4. Penggunaan UML

Erwin Sutanto, S.T., M.Sc.

Daftar Isi

Penggunaan Standard UML Diagrams

Modeling Requirement with Use Case Diagram

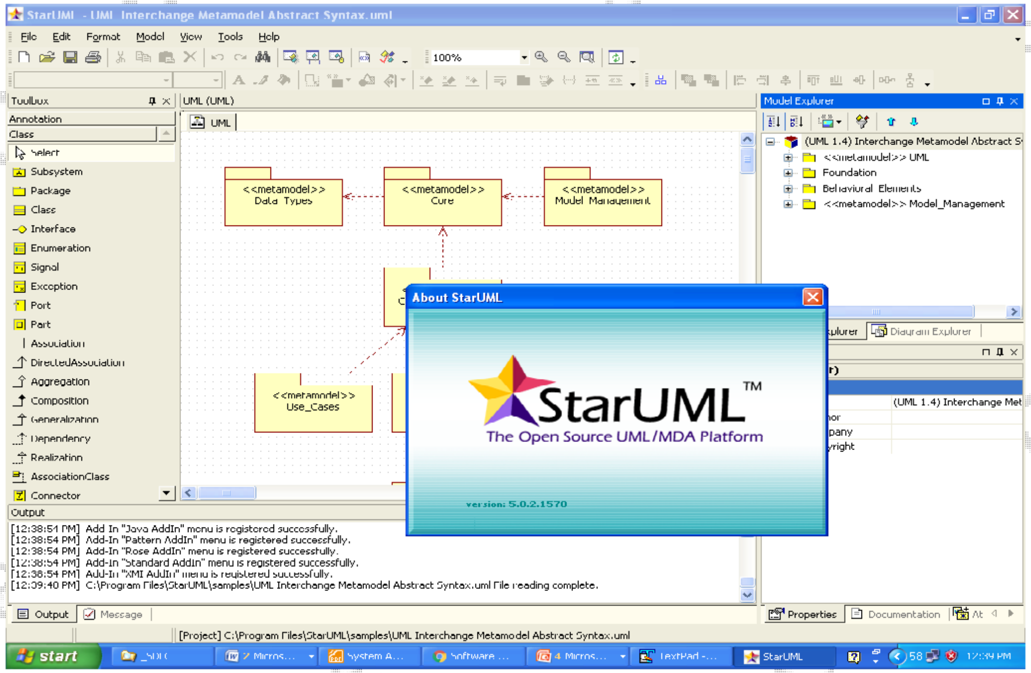
Modeling UI Navigation with Activity Diagram

Modeling Structure with Class Diagram

Mengaplikasikan Konsep SDLC dengan UML

1. Penggunaan Standard UML Diagrams

- To get more applied understanding, we will design for Books Catalog software.
- We will use Star UML 5.0 http://staruml.sourceforge.net/



```
[12:38:54 PM]  Add In "Java AddIn" menu is registered successfully.
[12:38:54 PM]  Add-In "Pattern AddIn" menu is registered successfully.
[12:38:54 PM]  Add-In "Rose AddIn" menu is registered successfully.
[12:38:54 PM]  Add-In "Standard AddIn" menu is registered successfully.
[12:38:54 PM]  Add-In "XMI AddIn" menu is registered successfully.
[12:39:40 PM]  C:\Program Files\StarUML\samples\UML Interchange Metamodel Abstract Syntax.uml File reading complete.
```

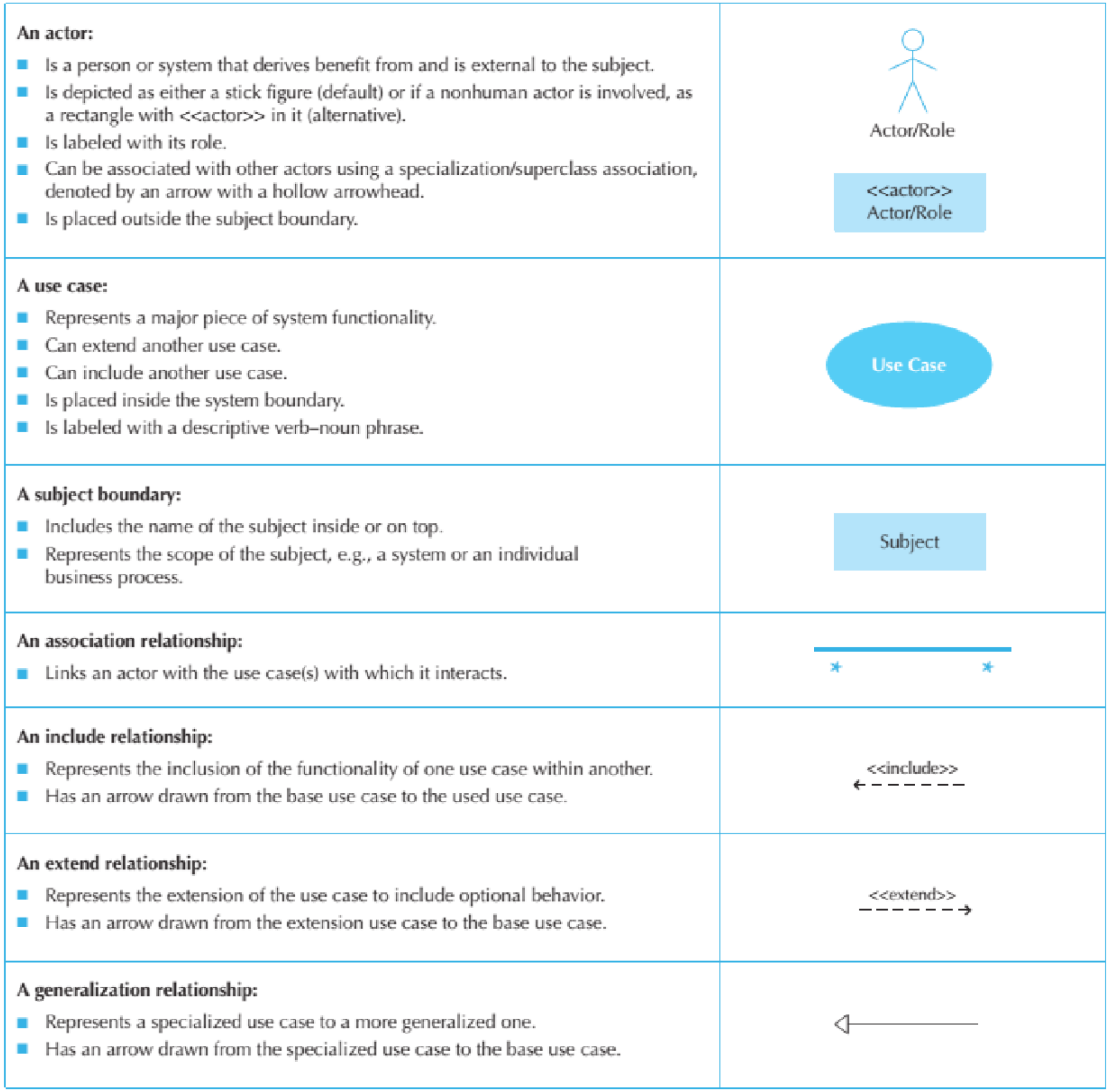
[Project] C:\Program Files\StarUML\samples\UML Interchange Metamodel Abstract Syntax.uml

2. Modeling Requirement with Use Case Diagram

- A **use case** is a methodology used in system analysis to identify, clarify, and organize **system requirements**.
- The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal.

Ref. 1 Chapter 4.2 – Use-Case Diagram

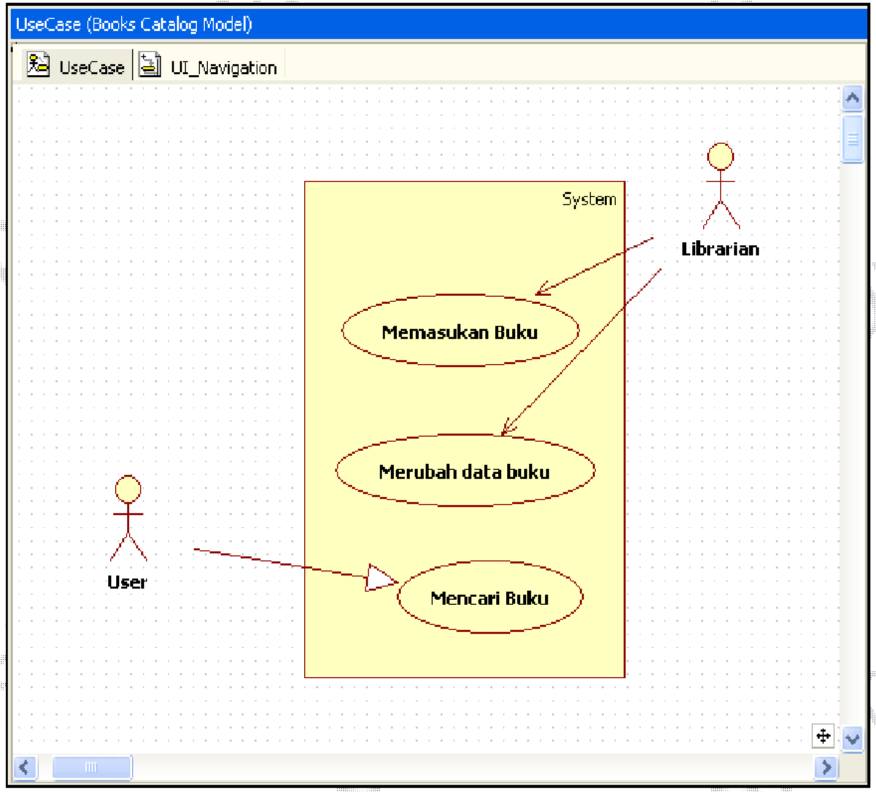
Syntax for Use-Case Diagram



An actor: - Is a person or system that derives benefit from and is external to the subject. - Is depicted as either a stick figure (default) or if a nonhuman actor is involved, as a rectangle with <<actor>> in it (alternative). - Is labeled with its role. - Can be associated with other actors using a specialization/superclass association, denoted by an arrow with a hollow arrowhead. - Is placed outside the subject boundary.	Actor/Role <<actor>> Actor/Role
A use case: - Represents a major piece of system functionality. - Can extend another use case. - Can include another use case. - Is placed inside the system boundary. - Is labeled with a descriptive verb–noun phrase.	Use Case
A subject boundary: - Includes the name of the subject inside or on top. - Represents the scope of the subject, e.g., a system or an individual business process.	Subject
An association relationship: - Links an actor with the use case(s) with which it interacts.	* *
An include relationship: - Represents the inclusion of the functionality of one use case within another. - Has an arrow drawn from the base use case to the used use case.	<<include>> ← - - - - - - -
An extend relationship: - Represents the extension of the use case to include optional behavior. - Has an arrow drawn from the extension use case to the base use case.	<<extend>> - - - - - - - →
A generalization relationship: - Represents a specialized use case to a more generalized one. - Has an arrow drawn from the specialized use case to the base use case.	◁————

Contoh Use Case Diagram

- Berikut adalah Use Case Diagram untuk penggunaan Katalog buku.

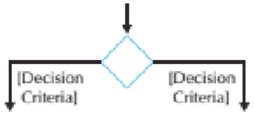
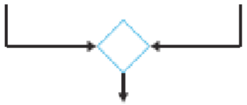


3. Modeling User Interface Navigation with Activity Diagram

- Business process models describe the different activities that, when combined together, support a business process.
- Business processes typically cut across functional departments (e.g., the creation of a new product will involve many different activities that will combine the efforts of many employees in many departments).
- Activity Diagrams can be used also to describe User Interface Navigation. That is how the window / UI flow from one to another.

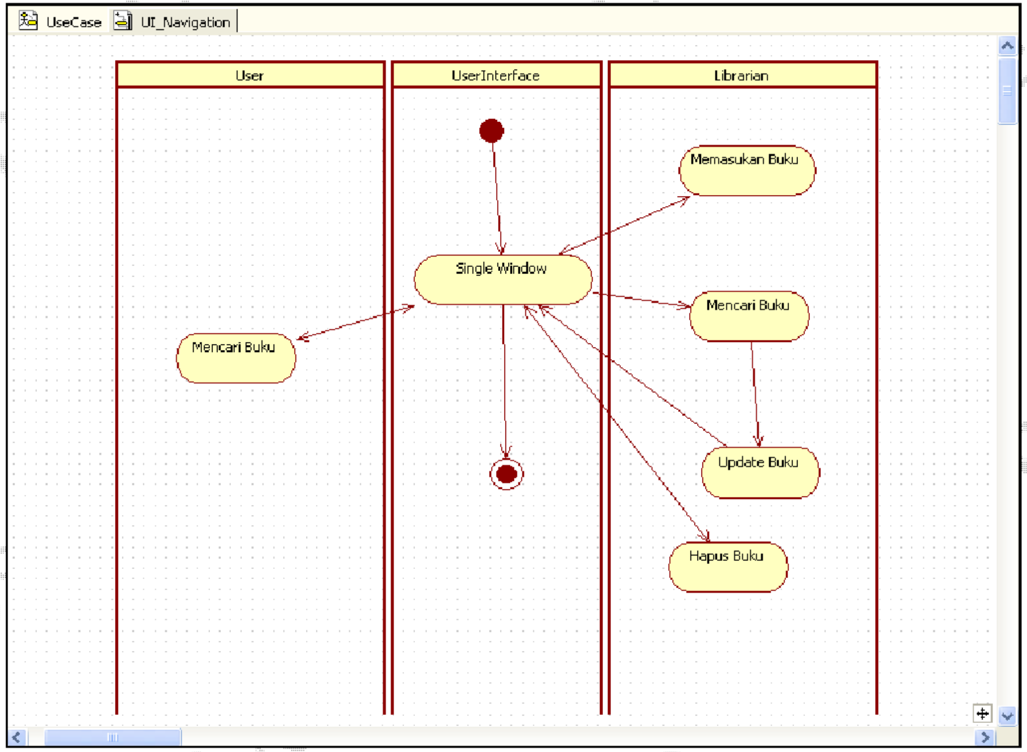
Ref. 1 Chapter 4.3 – Activity Diagram

Syntax for Activity Diagram

An action: - Is a simple, nondecomposable piece of behavior. - Is labeled by its name.	Action
An activity: - Is used to represent a set of actions. - Is labeled by its name.	Activity
An object node: - Is used to represent an object that is connected to a set of object flows. - Is labeled by its class name.	Class Name
A control flow: - Shows the sequence of execution.	⟶
An object flow: - Shows the flow of an object from one activity (or action) to another activity (or action).	- - - - - - - - →
An initial node: - Portrays the beginning of a set of actions or activities.	●
A final-activity node: - Is used to stop all control flows and object flows in an activity (or action).	◉
A final-flow node: - Is used to stop a specific control flow or object flow.	⊗
A decision node: - Is used to represent a test condition to ensure that the control flow or object flow only goes down one path. - Is labeled with the decision criteria to continue down the specific path.	
A merge node: - Is used to bring back together different decision paths that were created using a decision node.	

Contoh Activity Diagram

- Untuk Katalog buku ini, didisain dengan menggunakan satu jendela saja.
- Di mana ketiga operasi di atas dapat dilakukan dalam satu window atau View saja.

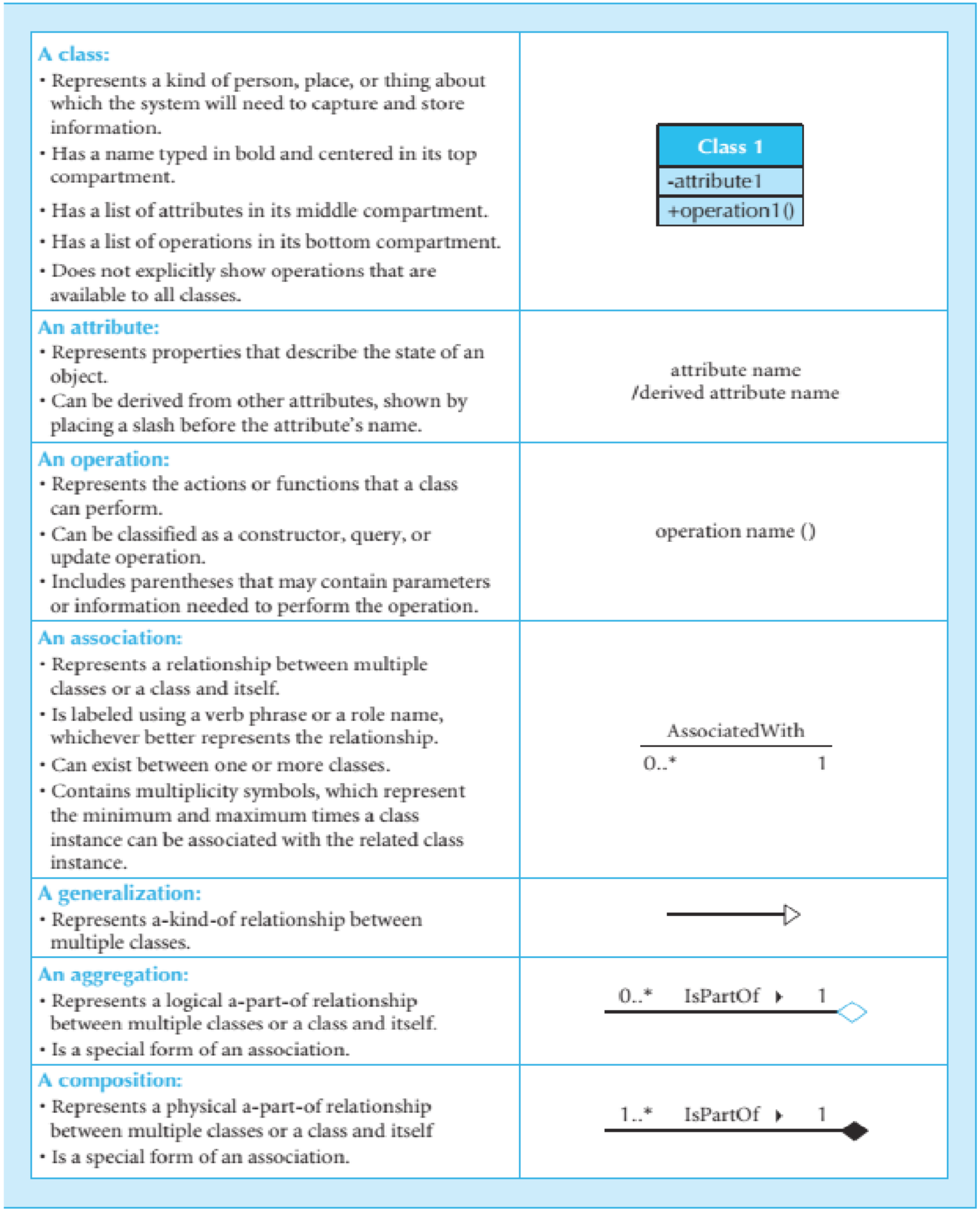


4. Modeling Structure with Class Diagram

- A class diagramis a static model that shows the classes and the relationships among classes that remain constant in the system over time.
- The class diagram depicts classes, which include both behaviors and states, with the relationships between the classes.
- Here is the structural design.
- There can be Layout class, Controller class, and any other kinds of class including Database class.

Ref. 1 Chapter 4.4 – Class Diagram

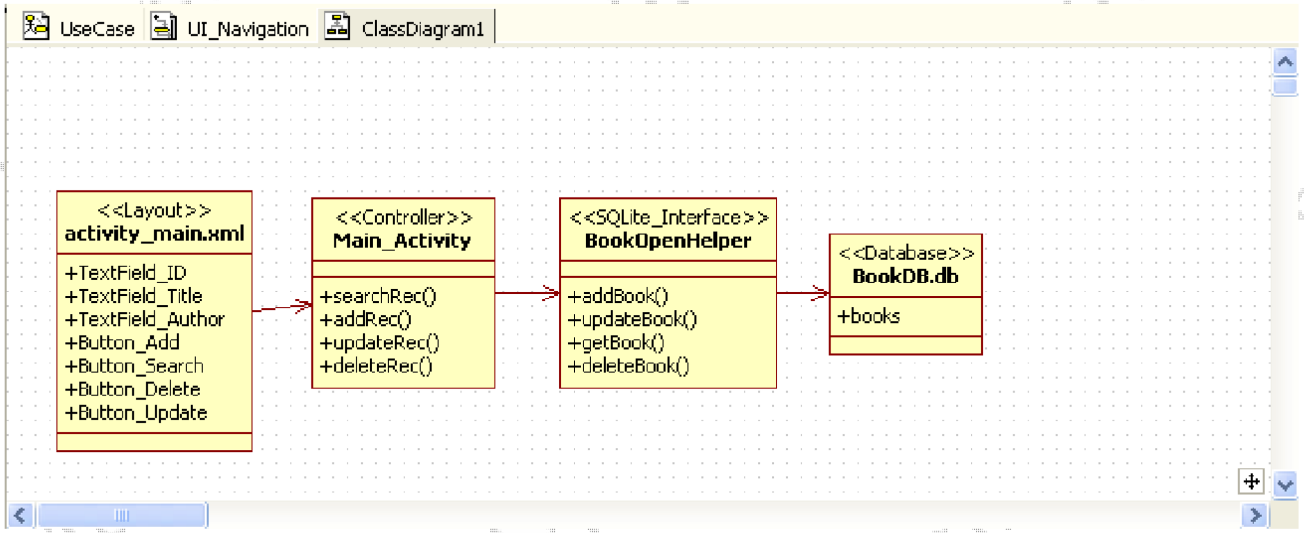
Syntax for Class Diagram



A class: • Represents a kind of person, place, or thing about which the system will need to capture and store information. • Has a name typed in bold and centered in its top compartment. • Has a list of attributes in its middle compartment. • Has a list of operations in its bottom compartment. • Does not explicitly show operations that are available to all classes.	**Class 1** -attribute1 +operation1()
An attribute: • Represents properties that describe the state of an object. • Can be derived from other attributes, shown by placing a slash before the attribute's name.	attribute name /derived attribute name
An operation: • Represents the actions or functions that a class can perform. • Can be classified as a constructor, query, or update operation. • Includes parentheses that may contain parameters or information needed to perform the operation.	operation name ()
An association: • Represents a relationship between multiple classes or a class and itself. • Is labeled using a verb phrase or a role name, whichever better represents the relationship. • Can exist between one or more classes. • Contains multiplicity symbols, which represent the minimum and maximum times a class instance can be associated with the related class instance.	AssociatedWith 0..* 1
A generalization: • Represents a-kind-of relationship between multiple classes.	—▷
An aggregation: • Represents a logical a-part-of relationship between multiple classes or a class and itself. • Is a special form of an association.	0..* IsPartOf ▸ 1 ◇
A composition: • Represents a physical a-part-of relationship between multiple classes or a class and itself • Is a special form of an association.	1..* IsPartOf ▸ 1 ◆

Contoh Class Diagram

- Di sini perlu ditekankan ketiga komponen dalam MVC: Model-View-Controller.
- Di mana didalamnya ditampilkan adanya Layout sebagai View, dan Controller, sedangkan Model ada pada komponen-komponennya.



<<Layout>>
activity_main.xml
+TextField_ID
+TextField_Title
+TextField_Author
+Button_Add
+Button_Search
+Button_Delete
+Button_Update

<<Controller>>
Main_Activity
+searchRec()
+addRec()
+updateRec()
+deleteRec()

<<SQLite_Interface>>
BookOpenHelper
+addBook()
+updateBook()
+getBook()
+deleteBook()

<<Database>>
BookDB.db
+books

5. Mengaplikasikan Konsep SDLC dengan UML

From the example above, SDLC Concept has been applied. That is where the four fundamental stages:

Plan - From the plan of building a software, a system analyst has to start the User Requirements from Stake holders.

Analysis - That User Requirement can be analyzed using Use-Case Diagram. Once the diagram is confirmed, it can be followed by user interface navigation using Activity Diagram.

Design - After the analysis stage, the structural model can be built using Class Diagram. After classes defined, the next diagram is the sequence Diagram. It will define the interaction between Classes. Then if there is a need to store data persistently, an ER Diagram can be used.

Implementation - Finally after all design models have been built. It can be stated that the system is ready for implementation. Here the IDE (Integrated Development Board) can be used. It will be following which programming language is going to be used. Here StarUML by default can generate codes for C++, & Java Language.

Referensi

Erwin S., Pemrograman Android Dengan Menggunakan Eclipse & StarUML, 2018 – Chapter 4

http://staruml.sourceforge.net/docs/user-guide(en)/toc.html

Denis,A., Barbara, H.W., David, T., 2009, Systems Analysis and Design with UML Version 2.0, 3rd Edition, John Wiley & Sons, Inc. – Chapter 5 & 6

Tugas

- Buatlah Diagram UML untuk proposal aplikasi yang telah dikumpulkan. Terdiri dari 3 macam diagram: Use Case Diagram, Activity Diagram, dan Class Diagram

- Deadline : 11-Mei-2019
- http://mooc.unair.ac.id