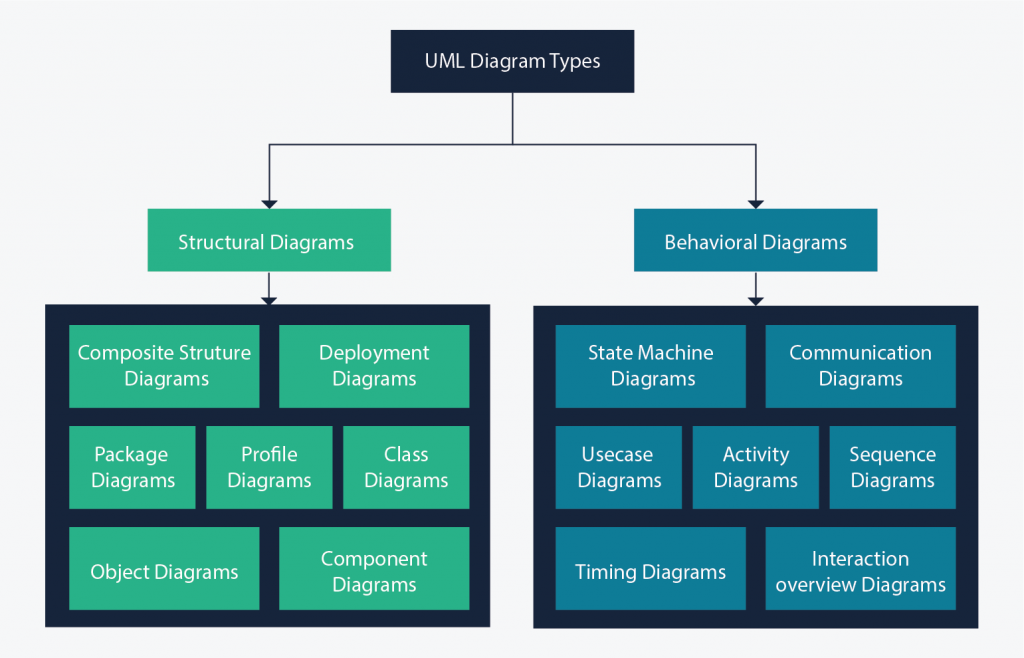
**UML diagrams**

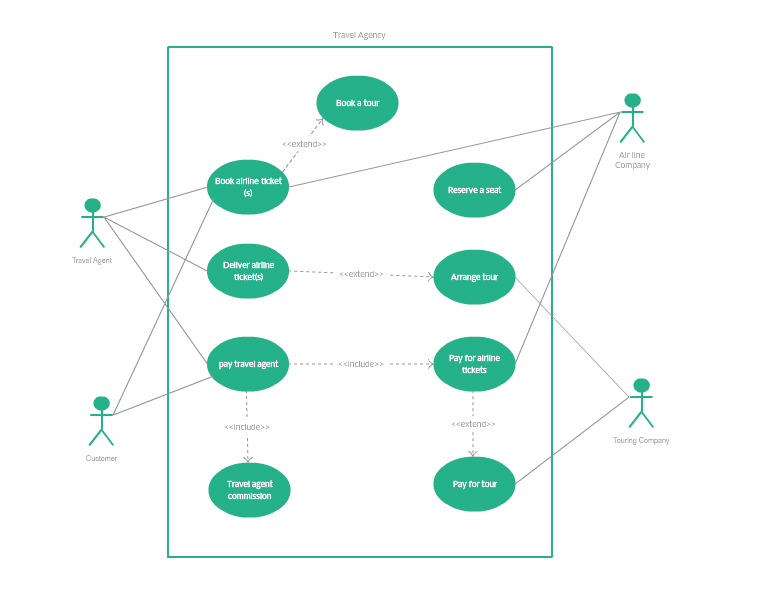
UML stands for **U**nified **M**odeling **L**anguage. It’s a rich language to model software solutions, application structures, system behavior and business processes. There are **14 UML diagram types** to help you model these behaviors.



* **Structure diagrams**: show the static relationships between the components in the system.
* **Behavior diagrams**: show how the components in the system react to each other, they capture how the system changes, and in some diagrams, how it changes over time.

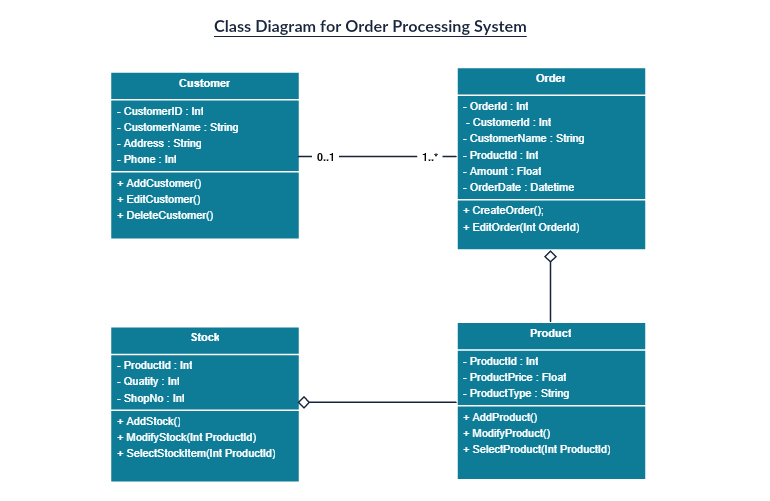
1. **Use case diagrams**

* Use case diagrams are created when you are looking at the requirements of your system or program.
* Use case diagrams give a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions interact.



#### Class diagrams

* Class diagrams are the main building block of any object-oriented solution. It shows the classes in a system, attributes, and operations of each class and the relationship between each class.
* Name at the top, attributes in the middle and operations or methods at the bottom. In a large system with many related classes, classes are grouped together to create class diagrams. Different relationships between classes are shown by different types of arrows.



1. **Sequence Diagram**

* Sequence diagrams are used to visualize both interactions within programs, business processes and IT infrastructures. They describe the sequence of interactions (messages) between actors and objects (things like databases or external interfaces).
* Sequence diagrams in UML show how objects interact with each other and the order those interactions occur. It’s important to note that they show the interactions for a particular scenario. The processes are represented vertically and interactions are shown as arrows.

A close up of a map

Description automatically generated

1. **Activity Diagram**

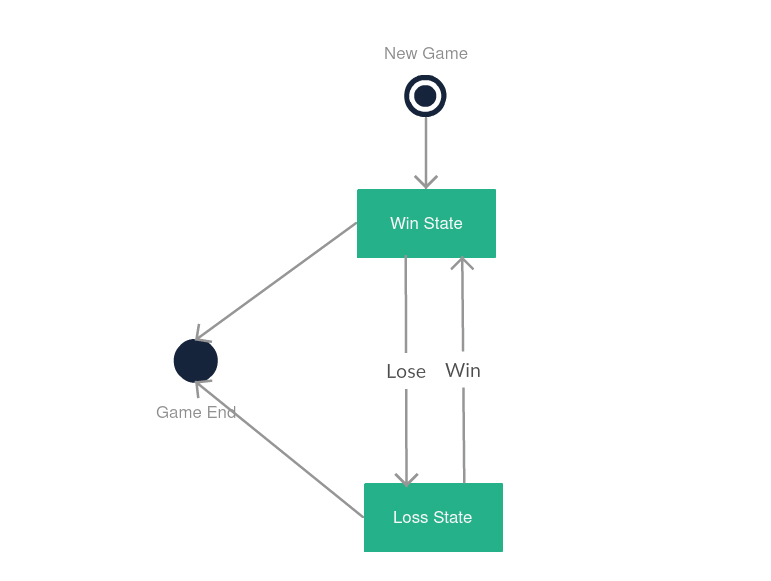
* Activity diagrams model the behavior of users and systems as they follow a process. They are a type of flow chart or workflow, but they use slightly different shapes.
* They can be used to describe the business workflow or the operational workflow of any component in a system.

A close up of a map

Description automatically generated

1. **State Diagrams**

* These are very useful to describe the behavior of objects that act differently according to the state they are in at the moment.
* State diagrams can be used to describe the various states that a system can be in for decades.



1. **Interaction overview diagram:**

These UML diagrams are a combination of an activity diagram and sequence diagrams, where each individual activity is placed in its own frame, so it is easier to program.

A screenshot of a social media post

Description automatically generated

#### Timing diagrams

This is a special type of sequence diagram that follows the behavior of objects over time. They are used more commonly when designing embedded systems, like programming control software for a washing machine.

#### Object diagrams

Object diagrams in UML are a specialized form of class diagrams. They visualize the more abstract class diagrams as if they have been instantiated and help software developers check whether their abstract structures will work in practice.

#### Component diagrams

A component diagram shows how the various components in a complex software system relate to each other and communicate using interfaces. They aren’t used for simpler or straight-forward systems.

#### Deployment diagrams

This is another specialized diagram for large and complex systems where software is deployed over multiple systems. These days, web services like [Amazon Web Services (AWS)](https://drawio-app.com/document-your-aws-architecture-with-cloudcraft-and-draw-io/) and [Google Cloud Platform (GCP)](https://drawio-app.com/updated-google-cloud-platform-icons-and-templates/) have their own notations – you can find these shapes in their draw.io shape libraries.

#### Package diagrams

Package diagrams show the dependencies in your system, and are used predominantly when you have a large codebase. Packages may contain use case diagrams to illustrate functionality of the system, and can visualize the layered architecture within a software system.

#### Composite structure diagrams

Where you have a complex software system with many different use cases, UML composite structure diagrams visualize what actually happens when the program is run. It represents the internal structure of a class and how it relates to different class components.

#### Profile diagrams

With a profile diagram, you can extend and customize UML to be used in a particular domain (e.g., aerospace, healthcare, financial) or platform (J2EE, .NET) with stereotypes, tagged values and constraints. Again, this type of diagram is usually used to model extremely large and complex systems.

**References**

<https://creately.com/blog/diagrams/uml-diagram-types-examples/> (Detailed description with images )

<https://drawio-app.com/uml-diagrams/> (brief)