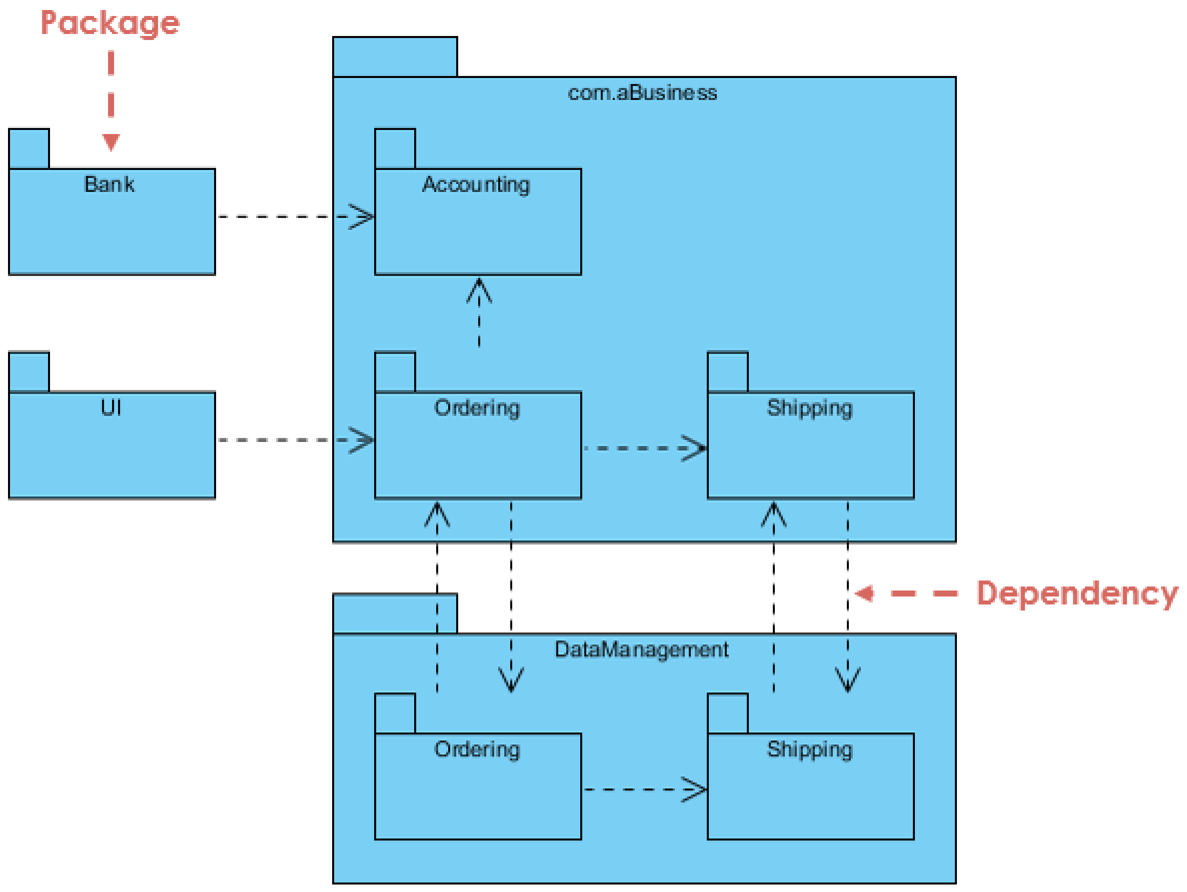
Package Diagram

Package diagram is used to simplify complex class diagrams, you can group classes into packages. A package is a collection of logically related UML elements.

The diagram below is a business model in which the classes are grouped into packages:

- Packages appear as rectangles with small tabs at the top.
- The package name is on the tab or inside the rectangle.
- The dotted arrows are dependencies.
- One package depends on another if changes in the other could possibly force changes in the first.



Package Diagram Namespace

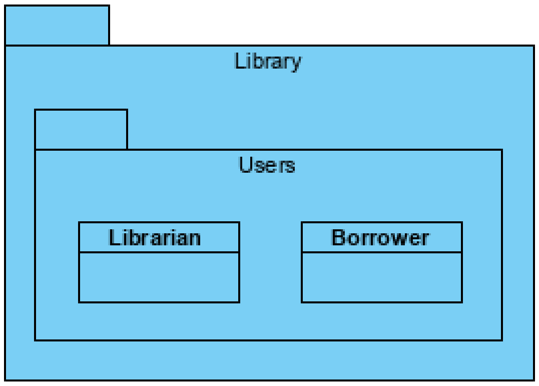
Packages contain different elements (packages too). A UML package establishes a namespace for specifying the context of a UML. A package defines what is known as an encapsulated namespace. When an element in one space needs to refer to an element in a different namespace, it has to specify both the name of the element it wants and the qualified name or pathname of the element (compare to file system analogy). You provide the fully scoped name, for example,

packageName::className

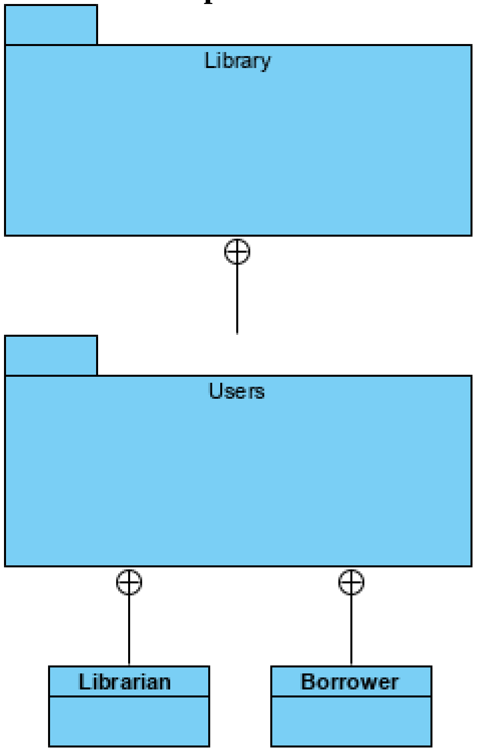
In Java, a fully-scoped name corresponds to specify the Java package. It is possible to specify visibility for owned and imported elements, such as, public or private as well.

- Qualified name of the class Librarian in the figure below is

Library::Users::Librarian

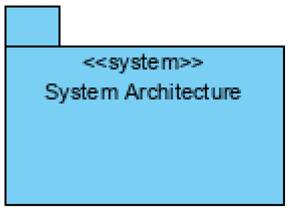


Alternative Representation of Package

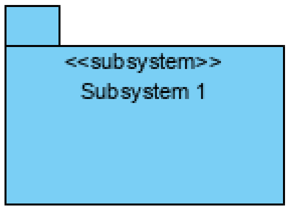


System and SubSystem

A system is represented as a package with the stereotype of <<system>> as shown in Figure below. The system represents all the model elements that pertain to the particular project. You can also break a system into <<business systems>> and <<application systems>> when building more detailed models to make them smaller and more workable. In the UML, packages are represented as folders.



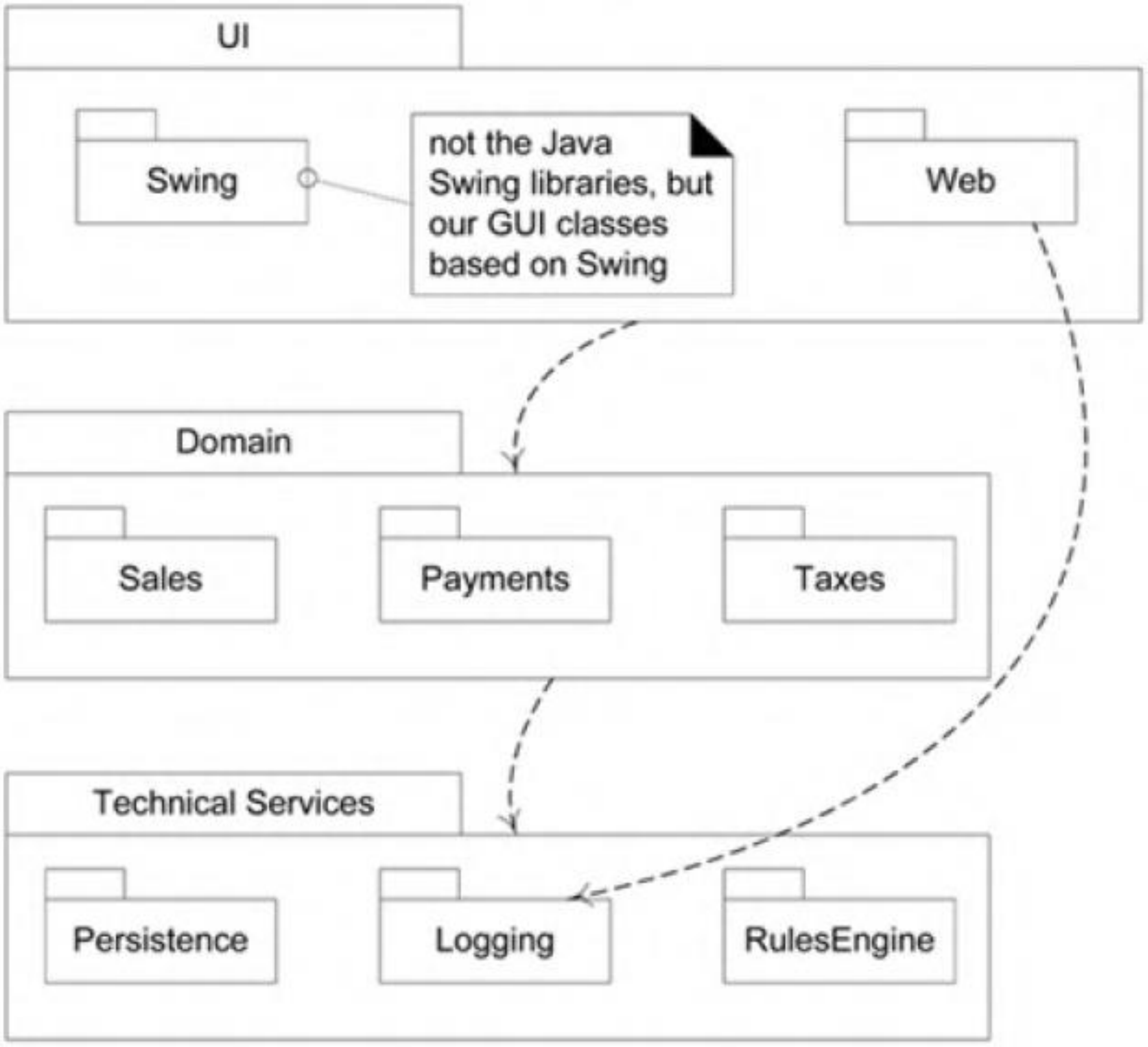
A subsystem is a grouping of model elements that are part of the overall system. Subsystems, like systems, are stereotyped packages with the stereotype of <<subsystem>> as shown in the Figure below.

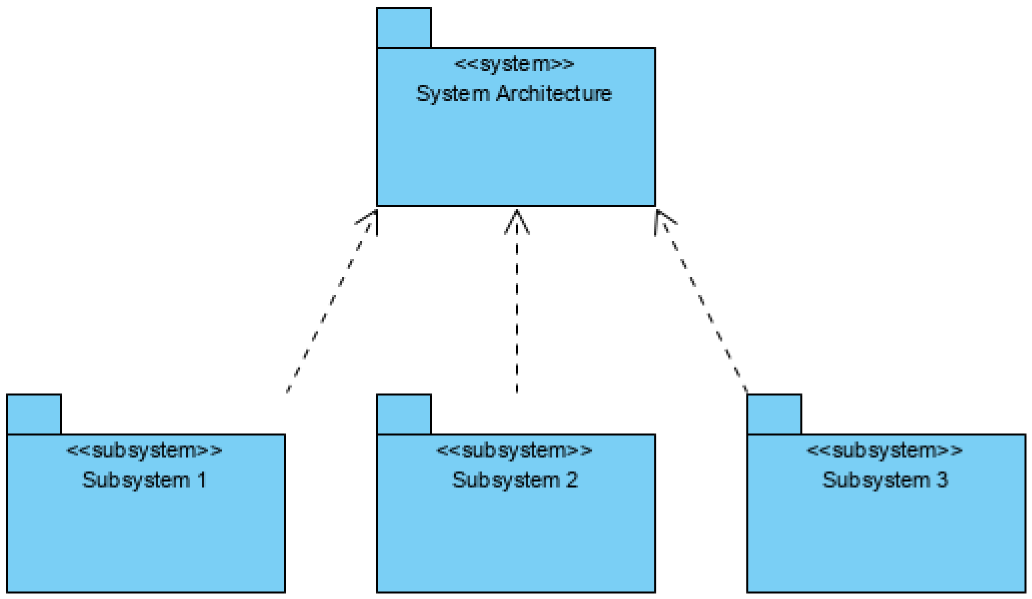


Packages for Software Structure

Because a system or subsystem is a stereotyped package, it has all the properties and rules of a package. This means those model elements that are contained by the system and subsystem are owned by that package and can only be part of them and no other.

The subsystem gives the project team an easy way to partition the system. Since a system contains multiple subsystems, everything contained within the subsystems is owned by the system that they roll up into. A diagram can display the logical architecture of a system.


UI
Swing
not the Java Swing libraries, but our GUI classes based on Swing
Web
Domain
Sales
Payments
Taxes
Technical Services
Persistence
Logging
RulesEngine



Question:

1. Draw logical architectural diagram for your project domain.
2. Implement your project-User Interface Layer