1. Структурные диаграммы UML

- **Диаграмма классов**. Эта диаграмма, наиболее распространенная при разработке ПО, используется для изображения логической и физической структуры системы и показывает ее классы. Она похожа на блок-схему, потому что классы представлены в виде блоков. Эта диаграмма предлагает визуальное представление о различных классах и о том, как они взаимосвязаны. У каждого класса есть три секции:
- Верхняя секция: имя класса
- Средняя секция: атрибуты класса
- Нижняя секция: методы или операции класса

- **Диаграмма объектов**. Часто эта диаграмма используется как способ проверить диаграмму классов на точность. Другими словами, будет ли это работать на практике? Она показывает системные объекты и их взаимосвязи и предлагает лучшее представление о потенциальных недостатках проекта, которые необходимо исправить.

- **Диаграмма компонентов**. Также известна как блок-схема компонентов, она показывает логические группы элементов и их взаимосвязи. Другими словами, она дает упрощенное представление о сложной системе, разбивая ее на более мелкие компоненты. Каждый из элементов показан в прямоугольной рамке с названием, написанным внутри. Соединители определяют отношения / зависимости между различными компонентами.

- **Составная структурная диаграмма**. Этот тип редко используется кем-либо за пределами разработки программного обеспечения. Почему? Хотя она похожа на диаграмму классов, она требует более глубокого погружения, описывая внутреннюю структуру нескольких классов и показывая взаимодействие между ними. Если вы не разработчик, верхний уровень дает достаточно информации.

- **Диаграмма развертывания**. На этой диаграмме показаны аппаратные (узлы) и программные (артефакты) компоненты и их взаимосвязи. Она предлагает наглядное представление о том, где именно развернут каждый программный компонент.

- **Диаграмма пакетов**. Этот тип используется, чтобы изобразить зависимости между пакетами, которые составляют модель. Основная цель — показать взаимосвязь между различными крупными компонентами, которые образуют сложную систему.

- **Диаграмма профиля**. Этот тип меньше похоже на диаграмму и больше — на язык. Диаграмма профиля помогает создавать новые свойства и семантику для диаграмм UML путем определения пользовательских стереотипов, теговых значений и ограничений. Эти профили позволяют настраивать метамодель UML для различных платформ (например, Java Platform, Enterprise Edition (Java EE) или Microsoft .NET Framework) и доменов (например, моделирование бизнес-процессов, сервис-ориентированная архитектура, медицинские приложения и т. д.).

2. Поведенческие диаграммы UML

- **Диаграмма деятельности**. Этот тип изображает пошаговый процесс с четким началом и концом. Это набор операций, которые должны быть выполнены, чтобы достичь цели. Она показывает, как каждое действие ведет к следующему, и как все они связаны. Помимо разработки программного обеспечения, они могут использоваться практически в любой бизнес-среде. Их также называют картированием или моделированием бизнес-процессов.

- **Диаграмма вариантов использования**. В этом типе описывается, что делает система, но не то, как она это делает. Вариант использования — это набор событий, которые происходят, когда "оператор" использует систему для завершения процесса. Оператор определяется как кто-либо или что-либо, взаимодействующее с системой (человек, организация или приложение) из-за пределов системы. Таким образом, диаграмма вариантов использования визуально описывает этот набор последовательностей и представляет функциональные требования системы.

- **Обзорная диаграмма взаимодействия**. Эта зачастую сложная диаграмма похожа на диаграмму деятельности, так как обе показывают пошаговую последовательность действий. Но обзорная диаграмма взаимодействия — это диаграмма деятельности, составленная из разных диаграмм взаимодействия. Они используют те же аннотации, что и диаграмма деятельности (начальная, конечная, решение, слияние, разветвление и соединение узлов) с добавлением таких элементов, как взаимодействие, использование взаимодействия, ограничение по времени и ограничение продолжительности.

- **Временная диаграмма**. Когда время имеет критическое значение, используется этот тип диаграмм UML. Известная также как последовательность или диаграмма событий, она не показывает, как объекты взаимодействуют или изменяют друг друга. Функционально она показывает, как объекты и операторы действуют на временной шкале. Основное внимание здесь уделяется тому, сколько времени занимают события и какие изменения происходят в зависимости от ограничений продолжительности. Основные части временной диаграммы включают в себя:
    - Линия жизни: индивидуальный участник
    - Хронология состояний: разные состояния, через которые проходит линия жизни
    - Ограничение продолжительности: время, необходимое для выполнения ограничения
    - Ограничение по времени: время, за которое участник должен выполнить что-то
    - Возникновение разрушения: где заканчивается линия жизни объекта. Никакое другое событие не произойдет после появления разрушения на линии жизни.

- **Диаграмма конечного автомата**. Эта диаграмма, также называемая диаграммой состояний, применяется, когда поведение объектов является сложным, а детали — существенными. Она помогает описать поведение одного объекта (или иногда оператора) и то, как оно изменяется в зависимости от внутренних и внешних событий.

- **Диаграмма последовательности**. Эта визуально привлекательная диаграмма, популярная не только в сообществе разработчиков, хорошо показывает все типы бизнес-процессов. Она просто раскрывает структуру системы, показывая

последовательность сообщений и взаимодействий между операторами и объектами в хронологическом порядке. Диаграммы последовательности отображают простую итерацию и ветвление. Это имеет преимущества для многозадачности.

- **Диаграмма связи**. Диаграмма связи или сотрудничества аналогична диаграмме последовательности. Тем не менее, она подчеркивает связь между объектами, показывает организацию объектов, участвующих во взаимодействии, и предлагает более сложные итерации и ветвления.

В объектно-ориентированном моделировании существуют три вида связей между классами, которые наиболее важны: зависимость, представляющая связи использования между классами (включая уточнение, трассировку и связывание); обобщение, которое связывает обобщенные классы с их специализациями; ассоциации, описывающие структурные связи объектов. Каждая из этих разновидностей представляет отдельный способ комбинирования абстракций.

Стереотипы являются одним из трех типов механизмов расширяемости в унифицированном языке моделирования (UML). Они позволяют проектировщикам расширять словарь UML для создания новых элементов моделирования, получаемых из существующих, но имеющих определенные свойства, которые подходят для конкретной проблемы предметной области или для другого специализированного использования. Например, при моделировании сети вам могут понадобиться символы для представления маршрутизаторов и концентраторов. С помощью стереотипных узлов вы можете представлять их в виде примитивных строительных блоков.