

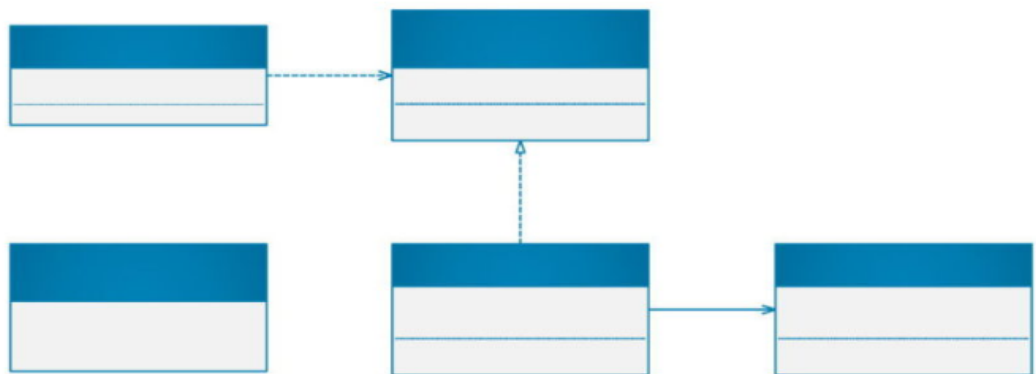
Эссе к 16.09.2023. 212-Соболев Никита

1) Структурные диаграммы UML:

а) Диаграмма классов (от англ. "class diagram") предназначена для представления внутренней структуры программы в виде классов и связей между ними.

Все сущности реального мира, с которыми собирается работать программист, должны быть представлены объектами классов в программе. При этом у каждого класса должно быть только одно назначение и уникально осмысленное имя, которое будет связано с этой целью.

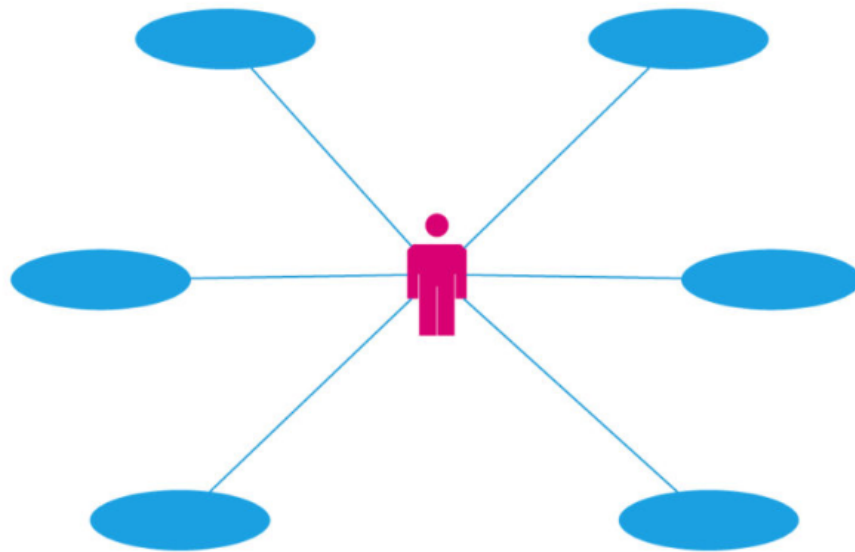
- Верхняя секция: имя класса
- Средняя секция: атрибуты класса
- Нижняя секция: методы или операции класса



б) Диаграмма профиля. Этот тип меньше похож 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.).

Поведенческие диаграммы UML:

а) 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.



б) 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:

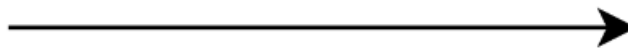
- Хронология состояний: разные состояния, через которые проходит линия жизни
- Ограничение продолжительности: время, необходимое для выполнения ограничения
- Линия жизни: индивидуальный участник

- 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.

в) 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

2) Виды отношений между классами.

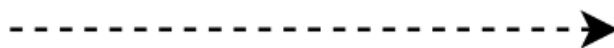
а) Отношение ассоциации. Отношение ассоциации используют, чтобы показать, что между классами (например, между двумя классами) существует некоторая связь. Обычно с помощью него на диаграмме классов показывают, что один класс пользуется функционалом другого класса.



Отношение ассоциации (от англ. "association relationship")

Стрелка ассоциации направлена от класса *пользователя* к классу *владелец* используемой функциональности. Для пояснения того, каким образом один класс использует другой класс, вы можете описать данный процесс в *вспомогательном тексте*.

б) Отношение зависимости. Отношение зависимости используют, чтобы показать, что изменение одного класса требует изменение другого класса.



Отношение зависимости (от англ. "dependency relationship")

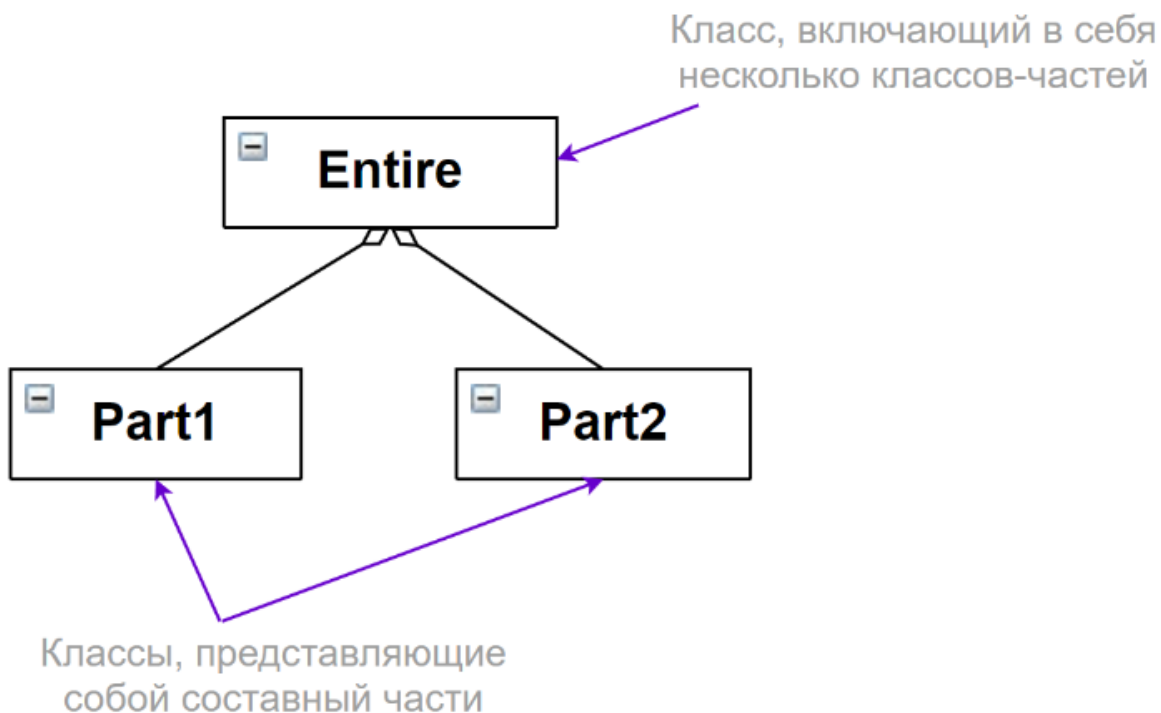
в) Отношение наследования. Оно используется, чтобы показать, что один класс является родителем (базовым классом или суперклассом) для другого класса (потомка, производного класса).

→
Отношение обобщения (от англ. "generalization relationship")
Отношение наследования (от англ. "inheritance relationship")


г) Отношение агрегации.

◊
Отношение агрегации (от англ. "aggregation relationship")

Показывает, что один из них включает в себя другой класс в качестве составной части. При этом класс-часть может и существовать обособленно от класса-целого



д) Отношение композиции. Является частным случаем отношения агрегации. Однако у него есть одно отличие – классы-части, которые он соединяет с классом-целым, не могут существовать обособленно.


Отношение композиции (англ. "composition relationship")

3) Что такое стереотип?

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.

Графически стереотип отображается как имя, заключенное в кавычки («», или, если такие кавычки недопустимы, <<>>) и расположенное над именем другого элемента. В дополнение или в качестве альтернативы он может быть обозначен соответствующей иконкой. Значок может даже заменить весь символ UML.