Визуальное моделирование. Язык UML.

Визуальное моделирование позволяет упростить процесс проектирования и разработки сложных программных систем. Визуальное моделирование позволяет достичь следующих целей:

1. Повышение качества программного продукта.
2. Сокращение стоимости проекта.
3. Реализация проекта в установленные сроки.

Одним из методов визуального моделирования является применение языка UML – Unified Modeling Language.

UML – визуальный язык моделирования общего назначения, предназначенный для спецификации, визуализации, проектирования и документирования всех артефактов – любых объектов, порожденных системой или используемые ею.

Язык UML используется для любых систем, не только информационных.

Области применения:

1. Телекоммуникация
2. Образование и наука
3. Промышленность
4. Медицинская электроника
5. Т. д.

Диаграммы UML:

1. Диаграммы структуры
   1. Классов – в понятиях ООП
   2. Компонентов
   3. Развертывания – распределение объектов системы в пространстве
   4. Объектов
2. Диаграммы поведения
   1. Последовательности – описывает объекты, которые участвуют в деятельности, их сообщения
   2. Состояния
   3. Вариантов использования
   4. Деятельности

Диаграмма – модель поведения или системные структуры.

Сущность:

1. Статические
   1. Объект
   2. Класс – описание множества объектов с общими атрибутами, определяющими состояние и операциями, определяющими поведение.
   3. Операции
2. Динамические – actor – лицо, использующее и взаимодействующее с системой
3. Аннотационные (комментарий)
4. Группирующие
5. Артефакты – файлы и данные, порождаемые или использующиеся системой
6. Узел – любой вычислительный ресурс, на котором размещаются объекты информационной системы
7. Компонент – модульная часть системы с четко определенным набором требуемых и предоставляемых интерфейсов
8. Интерфейс – набор операций, который может выполняться над экземпляром сущности

Динамические (поведенческие) сущности

1. Вариант использования
2. Состояние – в котором он удовлетворяет некоторому условию и осуществляет собственную деятельность или ожидает наступления некоторого события
3. Деятельность

Группирующие сущности – пакет

Отношения между сущностями

1. Зависимость

Изображается пунктирной линией

1. Ассоциация

Обычная линия

1. Обобщение

Тип связи, отражающий, что одна сущность является частным представлением другой сущности.

Изображается как стрелка от частного к общему.

1. Реализация

Подразумевает реализацию одной сущности посредством другой.

Пунктирная стрелка от реализующей к реализуемому сущностям.

Диаграммы вариантов использования (диаграмма прецедентов или USECASE (UC))

Содержит динамические лица и варианты использования.

UC позволяет описать участников взаимодействия пользователя с системой, функции, выполняемые ею, и отношения между участниками системы.

Цели построения UC:

1. Описание общего контекста моделируемой системы.
2. Сформулировать функциональные требования к системе.
3. Подготовка к документации для взаимодействия разработчика и заказчика.
4. Проектирование пользовательского интерфейса.

Отношения в UC:

1. Ассоциация

Между действующим лицом и вариантом использования

Обычная линия

Покупатель - заказ товара

1. Включение

Между вариантами использования

Подразумевает, что базовый прецедент не может быть выполнен без используемого.

Пунктирная стрелка с надписью <<include>> над ней.

Заказ товара --> <<include>> Оплата

1. Расширение

Между вариантами использования

Один прецедент может изменять поведение другого, расширяя его.

Пунктирная стрелка с стереотипом <<extend>>

Заказ товара <-- <<extend>> изменение корзины

Заказ товара расширяется изменением корзины.

1. Обобщение

Между действующими лицами

Стрелка от частного к общему

Потоки событий в UC диаграммах

1. Основной поток (1..н)

Описывает сценарий выполнения действий, который приводит к реализации варианта использования.

1. Альтернативный (А1..Ан)

Сценарий, который не рассматривается как ошибочный и описывает отклонение от основного потока.

1. Поток ошибок (Е1..Ен)

Описывает последовательность действий, которые приводят к возникновению ошибок в системе.

Классная работа. Описание форм.

Новый пользователь:

Форма регистрации.

Содержит поля для ввода нового пароля и почты пользователя.

Зарегистрированный пользователь:

Форма авторизации.

Позволяет пользователю войти в свою учетную запись после ввода корректного пароля и электронной почты.

Форма списка заказов.

Выглядит по-разному в зависимости от роли и привилегий пользователя.

Директор: отображает последние заказы, совершенные за последнее время на сайте и их статус. Могут применяться фильтры по ценам, категориям тортов, мастерам и другие.

Заказчик и менеджер по закупкам: отображает все заказы авторизованного пользователя и их статус. Также могут быть отсортированы и отфильтрованы.

Мастер: отображает заказы, выполняемые или выполненные авторизованным мастером и их статус.

Заказчик:

Главная страница:

Список всех товаров на сайте. Каждый товар можно добавить в корзину.

Страница товара:

На ней можно разглядеть товар и прочитать его описание. Указан состав торта, который при желании можно отредактировать. Также указан размер, который можно настраивать.

Корзина:

Тут можно просмотреть выбранные товары.

Форма оформления заказа:

На этой форме вводится адрес и платежные данные, а также дополнительные данные, связанные с доставкой.

Форма отклонения заказа:

Подтверждение отклонения оформления заказа.

Мастер:

Главная страница:

Список изготавливаемых в данный момент изделий.

Форма отчета:

На ней мастер составляет отчет о сбое оборудования.

Страница изделия:

На ней отображается требуемый состав торта, размер, оборудование, которое должно применяться при создании, дата совершения заказа.

На ней же мастер может изменять статус готовности изделия.

Форма отчетов об ингредиентах:

Тут мастер может отследить наличие ингредиентов и их сроки годности и создать запрос на их заказ.

Директор:

Главная страница:

Отображает последние уведомления, такие как: информация о последних сбоях, общий статус готовности заказов, отставания от сроков, активность мастеров и менеджеров по закупкам.

На временной ленте показаны последние события: изменения статусов заказов, приготовление изделий и прочее.

Планировка цехов:

На этой странице директор отслеживает нагрузку по цехам и может изменить место работы отдельных мастеров (по окончанию выполнения заказа).

Окно отчетов по инструментам:

Тут содержатся отправленные мастерами отчеты о работоспособности инструментов. Там же указаны контактные данные, которые помогут приступить к решению проблем. Там директор для удобства может делать пометки об отдельном оборудовании и указать статус решения проблемы, который увидят мастера.

Окно инспекции оборудования:

Тут отображена вся информация об оборудовании в цехах и имеющихся инструментах.

Окно инспекции ингредиентов:

Тут директор проверяет наличие имеющихся ингредиентов, их сроков годности и может совершать их закупку через заданные контактные данные. Тут же содержится информация о расходе ингредиентов, формирующиеся исходя из отчетов мастеров и менеджеров по закупкам.

Менеджер по закупкам:

Главная страница:

Отображает отслеживаемых клиентов, позволяет связаться с ними и изменить статус обслуживания каждого заказа клиента.

Форма отчетов об ингредиентах:

Тут мастер может отследить наличие ингредиентов и их сроки годности и создать запрос на их заказ.

Форма связи с мастерами:

Содержит информацию о загруженности мастеров и статусах выполнения заказов.

Менеджер по работе с клиентами:

Главная страница:

Отображает совершаемые заказы отдельных клиентов и менеджеров по закупке.

Страница заказа:

Содержит всю информацию о заказе. Отсюда заказ можно распределить на конкретного мастера.

Диаграмма последовательности

Линия жизни – период, в течение которого объект существует в системе.

Фокус управления – период, в течение которого объект выполняет некоторое действие.

Сообщения:

1. Синхронное (ожидает ответа)
2. Асинхронное (без ответа от получателя)
3. Ответное
4. Создание <<create>>
5. Уничтожение <<destroy>>

Диаграмма последовательности строится для отдельно взятого варианта использования. В качестве объектов могут быть взяты классы, а в качестве сообщений их операции. Также в качестве объектов могут быть рассмотрены экранные формы.

Д. последовательности:

1) Регистрация () (успешная)

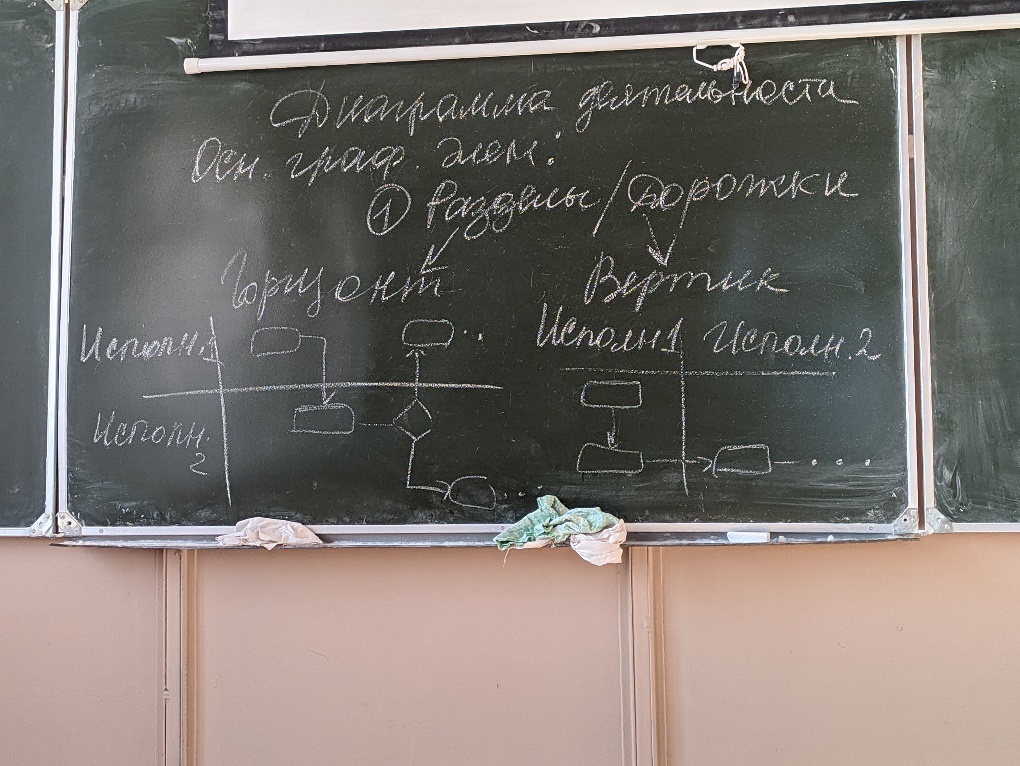
2) Авторизация () (успешная)

Диаграмма деятельности

Диаграмма деятельности позволяет описать набор действий, их последовательность и логику выполнения, описывающие некоторый сценарий.

Основные графические элементы:

1. Разделы / дорожки



1. Действия

Изображается в виде прямоугольника с закругленными углами.

1. Потоки

(стрелка)

Если поток между двумя действиями, то это поток управления

Если между объектами – поток данных

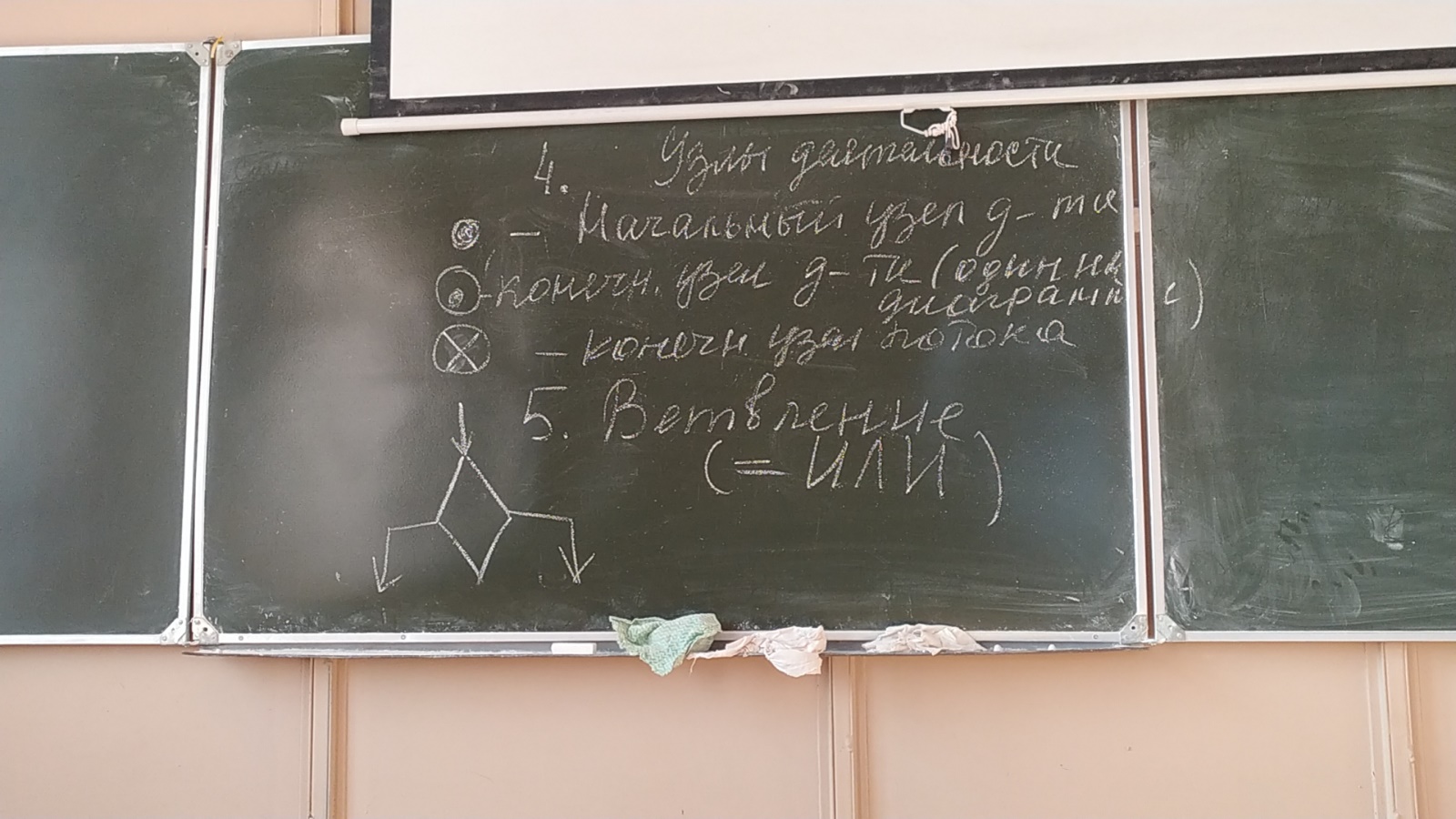
1. Узлы деятельности

Большой закрашенный кружок – начальный узел

Кружок с точкой внутри – конечный узел деятельности (один на диаграмме)

Кружок с крестиком – конечный узел потока

1. Ветвления



1. Слияние

Содержит точку слияния и разделения. После разделения потоков несколько, после слияния – один.

Точка слияния обеспечивает синхронизацию нескольких параллельных потоков. Они выполняются параллельно.

Диаграмма состояния

Состоит из состояний и объектов.

Entry – действие, при котором объект переходит в это состояние.

Do – действия, которые выполняются в состоянии.

Exit – действие, которое переводит объект в другое состояние.

Есть узел начального и конечного состояния.

На диаграмме состояний также могут быть отображены граничные условия. Они изображаются как стрелки, над которыми в квадратных скобках указываются условия.

Диаграмма реализации

К ней относится диаграмма компонентов и диаграмма размещения.

Диаграмма реализации описывает особенности физической реализации системы. Диаграмма компонентов позволяет описать все артефакты и компоненты, входящие в систему. Диаграмма размещения описывает физическое распределение компонентов.

Элементы:

1. Компонент
2. Интерфейс

Отношения:

1. Ассоциация ->
2. Зависимость -->

Виды компонентов:

Компоненты – рабочие файлы (док и исходным кодом, файлы справки, библиотеки, таблицы и т.д.)

Исполняемые файлы

Вычислительный узел – на нем может располагаться компонент.

Диаграмма классов

Диаграмма классов позволяет описать статическую структуру моделируемой программной системы. Под классом понимается множество однородных объектов, имеющих одинаковые атрибуты и операции.

Атрибут служит для представления отдельного свойства или признака объекта, который является общим для всех экземпляров класса.

Описание атрибута:

<квантор видимости><имя атрибута>:<тип атрибута>=<исходное значение>

Name:char(30)

Квантор видимости отражает способ взаимодействия между разными классами в пределах программы или пакета.

Атрибут или операция доступны или видны в пределах класса или программы. Тип protected, изображаемый с помощью решетки, обозначает, что атрибут или операция недоступны во всех классов кроме дочерних.

Private –

Public +

Ассоциации: агрегатная и композиционная.

Агрегатная связь – связь между особый тип отношений, при котором один класс является частью другого.

Композиция – одно не может существовать без другого.

Зависимость – при изменении одного класса будет меняться другой.

Обобщение – один класс является подклассом другого.