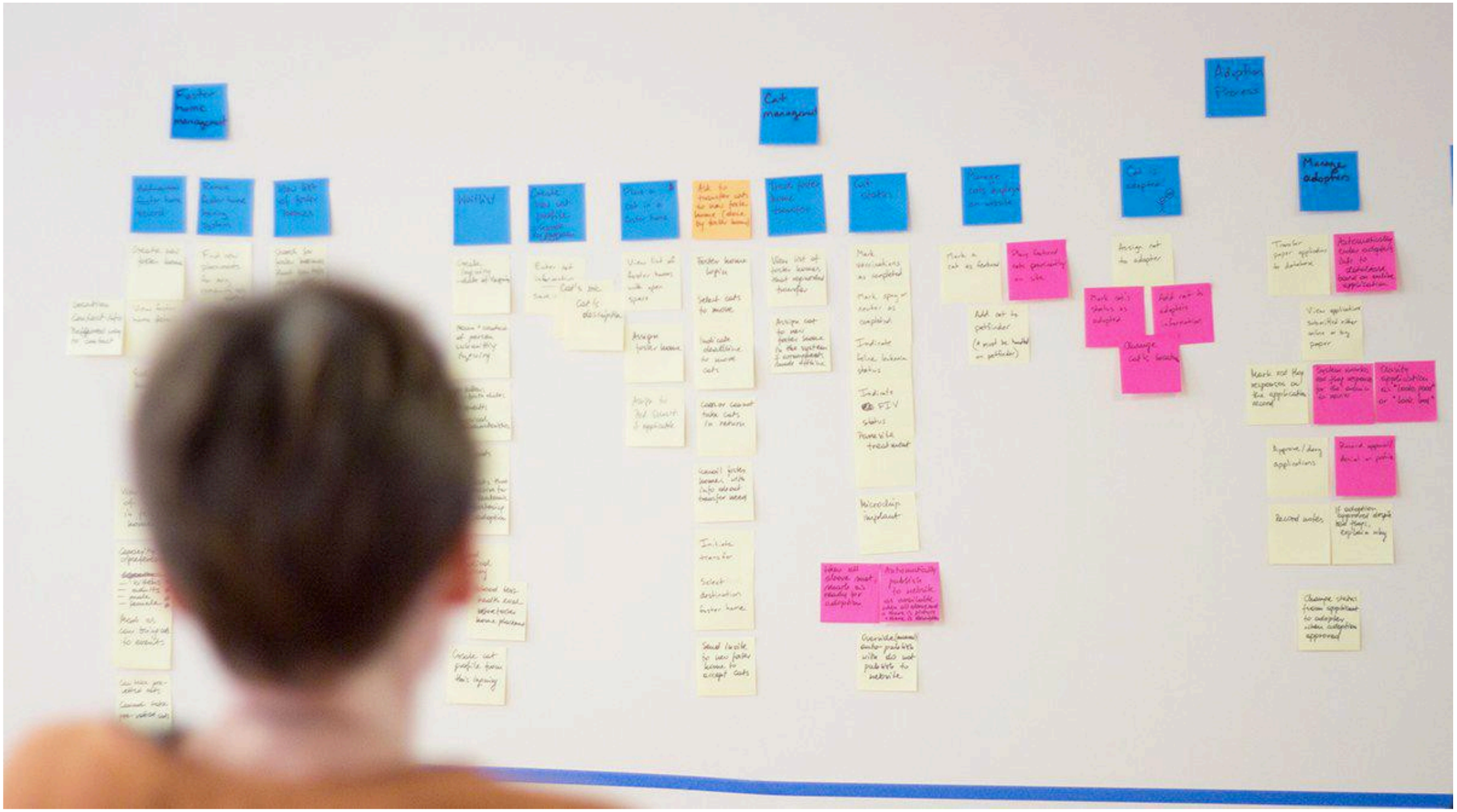


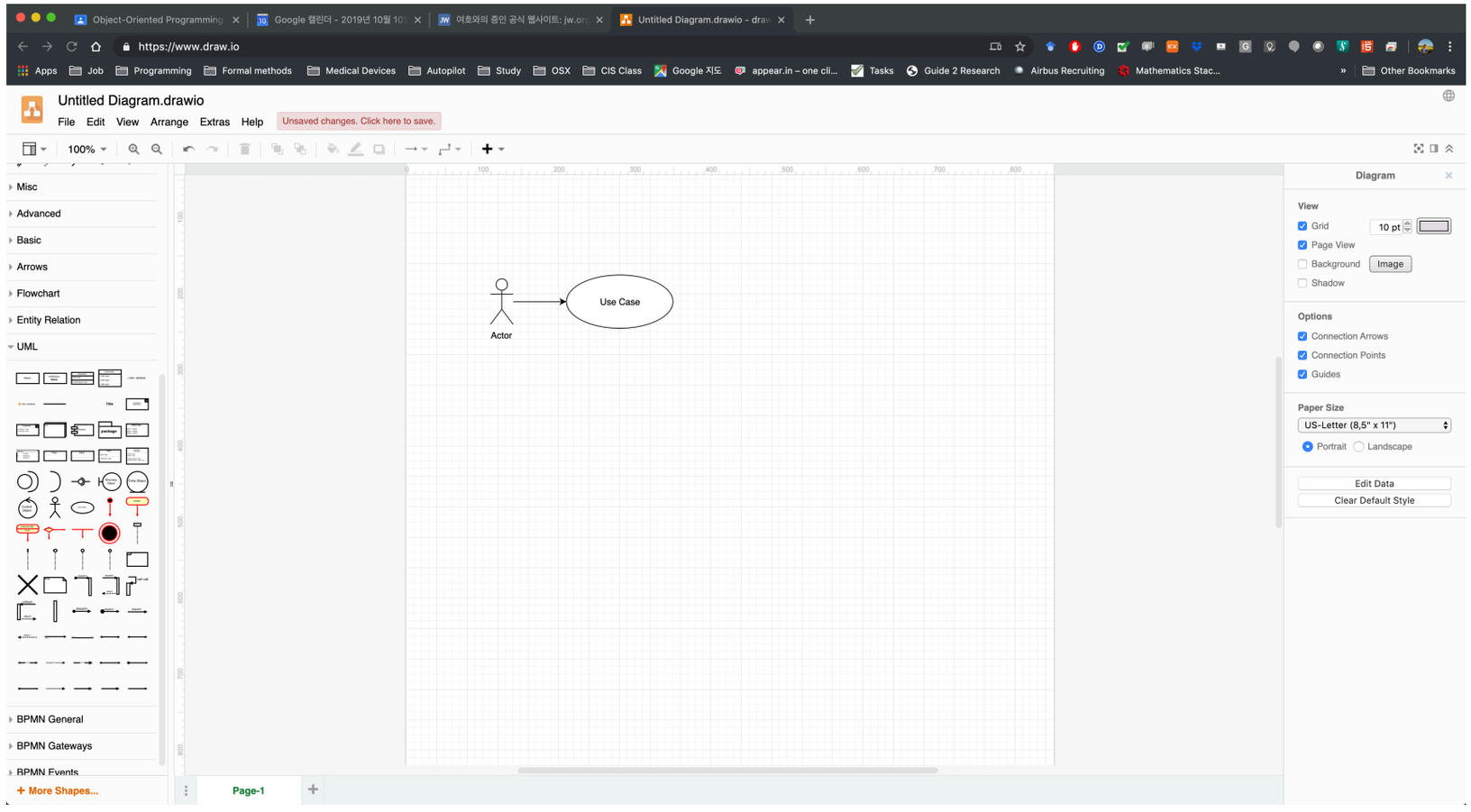
Use-Case Diagram

Jin Hyun Kim

Fall, 2019

UML Tools

- http://draw.io



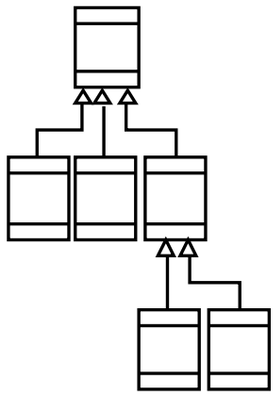
UML

- The Unified Modeling Language™ (UML) was developed jointly by Grady Booch, Ivar Jacobson, and Jim Rumbaugh with contributions from other leading methodologists, software vendors, and many users. The UML provides the application modeling language for:
  - Process modeling/ Requirement Analysis with Use-cases.
  - Static Design with Class and object modeling.
  - Dynamic Design with sequence, collaboration and activity diagrams.
  - Realtime Systems design models
  - Distribution and deployment modeling.

SW Lifecycle and UML

- Requirement Analysis
  - The functionality users require from the system (사용자가 요구하는 기능들)
  - **Use-case model**
- OO Analysis
  - Discovering classes and relationships
  - **Class diagram**
- OO Design
  - Result of Analysis expanded into technical solution (솔루션을 얻기 위해 분석 결과를 확장)
  - **Sequence diagram**, state diagram, etc.
  - Results in detailed specs for the coding phase (코딩을 위한 상세 설계로 결과가 도출)
- Implementation (Programming/coding)
  - Models are converted into code
- Testing
- Unit tests, integration tests, system tests and acceptance tests.

Modeling Notation

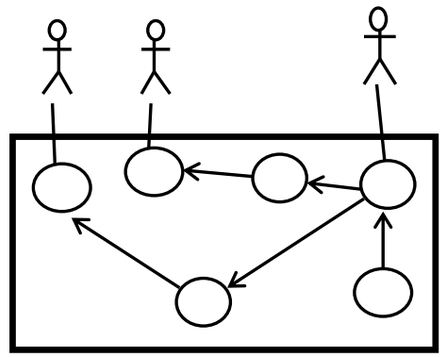


UML Class Diagrams

information structure

relationships between data items

modular structure for the system

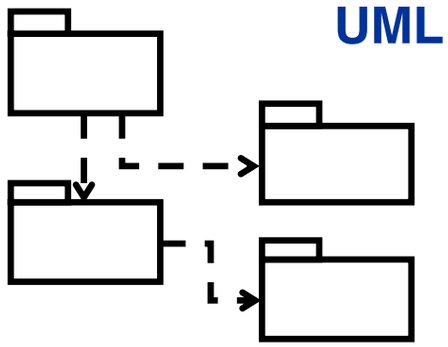


Use Cases

user's view

Lists functions

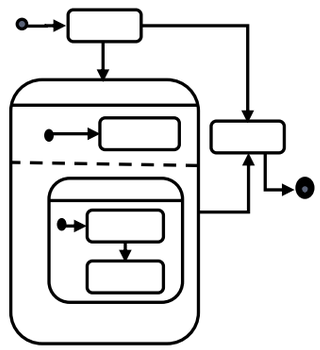
visual overview of the main requirements



UML Package Diagrams

Overall architecture

Dependencies between components

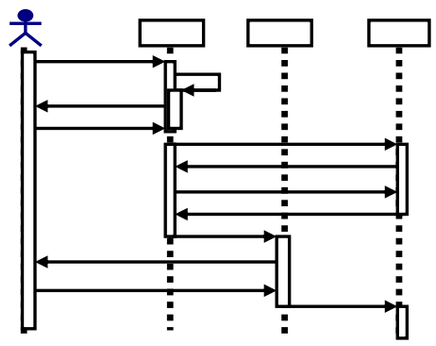


(UML) Statecharts

responses to events

dynamic behavior

event ordering, reachability, deadlock, etc

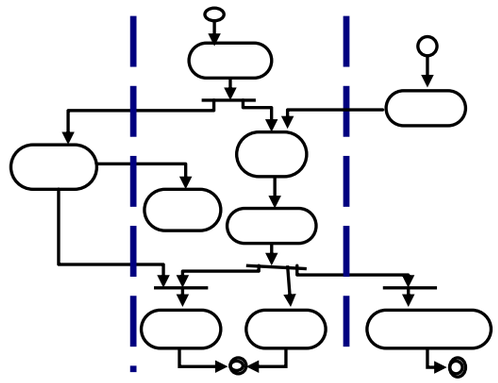


UML Sequence Diagrams

individual scenario

interactions between users and system

Sequence of messages



Activity diagrams

business processes;

concurrency and synchronization;

dependencies between tasks;

Use Case Modeling

- In use-case modeling, the system is looked upon as a black box whose boundaries are defined by its functionality to external stimuli.

- A formal way of representing how a business system interacts with its environment

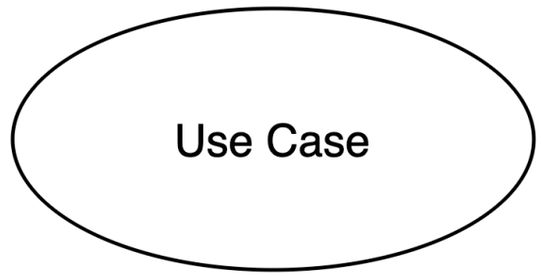
- Illustrates the activities that are performed by the users of the system

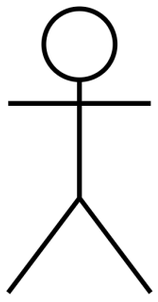
- A scenario-based technique in the UML

- A sequence of actions a system performs that yields a valuable result for a particular actor.

Components of UC

- System
- Actors
- Use-cases
- Stimulus



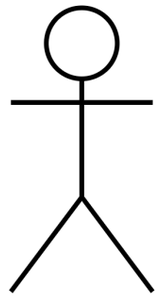


System

- As a part of the use-case modeling, the boundaries of the system are developed

- Define the scope of the system that you are going to design

Actors



Actor

- A user or outside system that interacts with the system being designed in order to obtain some value from that interaction

- Actor communicates with the system by sending and receiving messages.

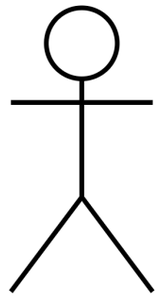
- An actor provides the stimulus to activate an Use-case.

- Message sent by an actor may result in more messages to actors and to Use-cases.

- Actors can be ranked: primary and secondary; passive and active.

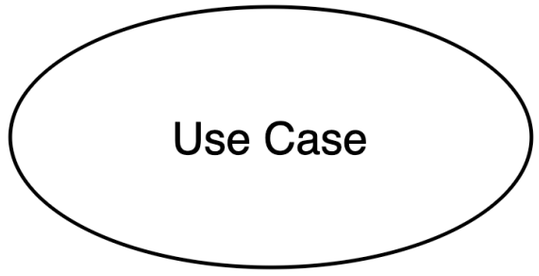
- Actor is a role not an individual instance.

Finding Actors



- The actors of a system can be identified by answering a number of questions:
  - Who will use the functionality of the system?
  - Who will maintain the system?
  - What devices does the system need to handle?
  - What other system does this system need to interact?
  - Who or what has interest in the results of this system?

Use Cases



- A Use-case in UML is defined as a set of sequences of actions a system performs that yield an observable result of value to a particular actor.
- Actions can involve communicating with number of actors as well as performing calculations and work inside the system
- A Use-case
  - is always initiated by an actor.
  - provides a value to an actor.
  - must always be connected to at least one actor.
  - must be a complete description.

Finding Use Cases

- For each actor ask these questions:
    - Which functions does the actor require from the system?
    - What does the actor need to do?
    - Could the actor’s work be simplified or made efficient by new functions in the system?
    - What events are needed in the system?
    - What are the problems with the existing systems?
    - What are the inputs and outputs of the system?

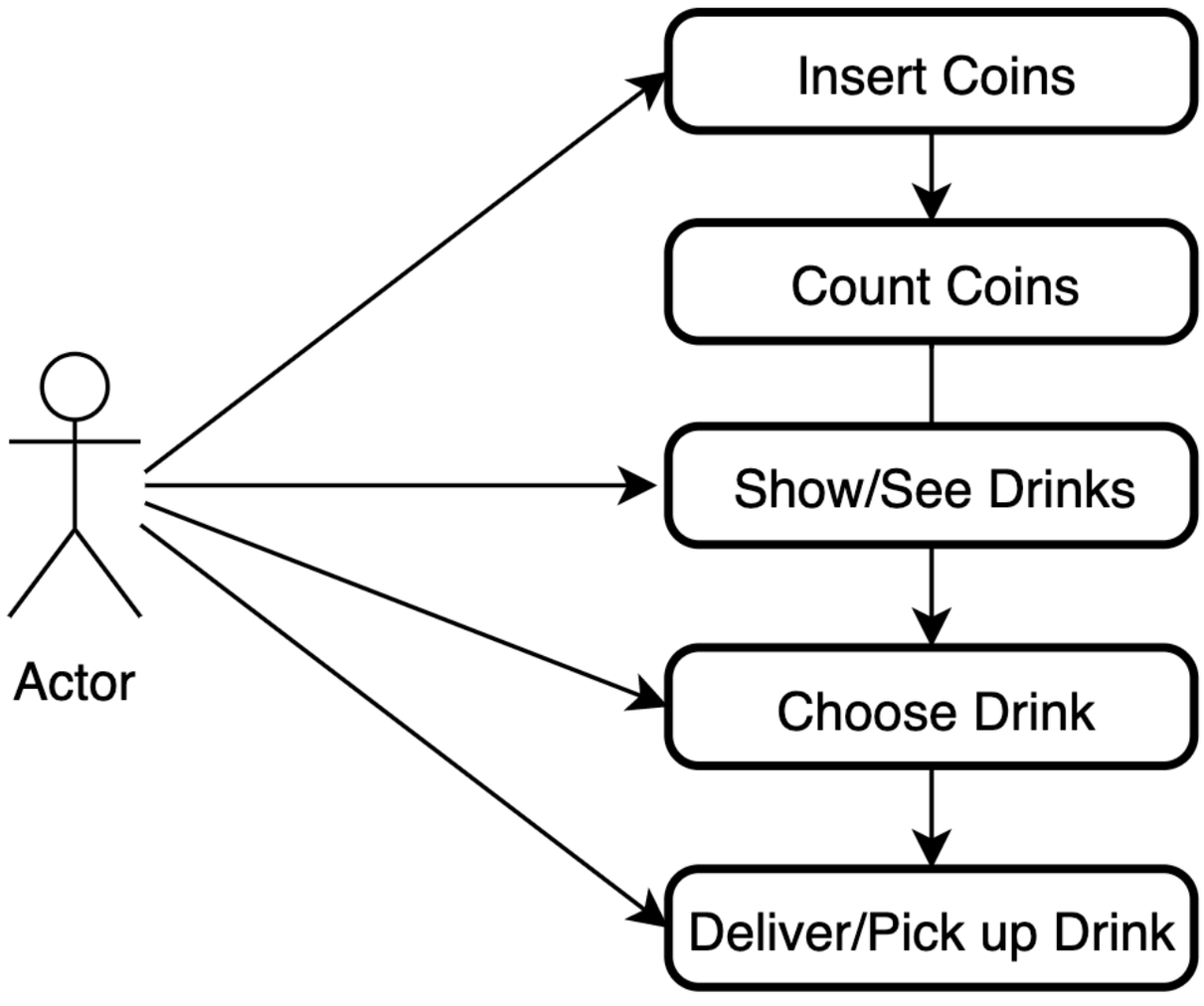
Describing Use-cases

- Use-case Name:
- Use-case Number: system#.diagram#.Use-case#
- Authors:
- Event(Stimulus):
- Actors:
- Overview: brief statement
- Related Use-cases:
- Typical Process description: Algorithm
- Exceptions and how to handle exceptions:

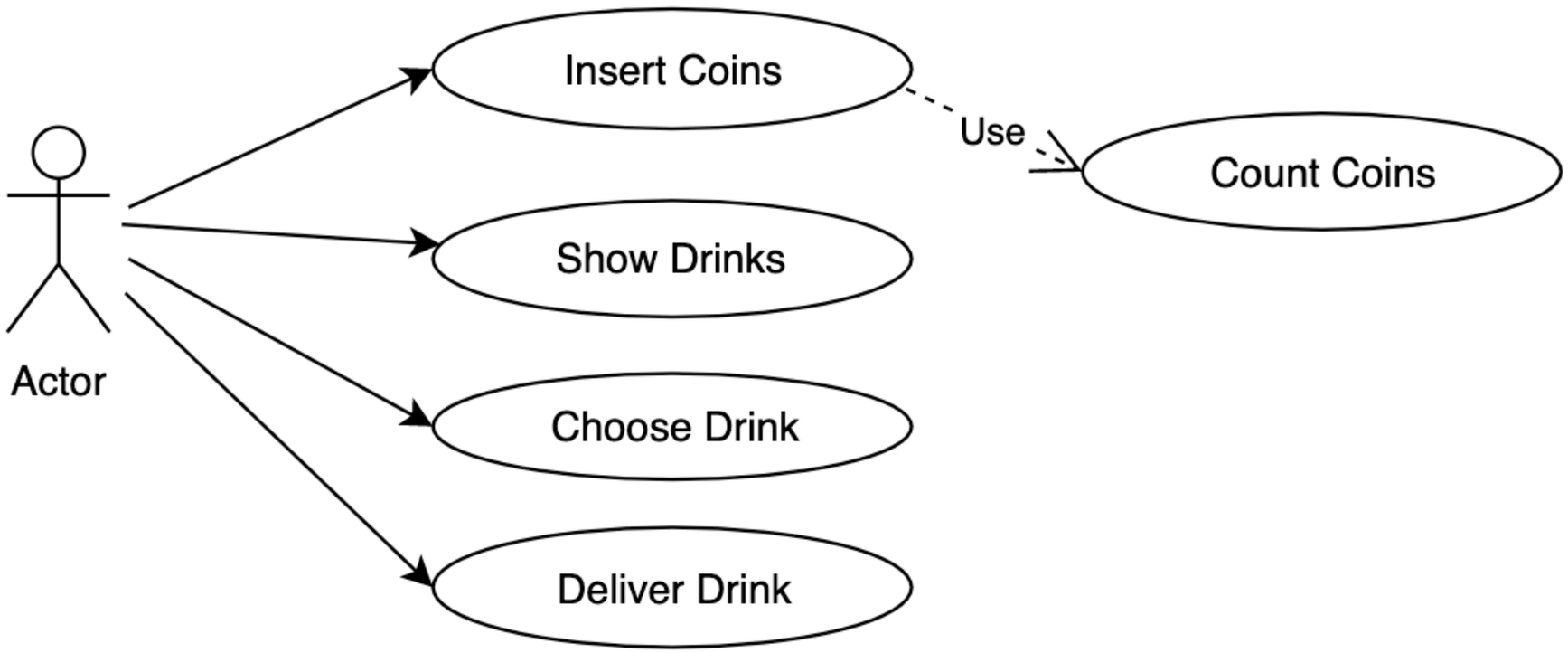
Example

- Number: A.132.4
- Name: Buy book online
- Author: B.Ramamurthy
- Event: Customer request one or more books
- System: Amazon.com
- Overview: Captures the process of purchasing one or more books and the transactions associated with it.
- Related Use-case: A. 132.5, A.132.8
- Typical Process Description with exceptions handled.

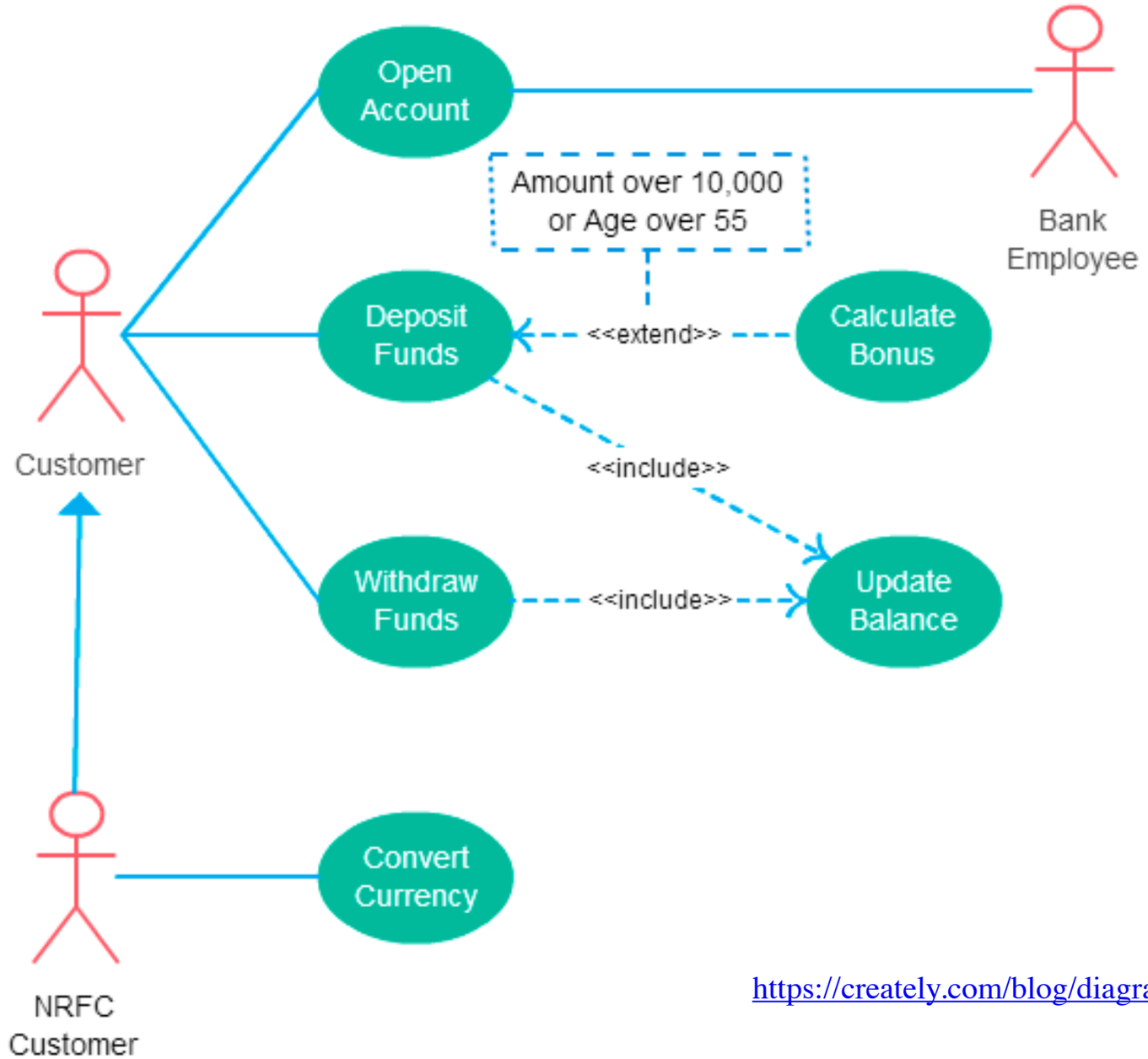
Interaction



Use Case Diagram

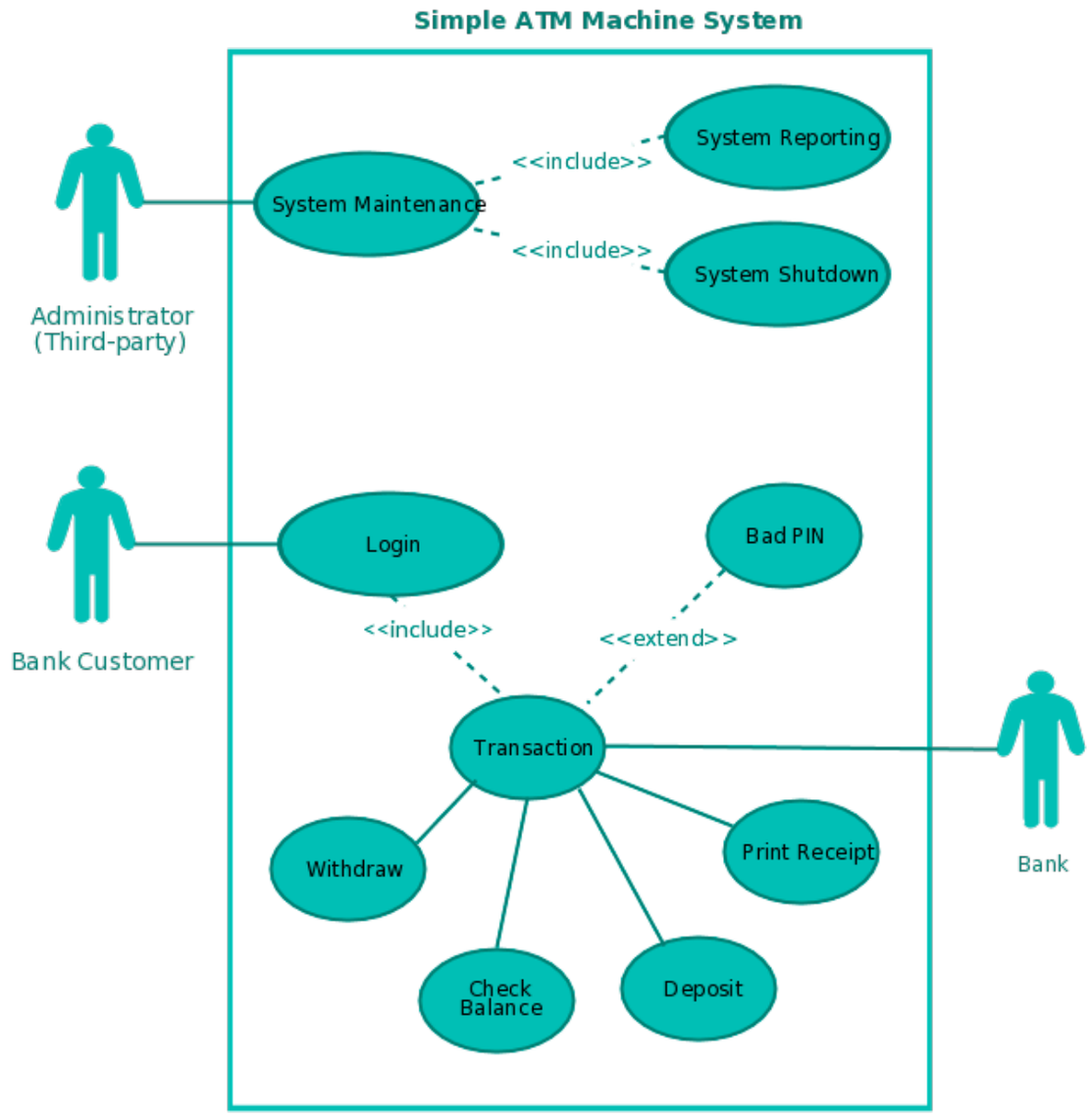


UC Diagram Example : Bank



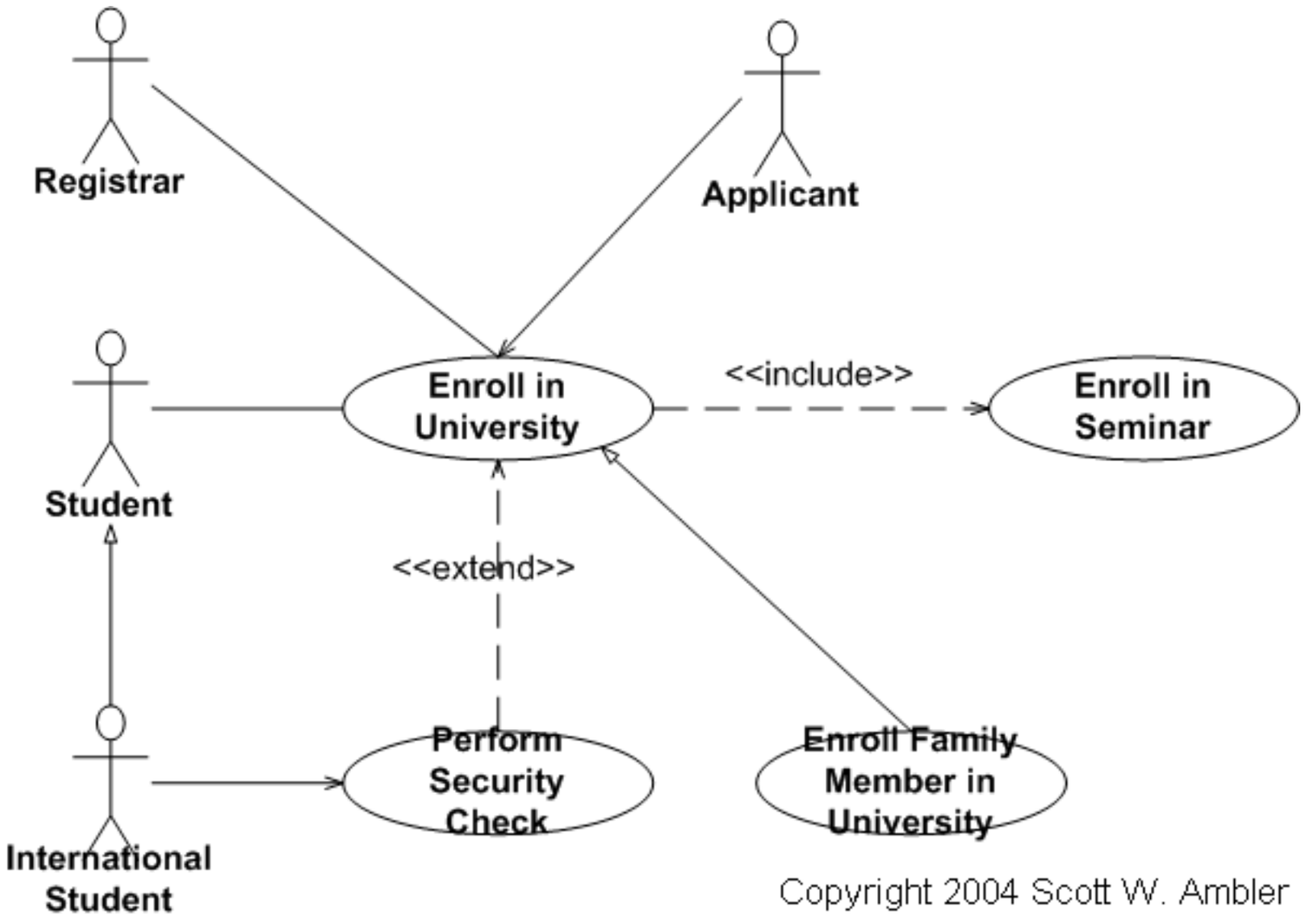
https://creately.com/blog/diagrams/use-case-diagram-tutorial/

UC Diagram Example : ATM

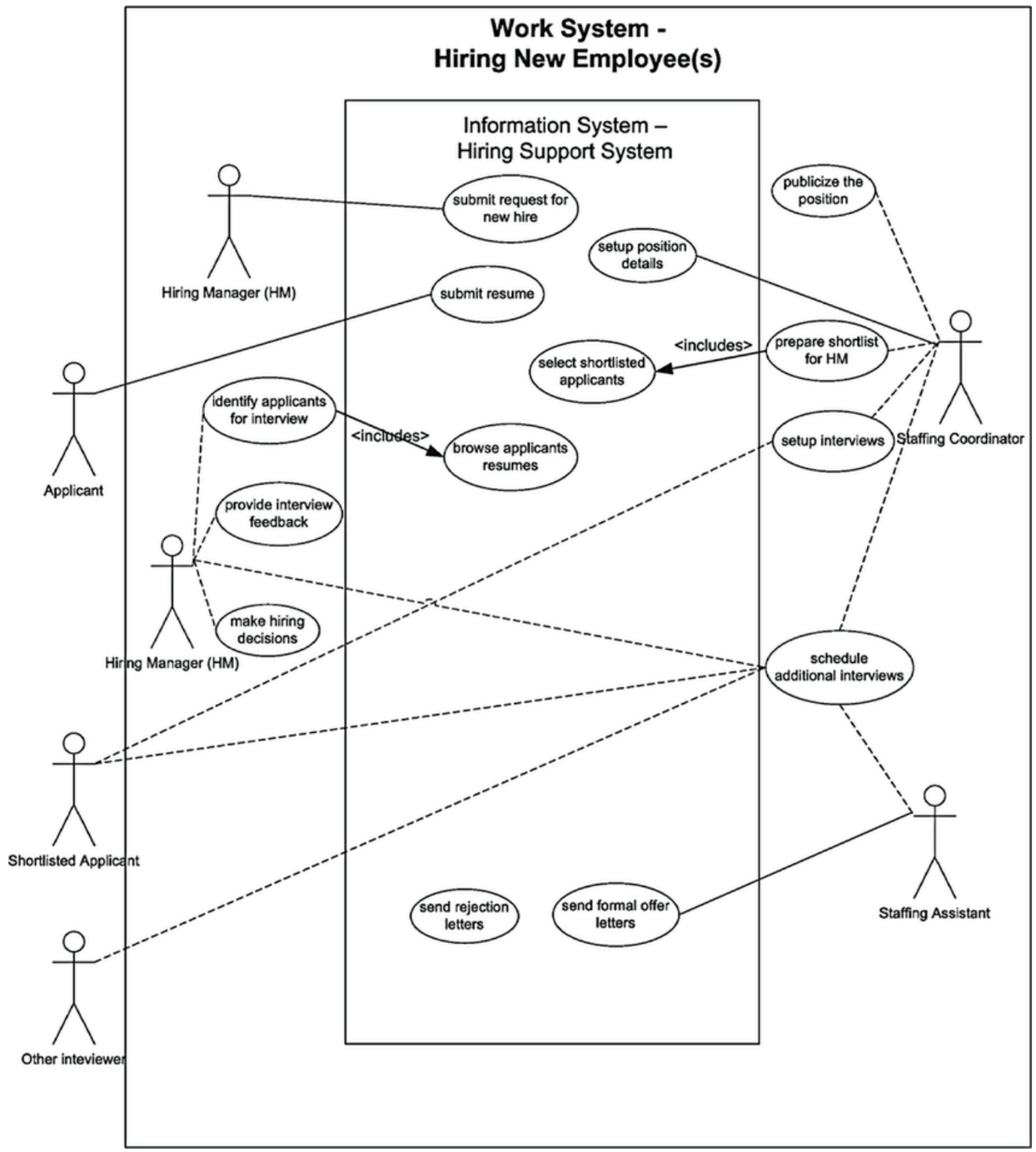


https://creately.com/blog/diagrams/use-case-diagram-tutorial/

UC Diagram Example
: University Administration



UC Diagram Example : Work Systems



Exercise

- 이 강의 Running problem의 예-은행 시스템-의 UC를 개발하라.

More Examples

- https://www.uml-diagrams.org/use-case-diagrams-examples.html

Use Case Diagram Guidelines for Better Use Cases ***

- https://creately.com/blog/diagrams/use-case-diagram-guidelines/