ВЫБОР ГРАФИЧЕСКОЙ ЧАСТИ ДИПЛОМНОГО ПРОЕКТИРОВАНИЯ ДЛЯ УЧАЩИХСЯ СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ

Д.В. КАРПОВИЧ, Е.А. САЛЬНИКОВА, В.И. КОНЧАНИН
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники» филиал «Минский радиотехнический колледж»

Аннотация: Качественная визуализация является неотъемлемой частью успешного проектирования дипломных работ. Графическая часть способствует наглядному представлению идей, концепций и функциональности проекта, делая его более доступным и понятным для пользователей. Для студентов, обучающихся среднему специальному образованию, выбор подходящих графических компонентов является важным этапом разработки, требующим особого внимания. В этой статье будут представлены несколько типов диаграмм, применяемых в различных видах приложений. Также будут рассмотрены их особенности, преимущества и области применения с целью помочь студентам выбрать наиболее подходящие графические компоненты для их дипломных проектов.

Графическое представление информации играет важную роль в передаче конкретных идей, усилении сформулированных выводов и выделении главных акцентов. Например, правильно подобранная диаграмма классов может значительно повлиять на восприятие информации. Такая диаграмма помогает лучше понять структуру программы, иллюстрируя взаимосвязи между классами и их методами. Это упрощает задачу программистам, помогает понять организацию кода и фокусироваться на ключевых аспектах разработки.

При разработке графической составляющей дипломных п*роектов можно использовать различные виды диаграмм в зависимости от целей и задач проекта. Некоторые из распространенных диаграмм включают в себя диаграмму классов, последовательности, вариантов использования, деятельности и сущность-связь.

Диаграмма классов часто используется для проектирования и моделирования систем, основанных на объектно-ориентированной парадигме, таких как приложения, разрабатываемые на языках программирования Java, C++ и C#. Она позволяет разработчикам лучше понять архитектуру системы, определить отношения между классами, а также идентифицировать повторно используемые компоненты.

Диаграмма классов особенно полезна при разработке средних и крупных приложений, где необходимо представить множество классов и связей между ними. Она подходит для веб-приложений, мобильных, клиент-серверных приложения и других, где объектно-ориентированный подход используется для организации кода и взаимодействия между компонентами. Примером использования диаграммы классов является разработка системы управления задачами для команды разработчиков. В этом случае, диаграмма классов будет полезна для описания структуры и взаимосвязей между классами, которые составляют основу системы.

Диаграмма последовательности предназначена для моделирования взаимодействия объектов системы во времени или обмена сообщениями между ними в рамках определенного сценария или функциональности. Данная диаграмма широко используется при анализе и проектировании систем, где важно понять порядок выполнения операций и взаимодействия между объектами. Также она полезна при разработке приложений с сложной логикой или систем, где важно понять последовательность событий и взаимодействий между компонентами.

Диаграмма последовательности подходит для различных типов приложений, включая веб-сервисы, распределенные системы, многопоточные приложения и другие, где важна взаимосвязь между объектами и порядок выполнения операций. Например, диаграмма последовательности полезна при разработке системы электронной коммерции для моделирования последовательности событий при оформлении заказа и обработке платежей. Она помогает понять взаимодействие между объектами, такими как «Пользователь», «Корзина», «Платежный сервис» и «База данных». Это позволяет выявить проблемы, оптимизировать процессы и обеспечить плавное выполнение операций для удобства пользователей.

Диаграмма вариантов использования, или диаграмма прецедентов, представляет собой графическое изображение функциональности системы в виде сценариев использования. Данная диаграмма широко используется на этапе анализа и проектирования системы, чтобы определить и описать функциональные требования и сценарии использования.

Диаграмма вариантов использования особенно полезна при разработке приложений с акцентом на пользовательский интерфейс и функциональность. Она подходит для приложений, где важно определить и моделировать сценарии использования пользователей: веб-приложения, мобильные, системы управления базами данных и другие. Примером использования диаграммы вариантов использования может быть разработка мобильного приложения для доставки еды. В этом случае, диаграмма вариантов использования будет полезна для моделирования сценариев, таких как «Заказ еды», «Отслеживание заказа» и «Оплата». Она позволит разработчикам лучше понять, как пользователи будут взаимодействовать с приложением, какие функции оно должно предоставлять и как система будет отвечать на запросы пользователей.

Диаграмма деятельности представляет собой графическое представление последовательности действий или процессов в системе. Она позволяет моделировать, как система или компоненты взаимодействуют друг с другом и как происходит поток управления между ними. Данная диаграмма подходит для приложений и систем, где важно моделирование последовательностей действий, бизнес-процессов, алгоритмов и потоков управления: бизнес-приложения, системы управления рабочими процессами, системы автоматизации и другие.

Пример использования диаграммы деятельности может быть при разработке приложения для управления проектами. В этом случае, диаграмма деятельности будет полезна для моделирования бизнес-процессов, таких как «Создание проекта», «Назначение задач», «Отслеживание прогресса» и «Завершение проекта». Она поможет визуализировать и анализировать бизнес-процессы, определить потоки данных и контрольные точки в системе управления проектами.

Диаграмма сущность-связь (ER-диаграмма) представляет собой графическое изображение сущностей, атрибутов и связей между ними в базе данных. Диаграмма сущность-связь широко используется на этапе проектирования баз данных для определения структуры и связей между таблицами и сущностями. Она позволяет разработчикам лучше понимать, как данные связаны между собой и как они будут храниться в базе данных. Диаграмма сущность-связь наиболее подходит для разработки приложений, которые работают с базами данных, таких как системы управления контентом, электронные магазины, системы управления проектами и другие. Она также может быть использована для проектирования архитектуры системы в целом, а не только базы данных [1].

Графическое представление информации через диаграммы играет важную роль в разработке дипломных проектов, особенно в области приложений. Выбор подходящих графических компонентов, таких как диаграмма классов, последовательности, вариантов использования, деятельности и сущность-связь, помогает ясно и наглядно передавать идеи и функциональность проекта.

Студентам рекомендуется внимательно выбирать и применять соответствующие графические компоненты, учитывая особенности каждой диаграммы и их применение. Грамотное использование графики делает проект доступным и привлекательным для пользователей, а также помогает студентам продемонстрировать свои навыки и профессионализм [2].

Для помощи учащимся в проектировании графической части был разработан репозиторий на GitHub с информацией по UML, главное окно которого представлено на рисунке 1.

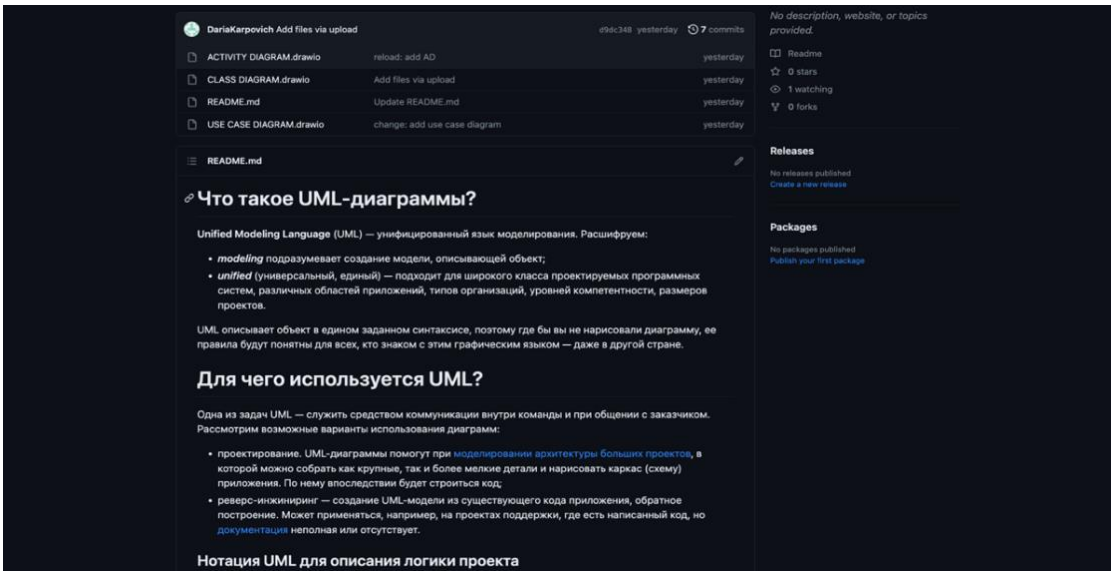

Рисунок 1 – Главное окно репозитория

В репозитории приведены шаблоны наиболее часто разрабатываемых диаграмм, которые доступны редактирования в Draw.io. Также к диаграммам приведено описание, правила составления и пример, окно представлено на рисунке 2.

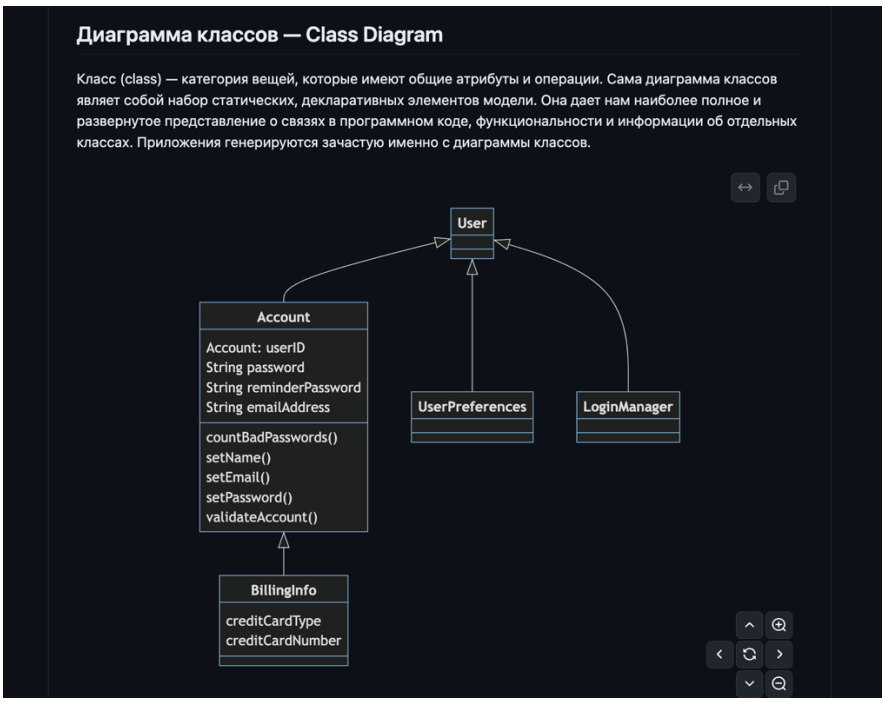

Рисунок 2 – Пример представления диаграммы классов

В заключении, с помощью графических нотаций можно визуализировать систему, объединить все компоненты в единую структуру. С помощью диаграмм можно визуализировать систему с различных точек зрения. Каждая диаграмма служит для определенной цели, от которой зависит эффективность визуализации данных. Выбор диаграммы зависит от того, в каком виде необходимо представить данные.

Список использованных источников

[1] Хританков А.С. Проектирование на UML / А.С. Хританков, В.А. Полежаев, А.И. Андрианов – 3-е изд. – Берлин, 2018. – 241 с.

[2] Гома Х. UML. Проектирование систем реального времени, параллельных и распределенных приложений: Пер. с англ. – Москва: ДМК Пресс, 2016. – 700 с.