[CS304] Tutorial 3 - UML Diagrams for Requirement Analysis

> Part of this tutorial is based on online resources and textbooks, see References.

What is UML?

UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object oriented software and the software development process.

The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

UML Diagrams

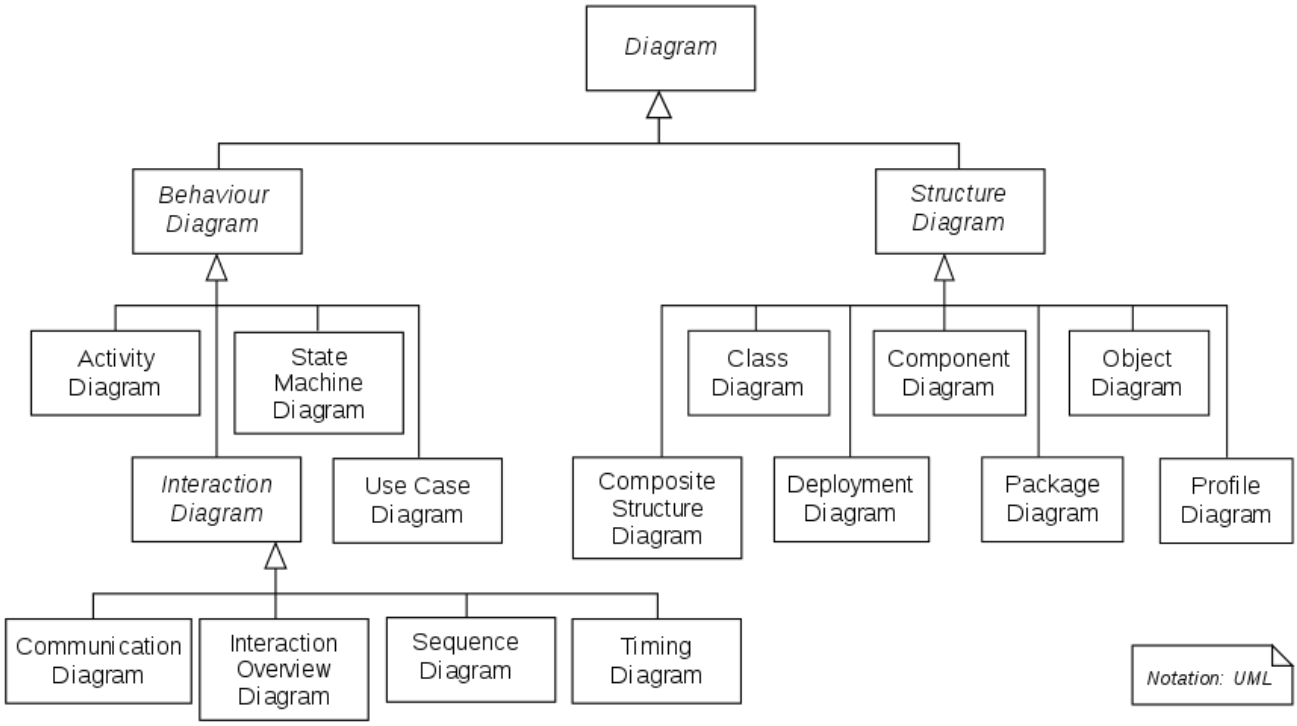
The first thing to notice about the UML is that there are a lot of different diagrams (models) to get used to. The reason for this is that it is possible to look at a system from many different viewpoints for different stakeholders, who require different level of detail.

For example, a coder needs to understand the design of the system and be able to convert the design to a low level code. By contrast, a technical writer is interested in the behavior of the system as a whole, and needs to understand how the product functions. The UML attempts to provide a language so expressive that all stakeholders can benefit from at least one UML diagram.

Types of UML Diagrams

UML 2 has many types of diagrams, which are divided into two categories:

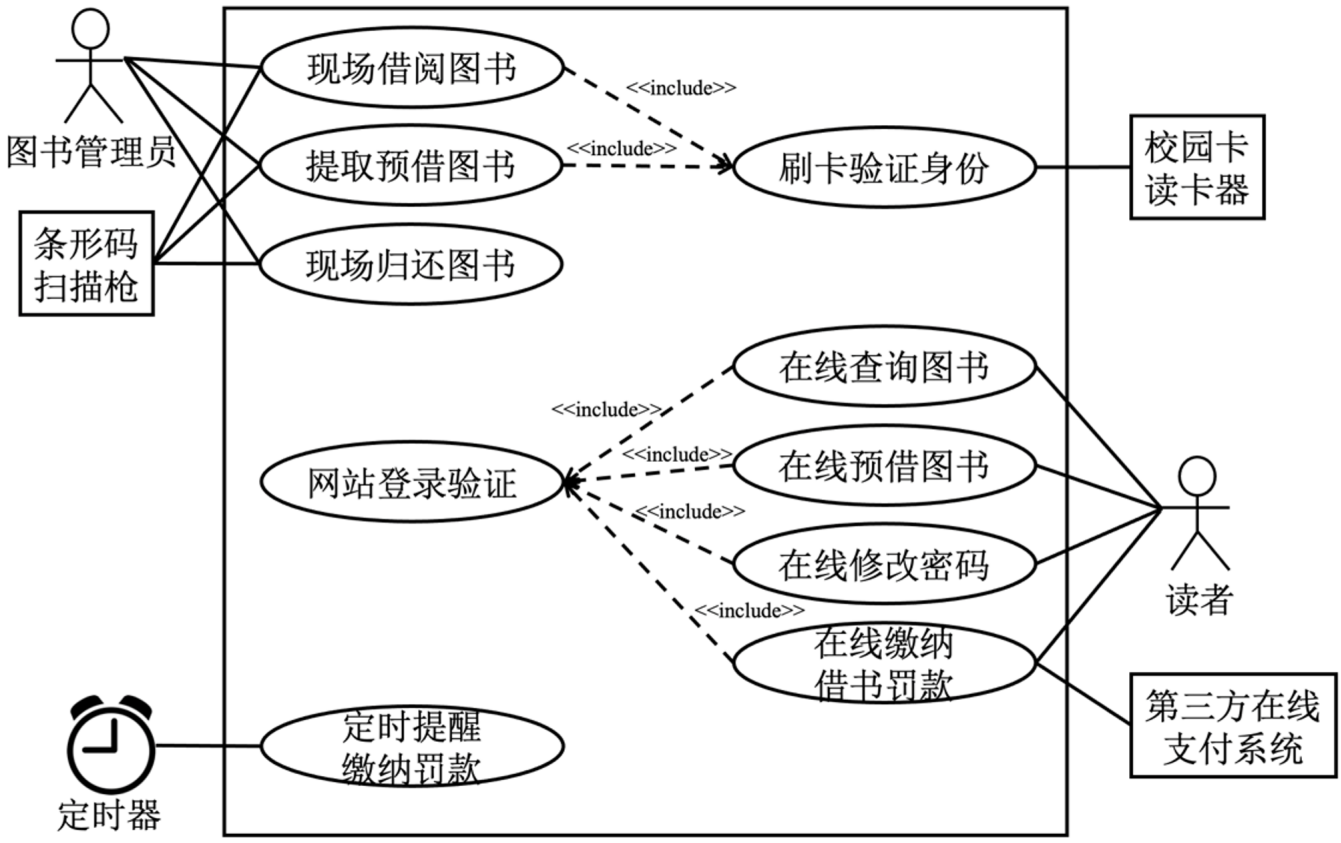
- **Structure Diagrams**: Structure diagrams represent the static aspects of the system. It emphasizes the things that must be present in the system being modeled. Structure diagrams depicts the elements of a specification that are **irrespective of time**, e.g., class diagrams.
- **Behavior Diagrams**: Behavior diagrams show the dynamic behavior of the objects in a system, which can be described as a series of **changes to the system over time**, e.g., the activity diagram describes the business and operational step-by-step activities of the components in a system. Interaction diagrams, a subset of behavior diagrams, emphasize the flow of control and data among the things in the system being modeled. For example, the sequence diagram shows how objects communicate with each other regarding a sequence of messages.



Examples

Use Case Diagram

Shows use cases, actors, and their interrelationships.

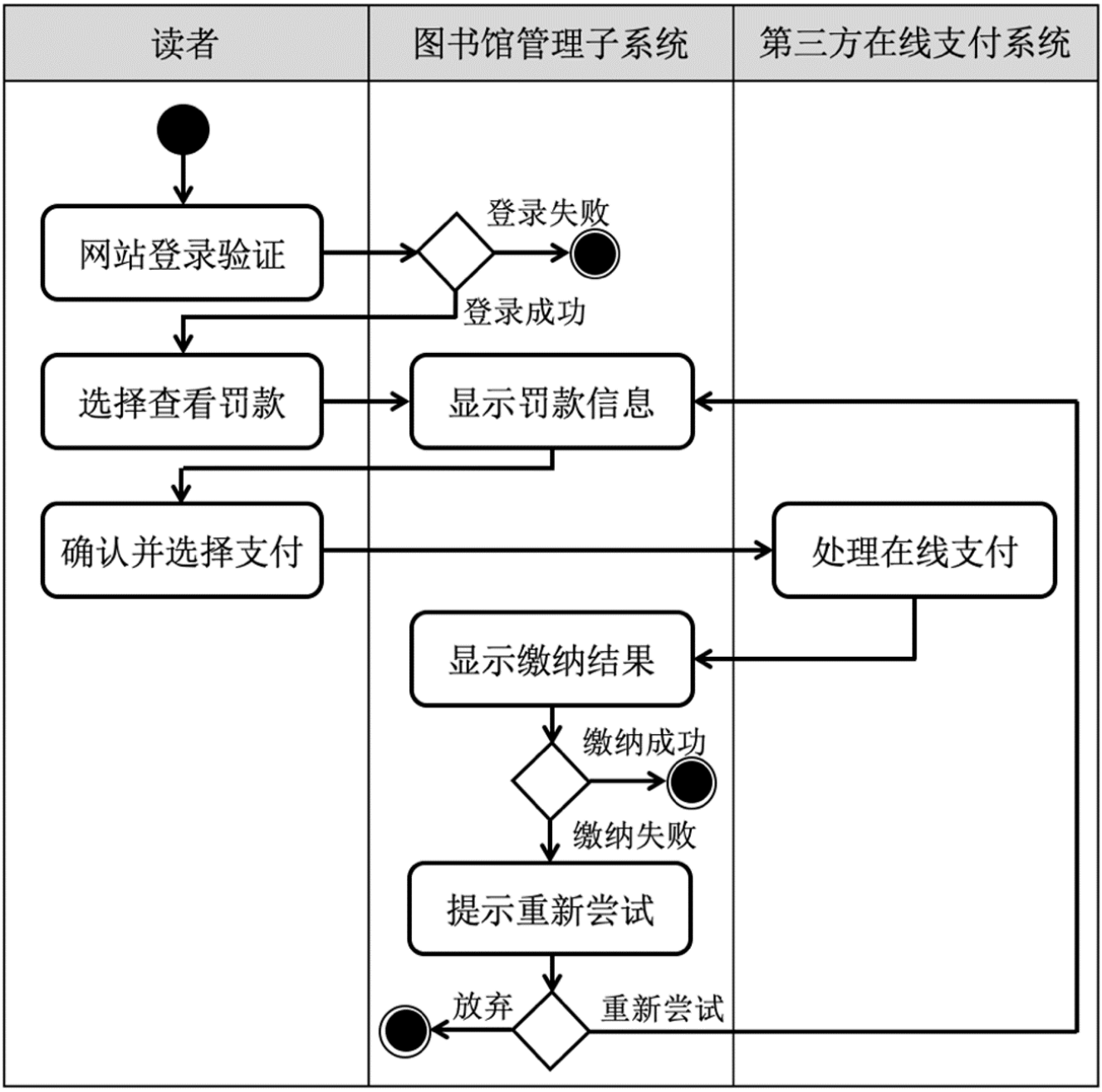


1.1 校园一卡通系统-图书馆管理子系统部分用例概览(UML用例图)

Use case diagram does not show detailed interaction between each use case. For that purpose, you may use *activity diagram* (or its variation *swim lane diagram*), or *sequence diagram*.

Activity Diagram or Swimlane Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.



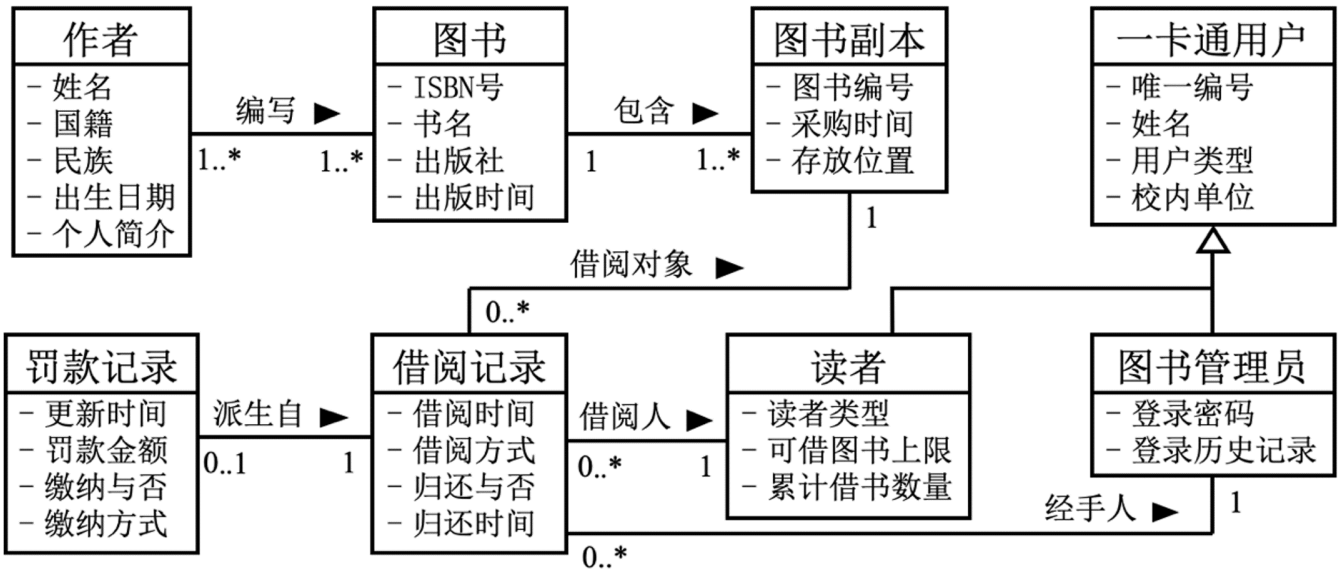
1.2 “在线缴纳借书罚款”用例详细描述(UML泳道图)

We might also use a natural-language template together with UML diagrams to provide a detailed description of the use case. The following example is a natural-language description of the use case in Figure 1.2.

用例名称： 在线缴纳借书罚款
参与者： 读者、图书馆管理子系统（目标软件系统）、第三方在线支付系统
目标： 读者希望能方便快捷地完成缴纳罚款的过程，同时罚款原因、金额、产生时间等明细信息公开透明，缴纳方式安全可信，结果能够准确、及时反馈；图书馆管理子系统希望读者能够及时缴纳罚款，同时第三方支付稳定可靠，支付结果准确无误。第三方在线支付系统希望单位时间内的支付请求量不要过高，以免对系统造成较大的压力。
前置条件： 当前用户拥有图书馆读者卡且读者卡处于正常可用状态。
触发条件： 读者登录网站并选择缴纳借书罚款。
主场景： 读者一次性完成借书罚款缴纳
1．读者进行网站登陆验证。
2．读者登录成功后选择查看罚款。
3．系统显示罚款信息。
4．读者确认罚款信息并选择支付。
5．第三方在线支付系统处理在线支付。
6．系统显示罚款缴纳成果结果。
其他场景： 1．登录失败
1）读者进行网站登陆验证。
2）系统显示登录失败。
2．罚款缴纳失败后再次尝试成功。
按照主场景执行到显示缴纳结果之后
1）系统显示缴纳失败并提示重新尝试。
2）读者选择重新尝试。
3）系统显示罚款信息后按照主场景继续执行，最终缴纳成功。
3．罚款缴纳失败后读者选择放弃
按照主场景执行到显示缴纳结果之后
1）系统显示缴纳失败并提示重新尝试。
2）读者选择放弃。
异常场景： 等待第三方在线支付结果超时
按照主场景执行到处理在线支付之后
1）系统等待第三方在线支付结果超时。
2）系统提示读者稍后电话联系图书馆服务中心核对支付结果。
发生频率：早上9点到晚上12点期间平均每小时10次，高峰期（如学生毕业离校手续办理截止日前夕）每小时可达500次。

Class Diagram

Class diagram describes the types of objects in the system and various kinds of static relationships which exist between them.

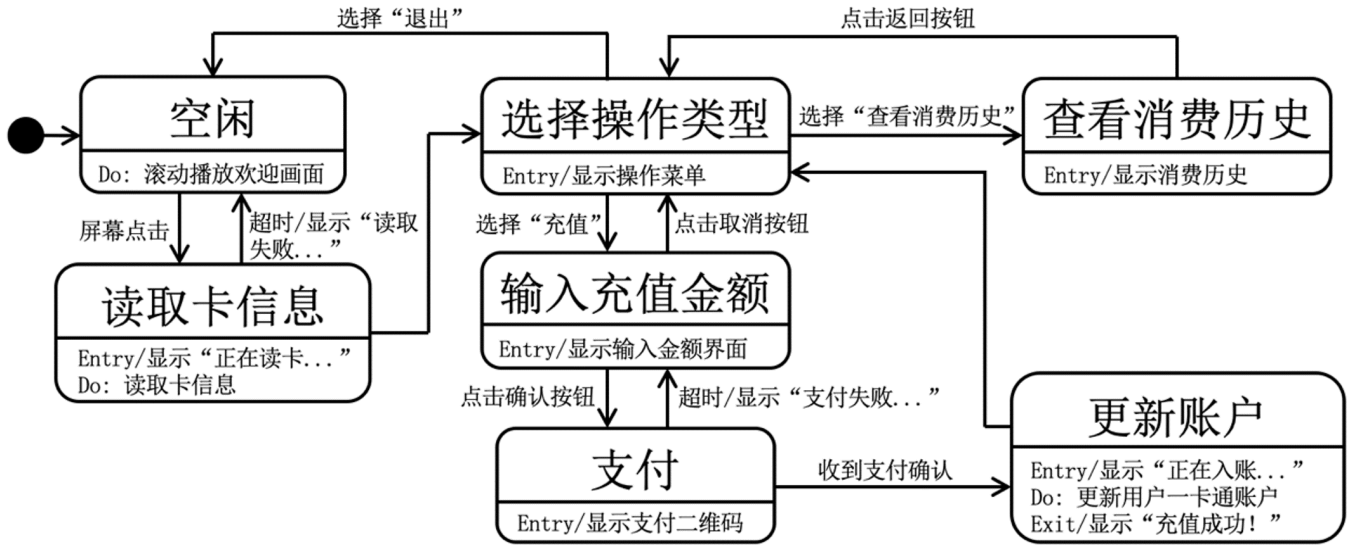


1.3 校园一卡通系统-图书馆管理子系统部分类模型(UML类图)

State Machine Diagram

A state machine diagram models the behaviour of a single object, specifying the sequence of events that an object goes through during its lifetime in response to events.

A state machine diagram typically consists of *states*, *transitions* (triggered by events), and *state actions* (e.g., entry action, do action, exit action).



1.4 校园一卡通系统自助充值子系统行为模型(UML状态机图)

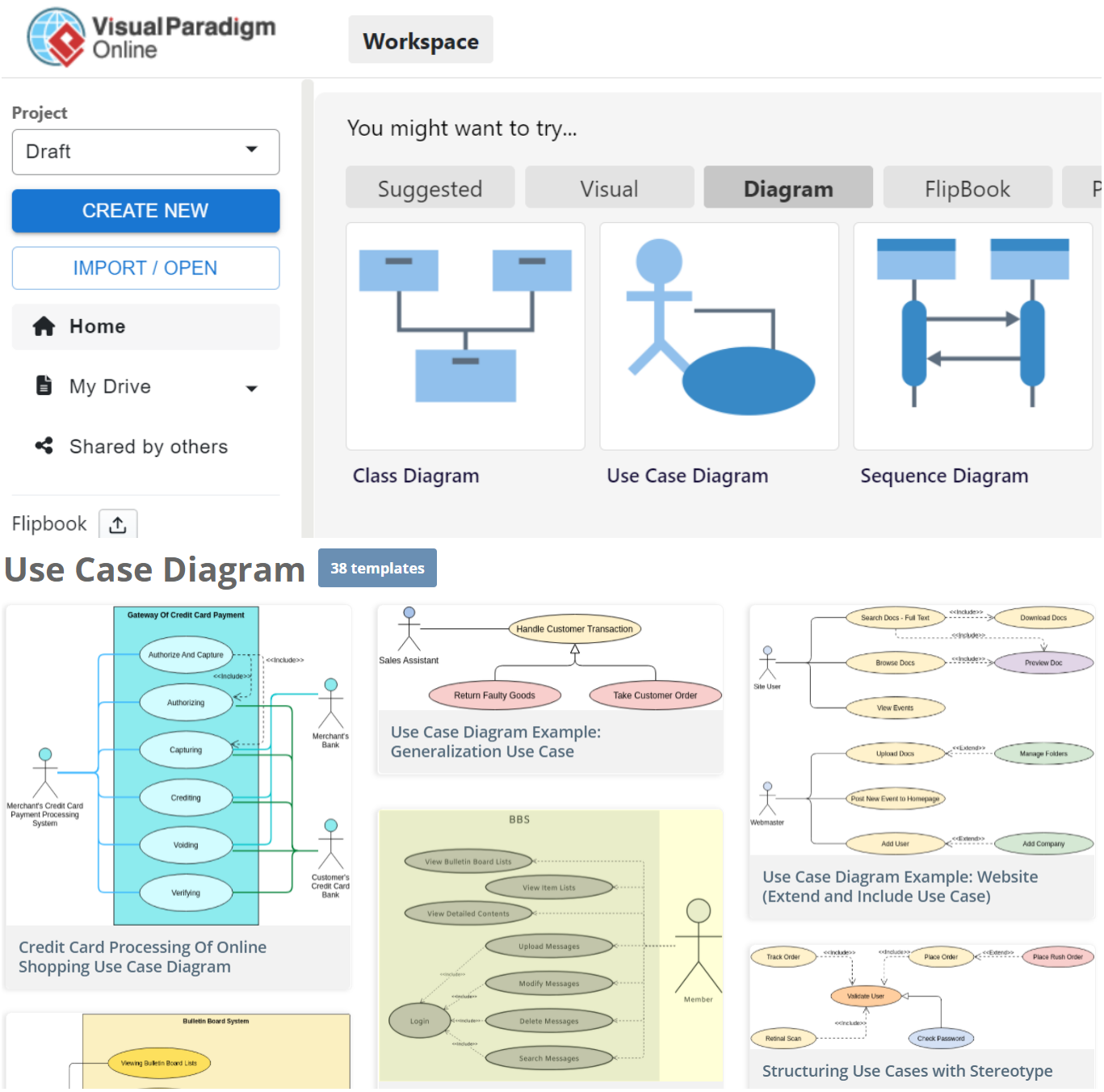
Tools

There are many tools available for designing and creating UML diagrams. For example,

- Visual Paradigm Online
- StarUML
- ArgoUML
- draw.io

- PlantUML
- ......

Take Visual Paradigm Online for example. You may browse various UML diagram templates as well as create your own diagrams



AI Assistants

There are also AI assistants available for creating UML diagrams. For example,

- ChatUML
- Diagramming AI

You may describe your diagram in natural language or sketches and the AI assistant will generate the corresponding UML diagram for you.

References

- 现代软件工程基础，第8章。彭鑫，游依勇，赵文耘。清华大学出版社
- Unified Modeling Language
- What is Unified Modeling Language (UML)?