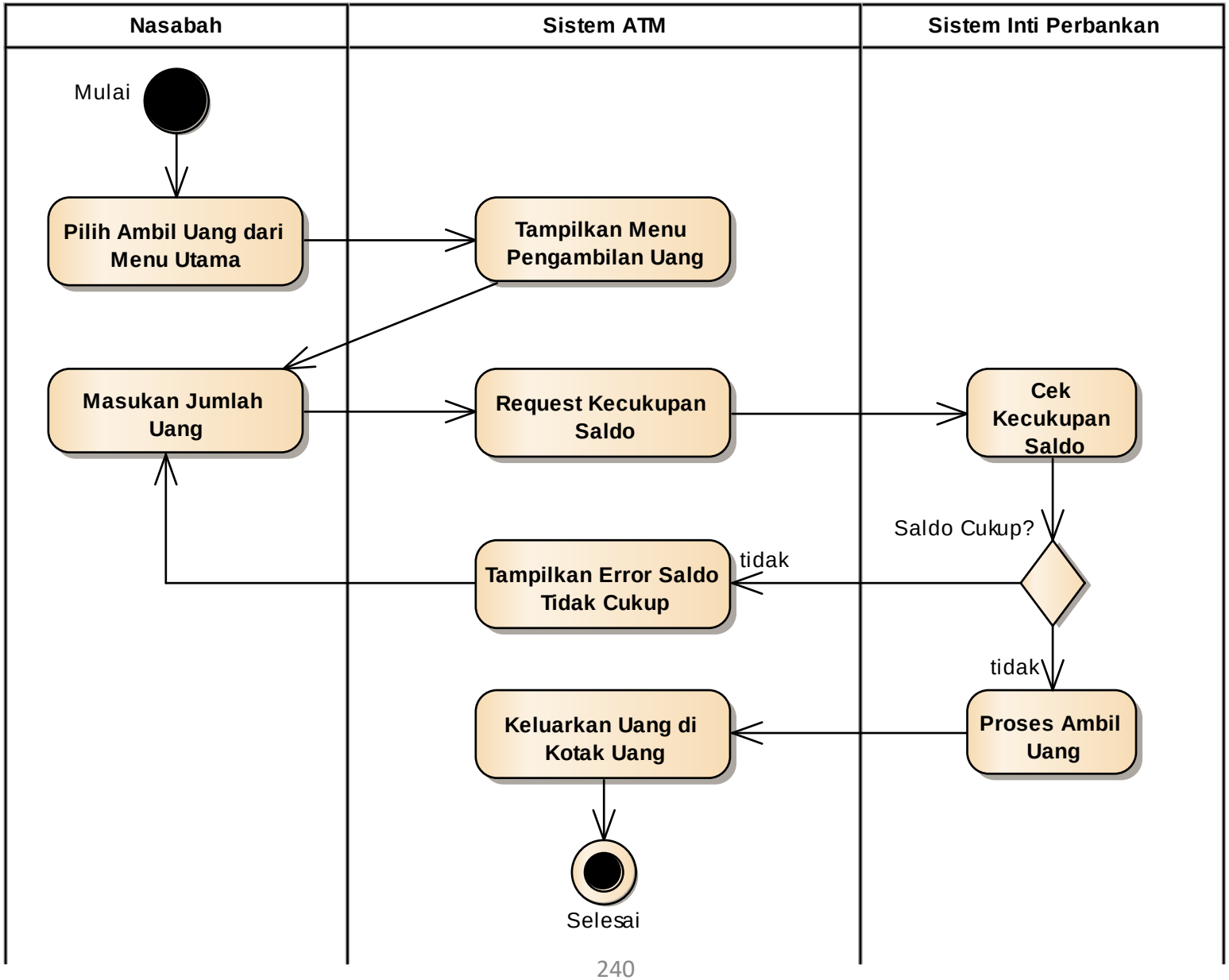
# Activity Diagram: Memasukkan PIN



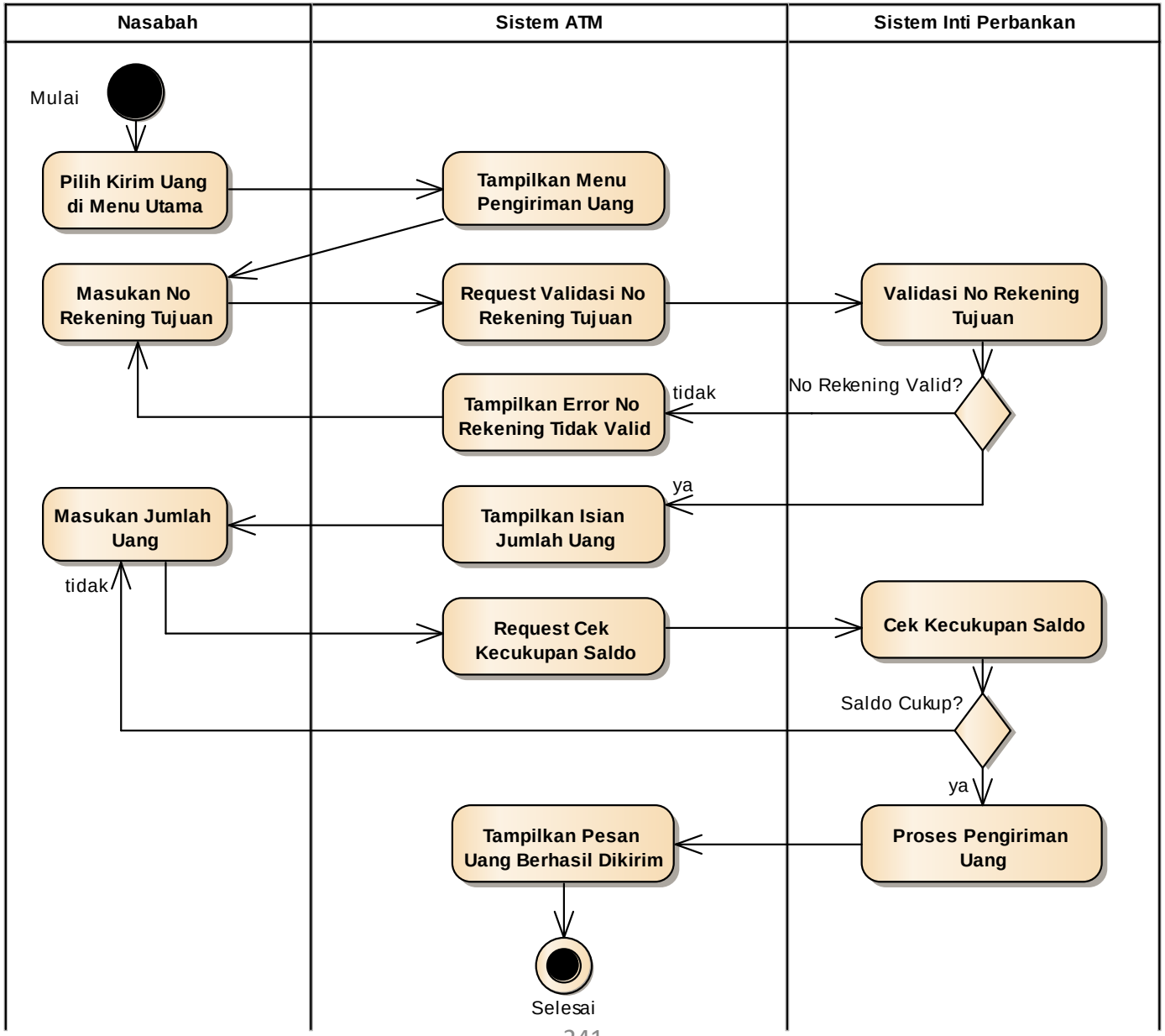
# Activity Diagram: Mengecek Saldo



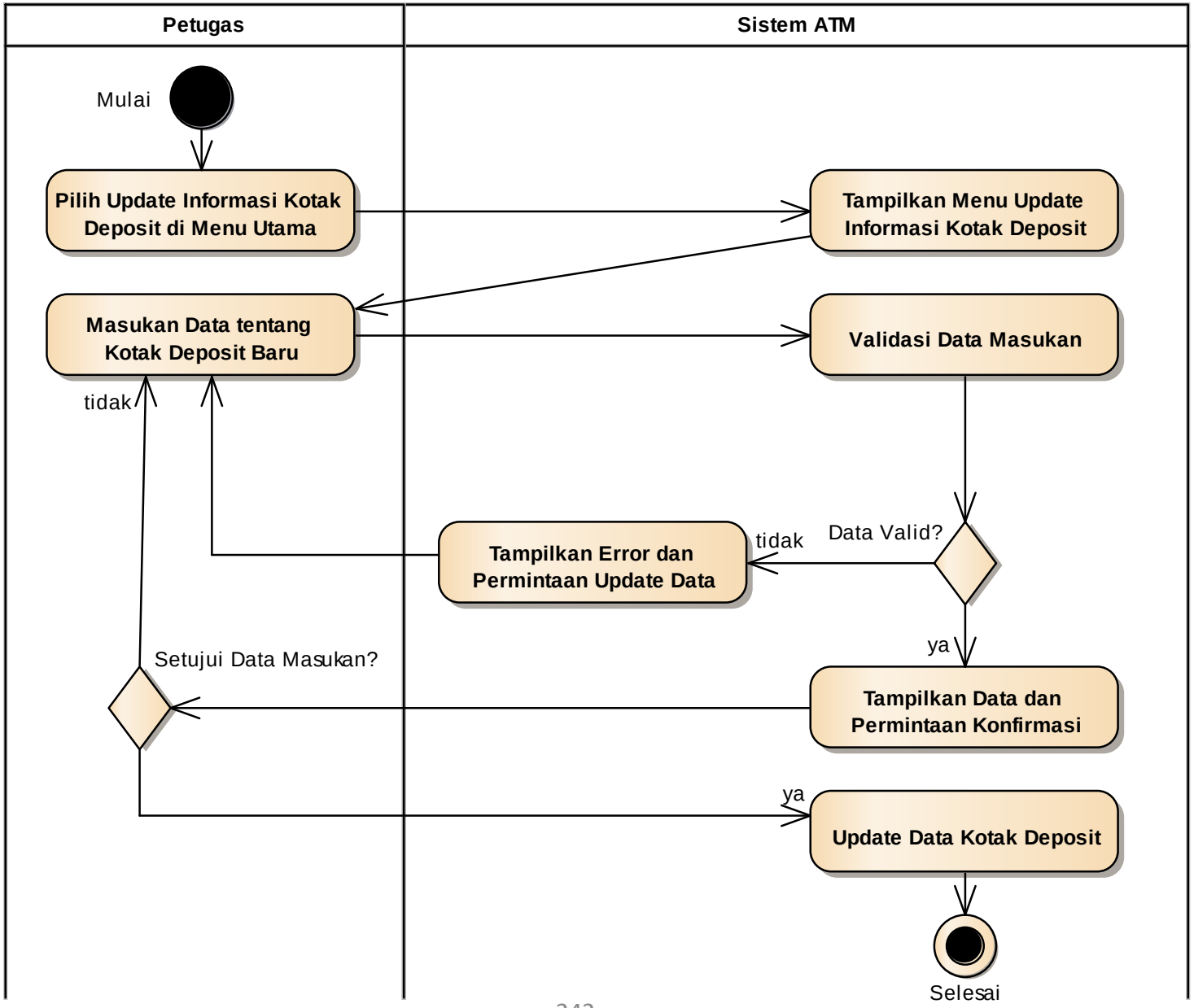
# Activity Diagram: Mengambil Uang



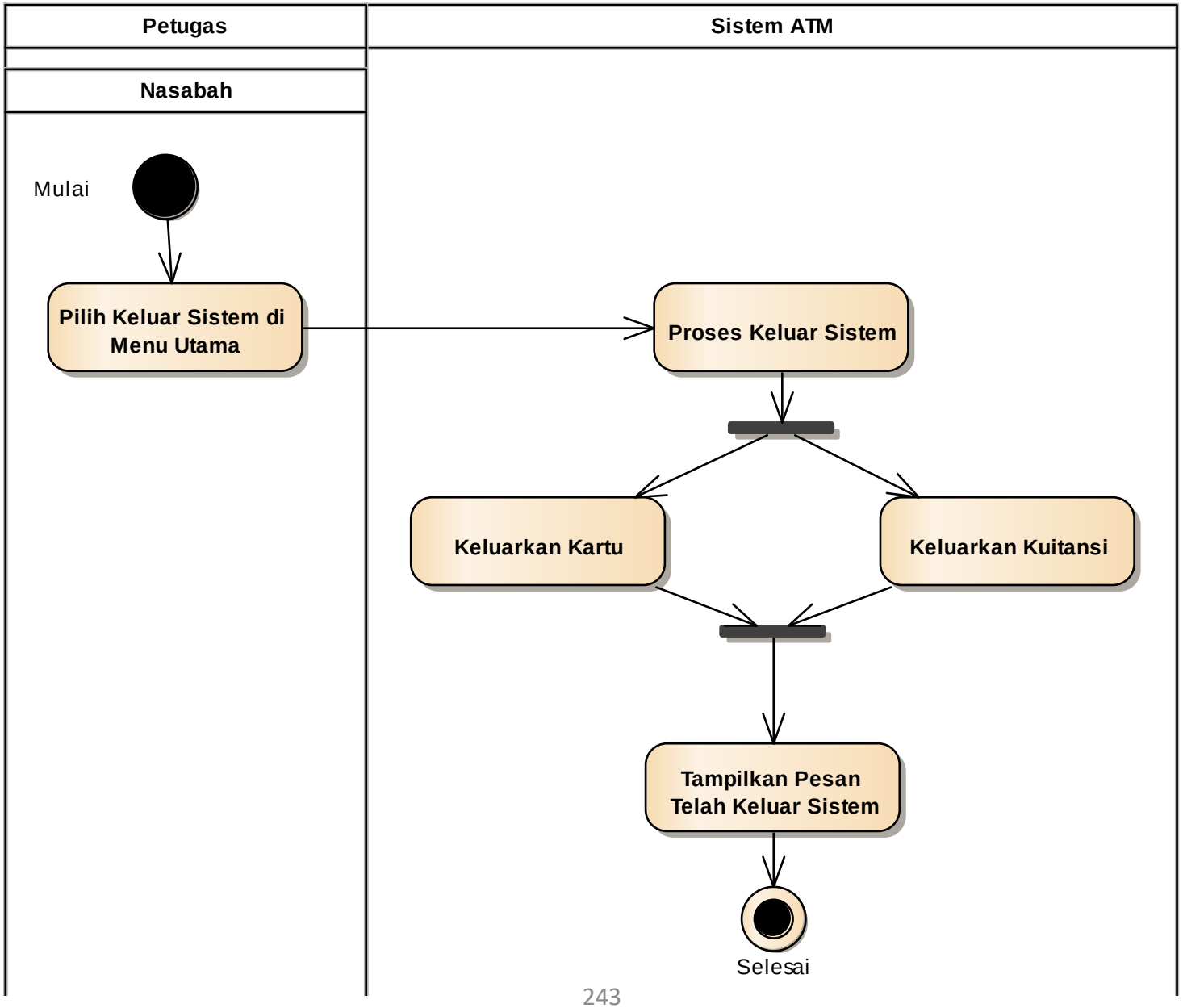
# Activity Diagram: Mengirim Uang



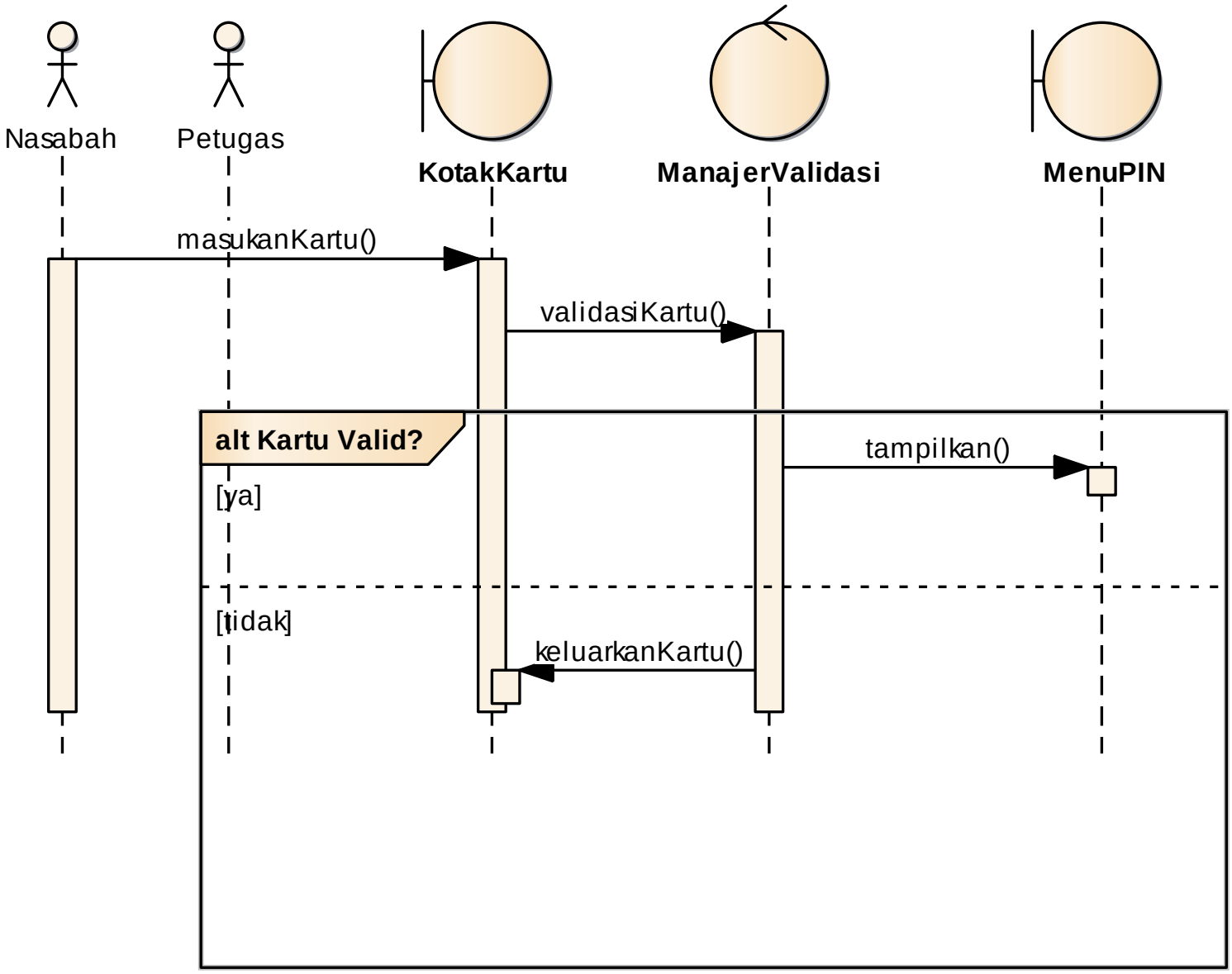
# Activity Diagram: Mengupdate Informasi Kotak Deposit



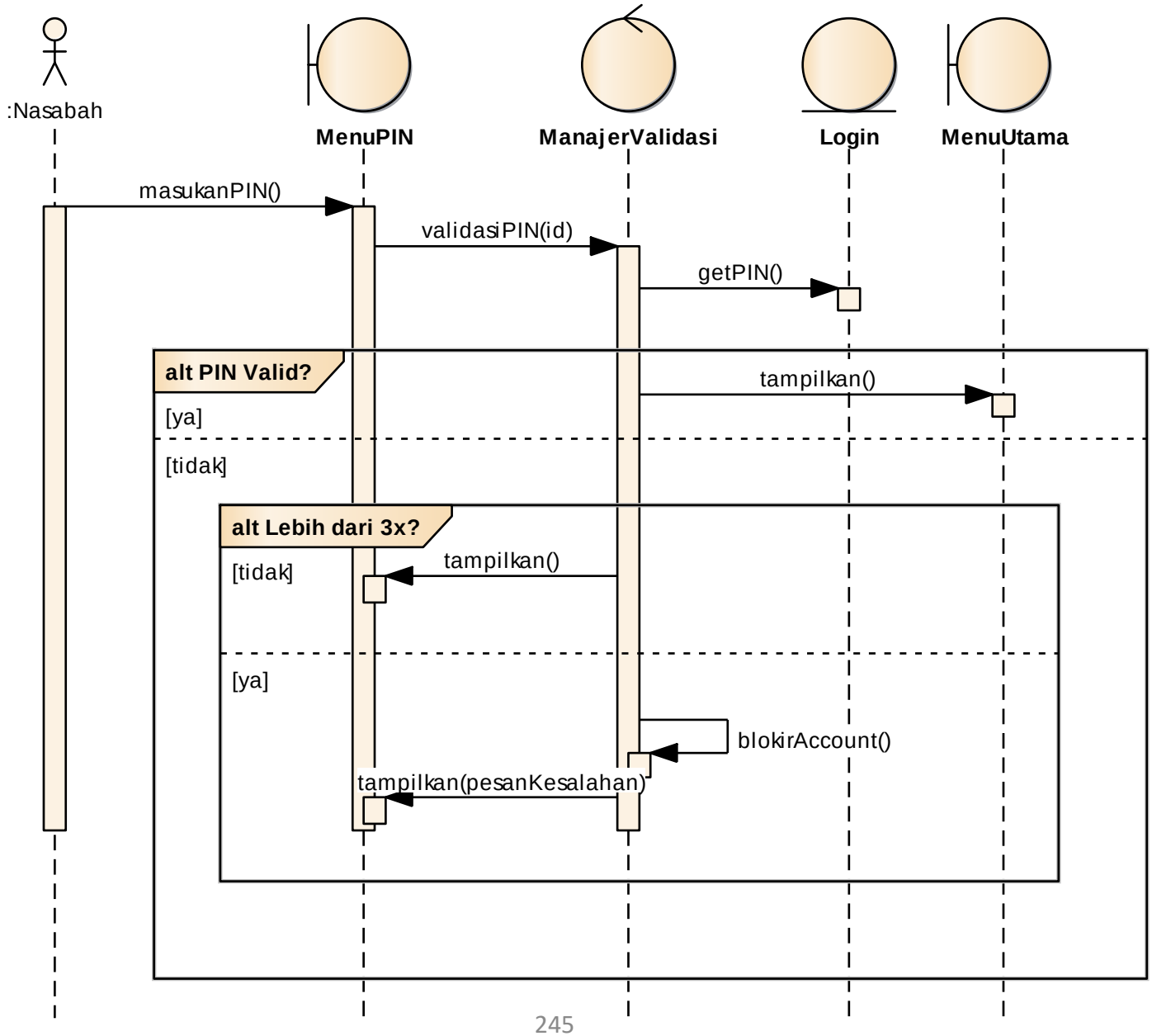
# Activity Diagram: Keluar Sistem



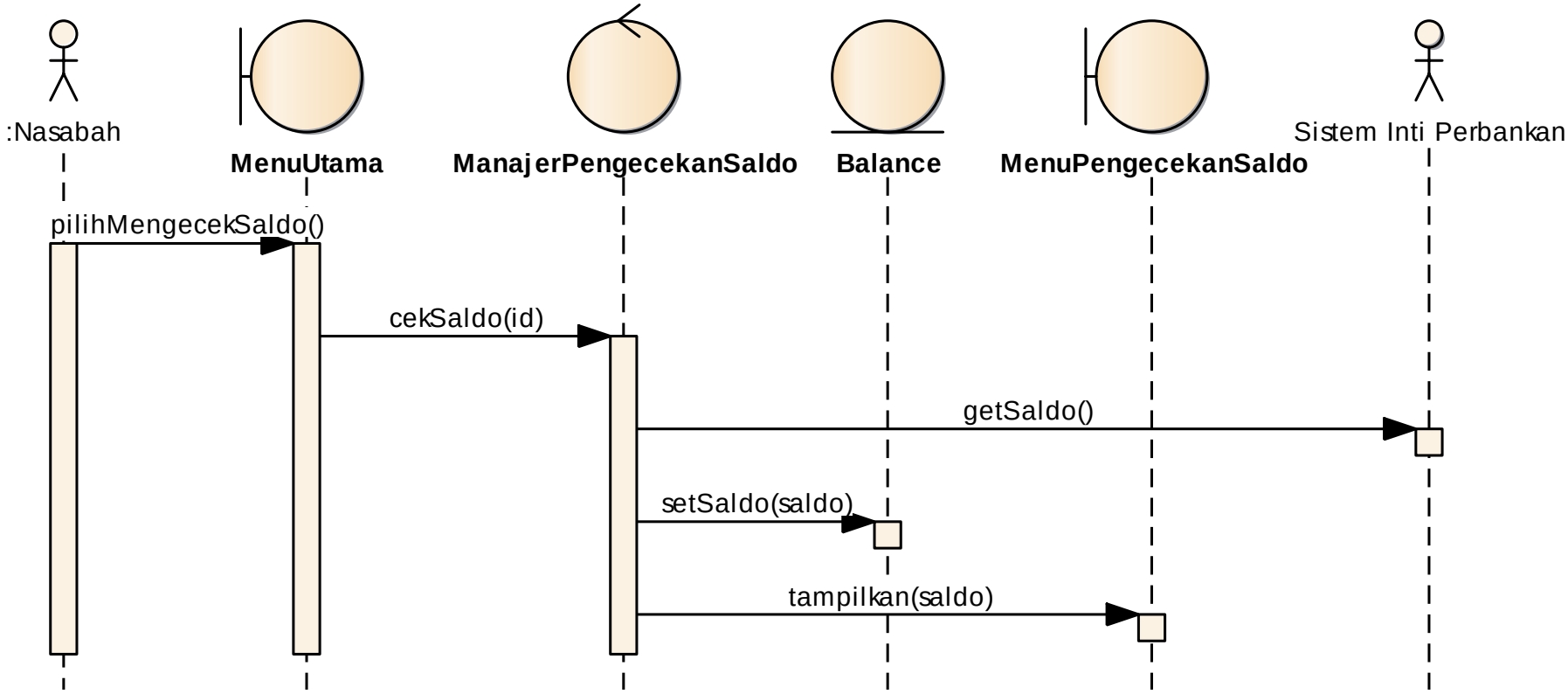
# Sequence Diagram: Memasukkan Kartu



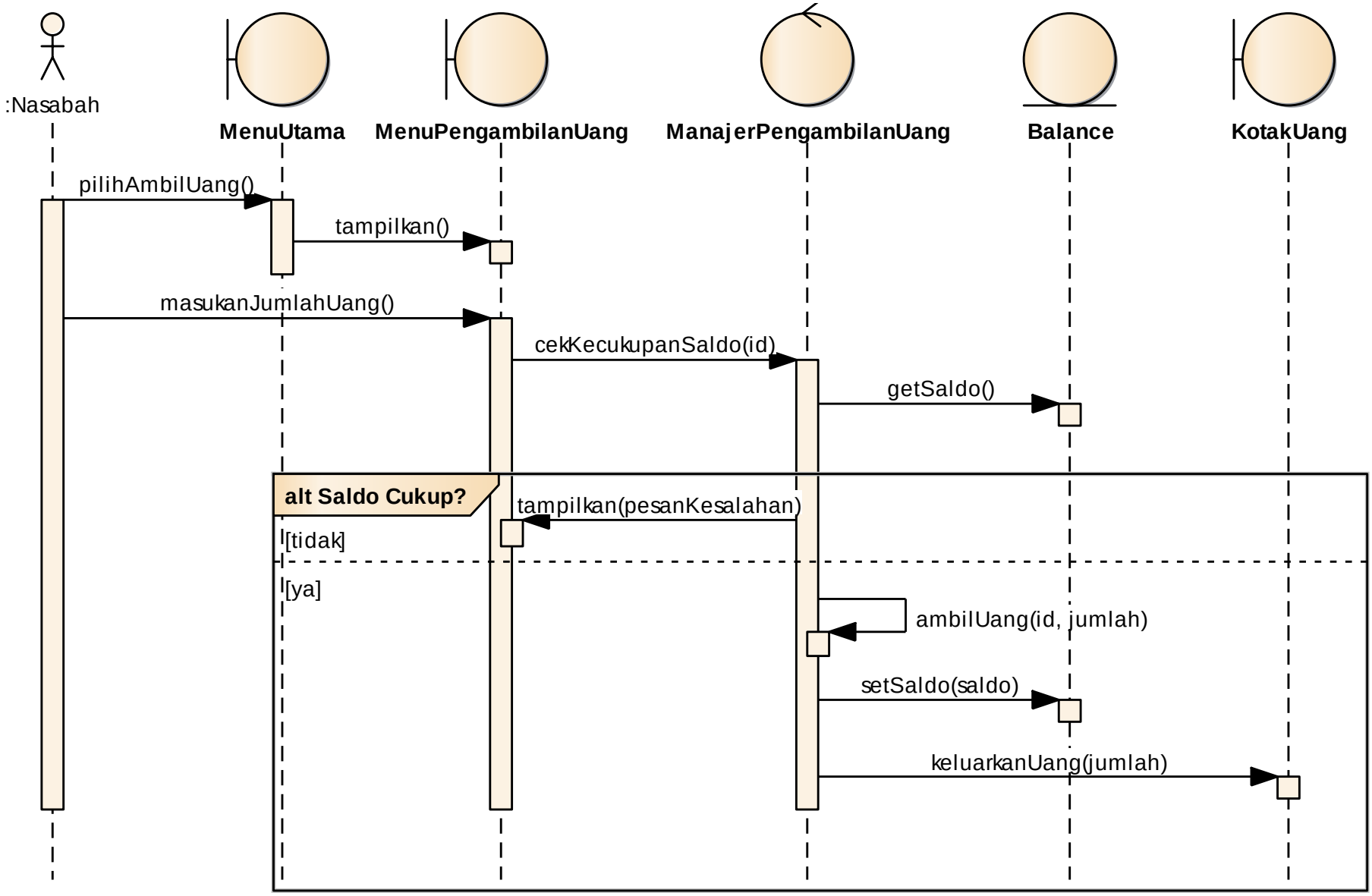
# Sequence Diagram: Memasukkan PIN



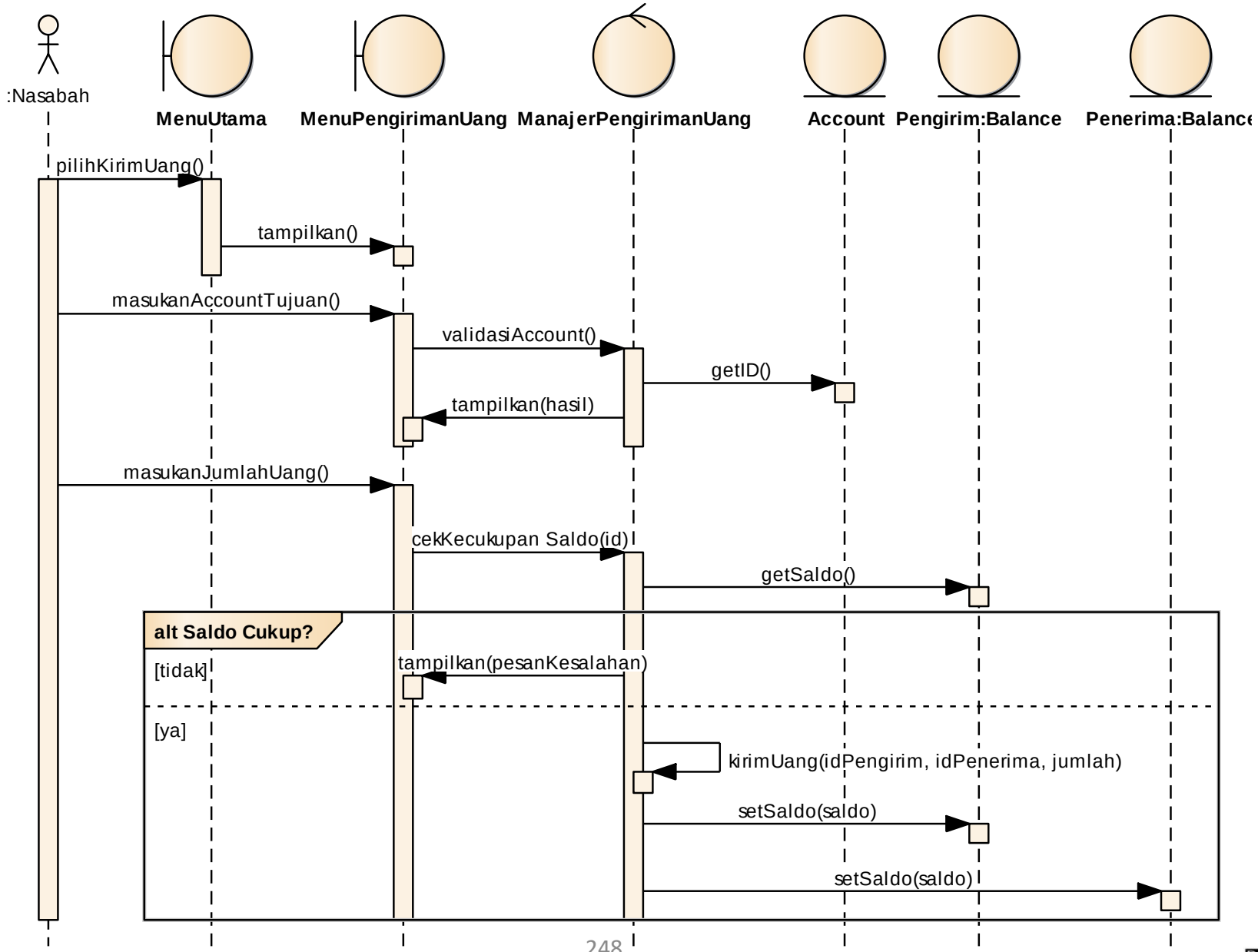
# Sequence Diagram: Mengecek Saldo



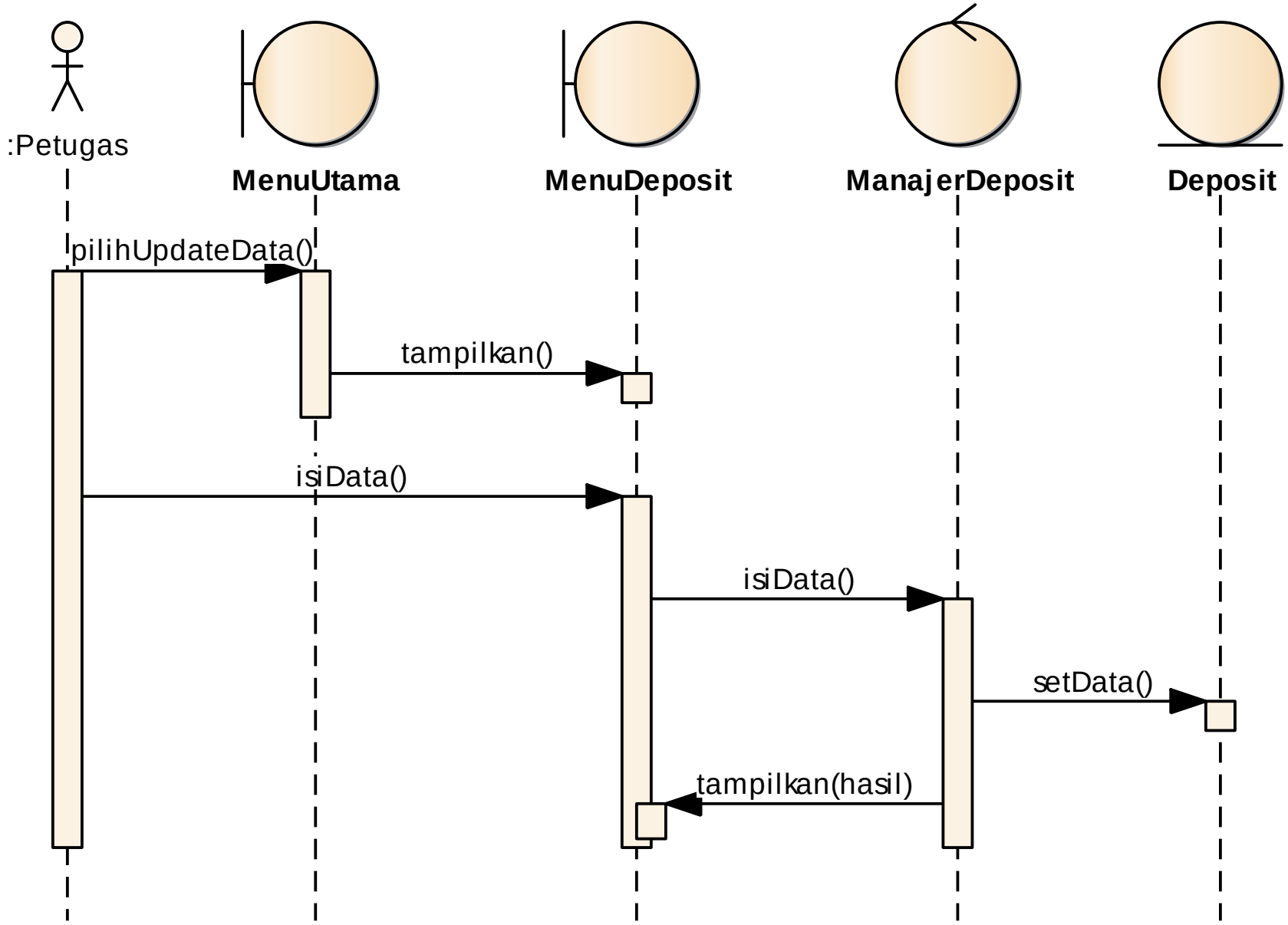
# Sequence Diagram: Mengambil Uang



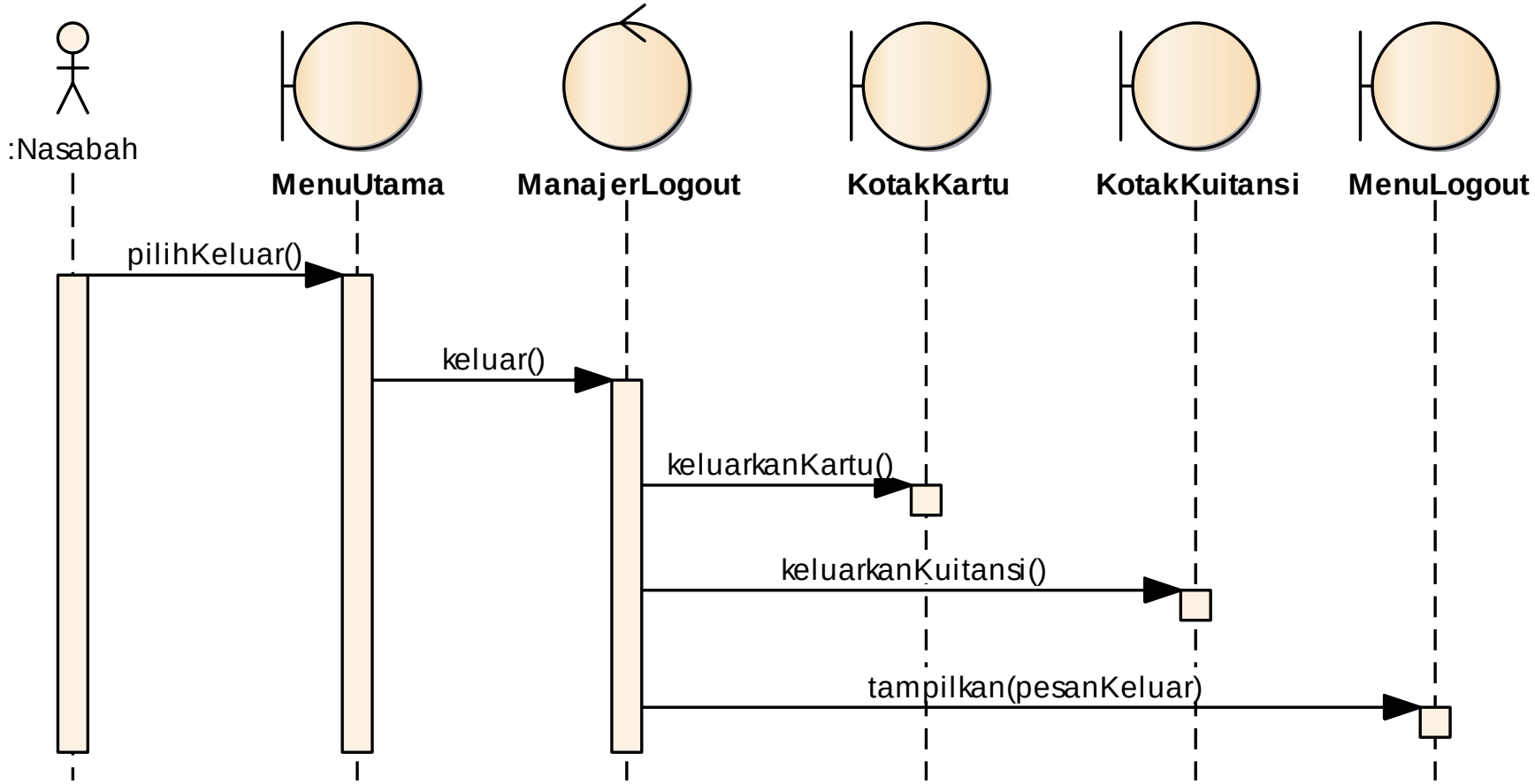
# Sequence Diagram: Mengirim Uang



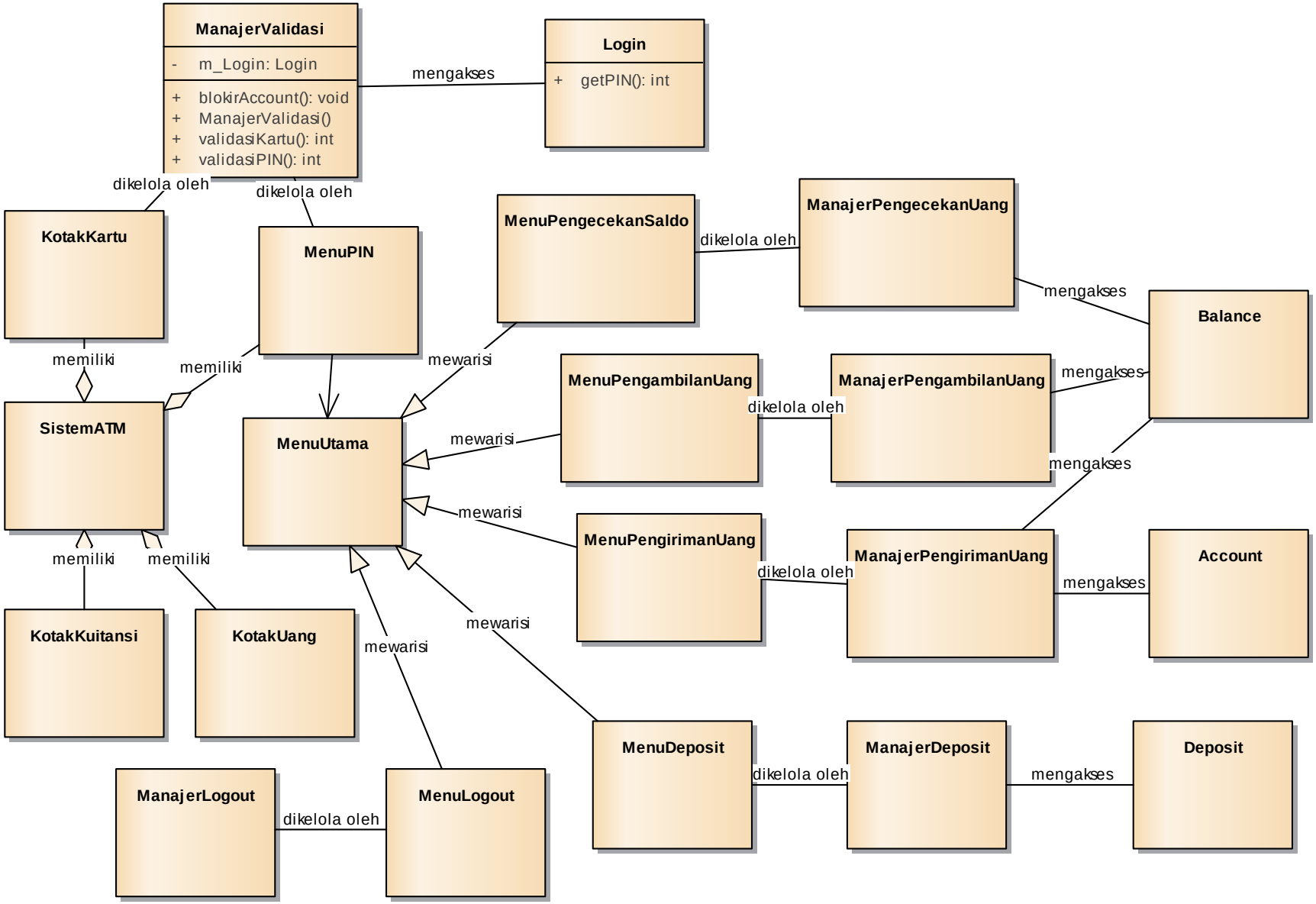
# Sequence Diagram: Mengupdate Informasi Kotak Deposit



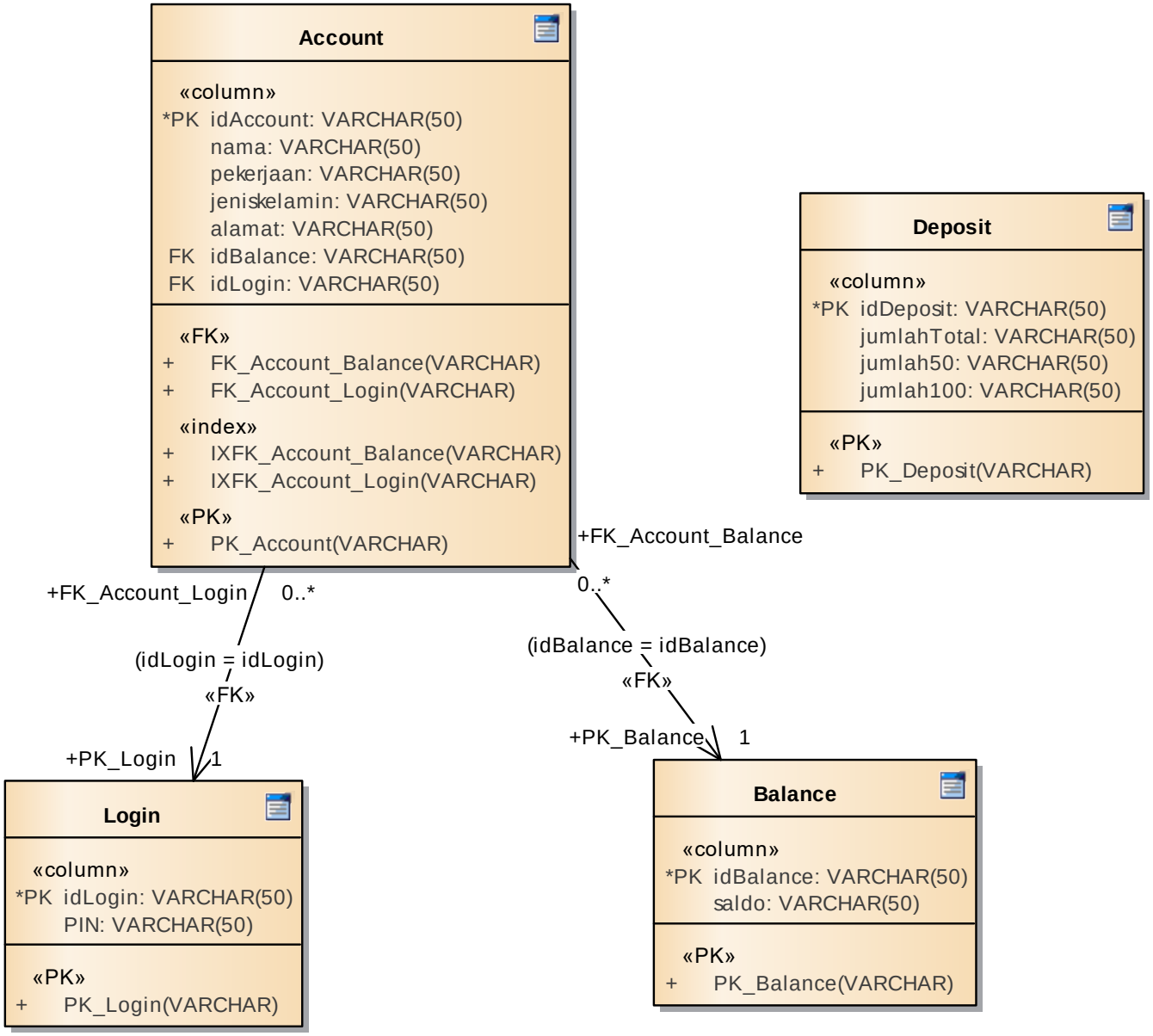
# Sequence Diagram: Keluar Sistem



# Class Diagram

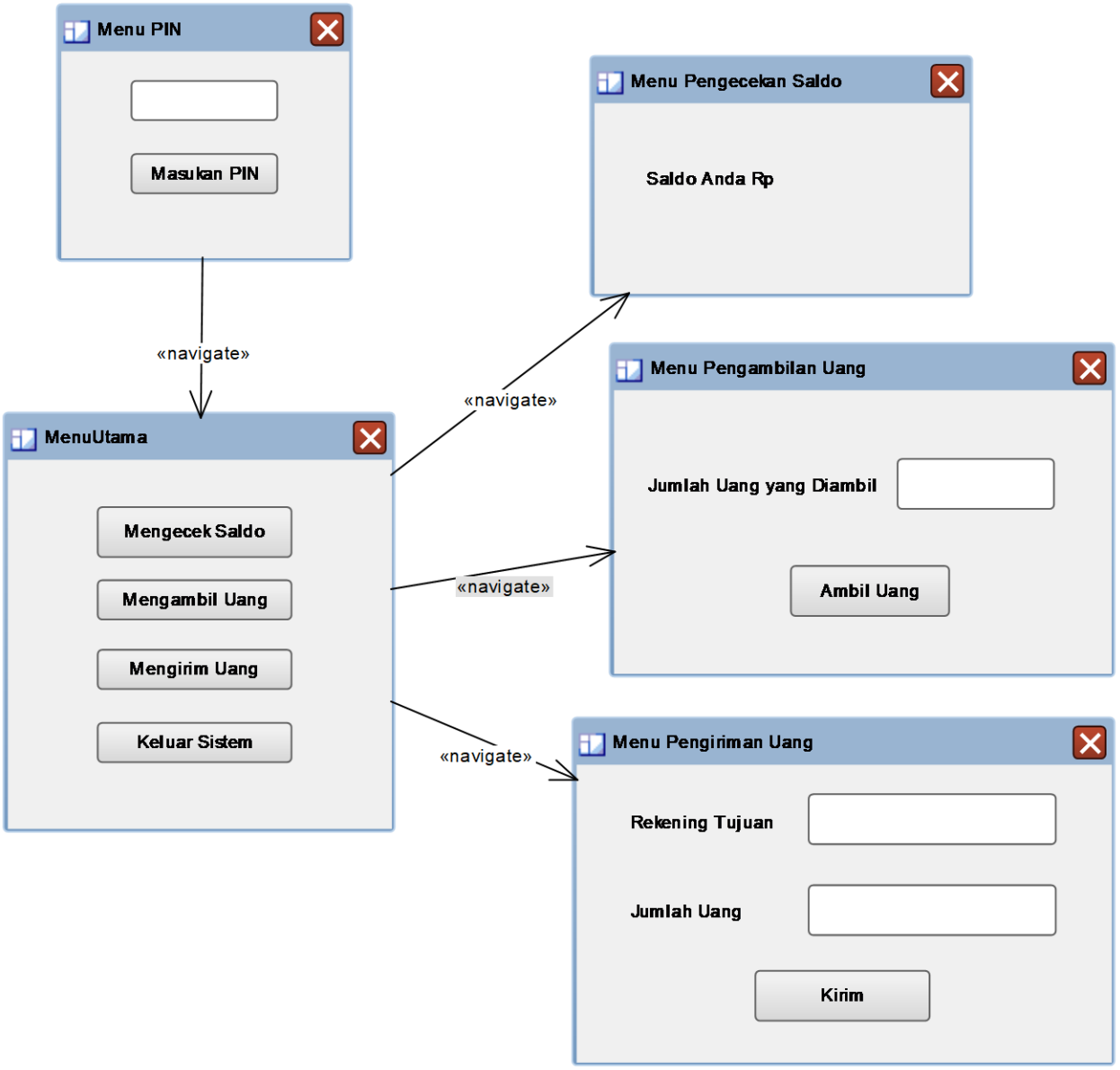


# Data Model

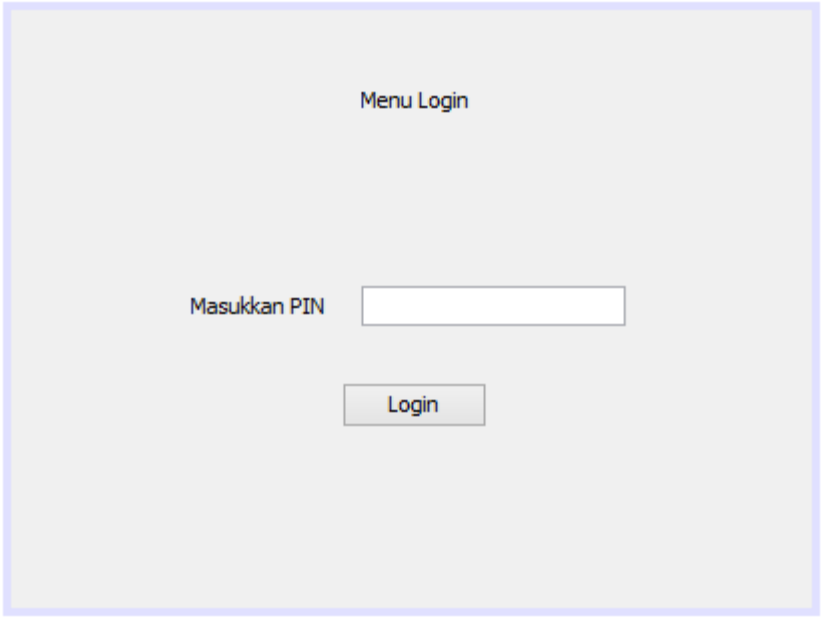


# User Interface Design

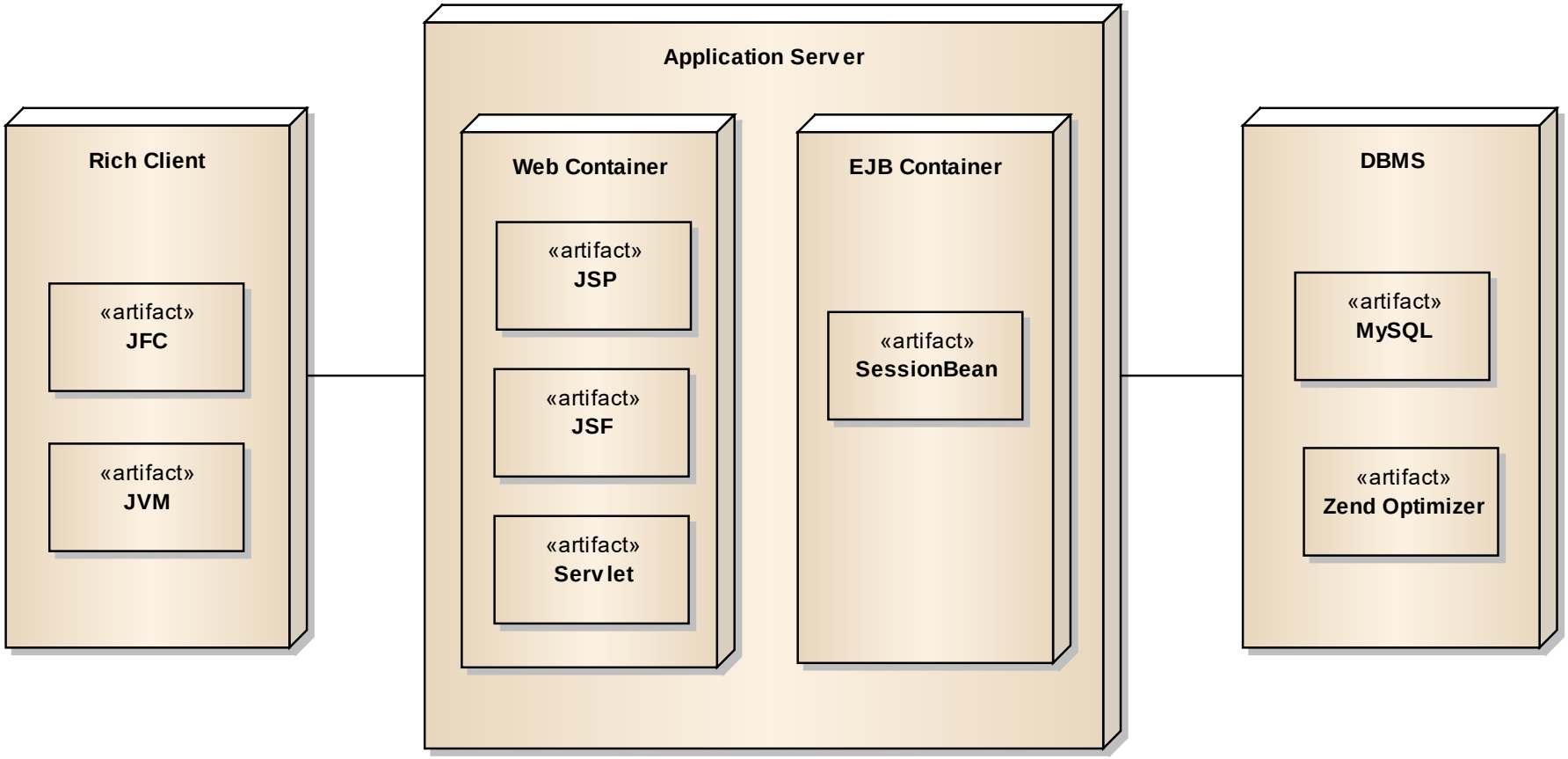
# User Interface Design (Sparx EA)



# User Interface Design (Netbeans)




# Deployment Diagram (3 Tier)



# Contoh Template System Specification (SRS, BRD, FSD)

1. System Planning
   - 1.1 Project Scope
   - 1.2 Project Schedule
   - 1.3 Project Team
2. System Design
   - 2.1 Functional Requirements
     - 2.1.1 Actor
     - 2.1.2 Use Case Diagram
     - 2.1.3 Activity Diagram (BPMN)
     - 2.1.4 Sequence Diagram
     - 2.1.5 Class Diagram
     - 2.1.6 Data Model
     - 2.1.7 User Interface Design
     - 2.1.8 Deployment Diagram
     - 2.1.9 Relational Matrices
       - 2.1.9.1 Actor – Activity Diagram
       - 2.1.9.2 Actor – Sequence Diagram
   - 2.2 Nonfunctional Requirements
     - 2.2.1 Operational
     - 2.2.2 Performance
     - 2.2.3 Security
     - 2.2.4 Hardware
     - 2.2.5 Development Platform
     - 2.2.6 Deadline
3. System Implementation
   - 3.1 Testing Strategy
   - 3.2 Installation Strategy
   - 3.3 Change Management Strategy

# Terima Kasih

Romi Satria Wahono
*romi@romisatriawahono.net*
*http://romisatriawahono.net*
*08118228331*