А.М. Соловчук
Беларусь, Брест, БрГУ имени А.С. Пушкина

АРХИТЕКТУРА И МЕХАНИЗМЫ РАБОТЫ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ НА ПРИМЕРЕ ВЕБ-ПРИЛОЖЕНИЯ

Функциональные модули системы реализуются с использованием шаблона MVC (Модель – Представление – Контроллер). Это схема использования нескольких шаблонов проектирования, с помощью которых модель данных приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента таким образом, чтобы модификация одного из компонентов оказывала минимальное воздействие на остальные [1].

Таким образом каждый модуль системы разделяется на подмодули, отвечающие за определенную выделенную функциональность в рамках модуля, а каждый функциональный подмодуль разделяется на три компонента.

Модель (Model) содержит или представляет данные, с которыми работают пользователи. Это могут быть простые модели представления, которые только представляют данные, передаваемые от контроллера представлению, или доменные модели, которые содержат данные домена, а также операции, преобразования и правила работы с этими данными. Здесь реализуются классы сущностей, используемых подмодулем, фабрики по созданию объектов, классов, сущностей и методы, отвечающие за хранение и доступ к объектам сущностей.

Представления (View) используются для того, чтобы обработать некоторые части модели в качестве пользовательского интерфейса, т.е. здесь реализуется пользовательское представление классов сущностей (пользовательский интерфейс, web-сервисы и т.д.).

Контроллеры (Controller) обрабатывают входящие запросы, выполняют операции для модели и выбирают представления для показа пользователю. В контроллере реализуется логика работы с сущностями подмодуля.

В MVC модель является независимым компонентом – любые изменения контроллера или представления не затрагивают модель [2]. Контроллер и представление являются относительно независимыми компонентами, и нередко их можно изменять независимо друг от друга.

Благодаря этому реализуется концепция разделение ответственности, в связи с чем легче построить работу над отдельными компонентами. Кроме того, вследствие этого приложение обладает лучшей тестируемостью. И если нам, допустим, важна визуальная часть или фронтэнд, то мы можем тестировать представление независимо от контроллера. Либо мы можем сосредоточиться на бэкэнде и тестировать контроллер [3].

Также популярным инструментом объектно ориентированного подхода к проектированию является язык UML – унифицированный язык моделирования, который предназначен для визуализации и документирования объектно ориентированных систем с ориентацией их на разработку программного обеспечения.

UML призван поддерживать процесс моделирования проектируемой системы на основе объектно ориентированного подхода, организовывать взаимосвязь концептуальных и программных понятий, отражать проблемы масштабирования сложных систем. Данный язык включает в себя систему различных диаграмм, на основании которых может быть

построено представление о проектируемой системе. В частности диаграммы прецедентов (диаграммы вариантов использования, use case diagrams) – это обобщенная модель функционирования системы в окружающей среде [4].

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Тархов, С. В. Математическая модель распределения учебной нагрузки между преподавателями кафедры / С. В. Тархов, С. Н. Султанова // Информационные технологии моделирования и управления. № 5. – Воронеж : Науч. кн., 2005. – С. 676–681.

2. Фримен, А. ASP.NET MVC 4 с примерами на C# 5.0 для профессионалов / А. Фримен, 2013. – 738 с.

3. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5 / Э. Троелсен. – 6-е изд. – М. : И. Д. Вильямс, 2013. –1312 с.

4. Бланшет, Ж. Qt 4: Программирование GUI на C++ / Ж. Бланшет, М. М. Саммерфилд : Кудиц-пресс, 2008. – 736 с.