Материалы Российской НТК "Информатика и проблемы телекоммуникаций", Новосибирск. 2008. – Т.1. – С.81-84.

АРХИТЕКТУРНОЕ РЕШЕНИЕ ДЛЯ ПРОГРАММНОЙ СИСТЕМЫ МНОГОМЕРНОГО СТАТИСТИЧЕСКОГО АНАЛИЗА

Танасейчук А.В., Лемешко Б.Ю.
НГТУ, Новосибирск,
Email: awtan@yandex.ru

К настоящему моменту сложность прикладного программного обеспечения возросла на столько, что стала очевидной необходимость формализованного, системного подхода к его разработке. В данной статье предлагается общее архитектурное решение для организации программной системы исследования закономерностей многомерного статистического анализа.

Специфика задач, для решения которых разрабатывается система, накладывает определенные требования, которые должны быть учтены на этапе проектирования. Основные задачи, решаемые системой: моделирование случайных величин, распределенных по различным многомерным законам, моделирование распределений статистик критериев многомерного статистического анализа, автоматическое вычисление мощности критериев, проверка статистических гипотез. Все эти задачи требуют от пользователя предварительной настройки системы и не требуют вмешательства во время выполнения. В процессе развития системы предполагается добавление новых типов распределений, новых статистик и гипотез.

Таким образом, можно выделить следующие требования и характеристики системы:

1) Расширяемость;
2) Легкость модификации;
3) Надежность, устойчивость к изменениям;
4) Слабая интерактивность пользовательского интерфейса.

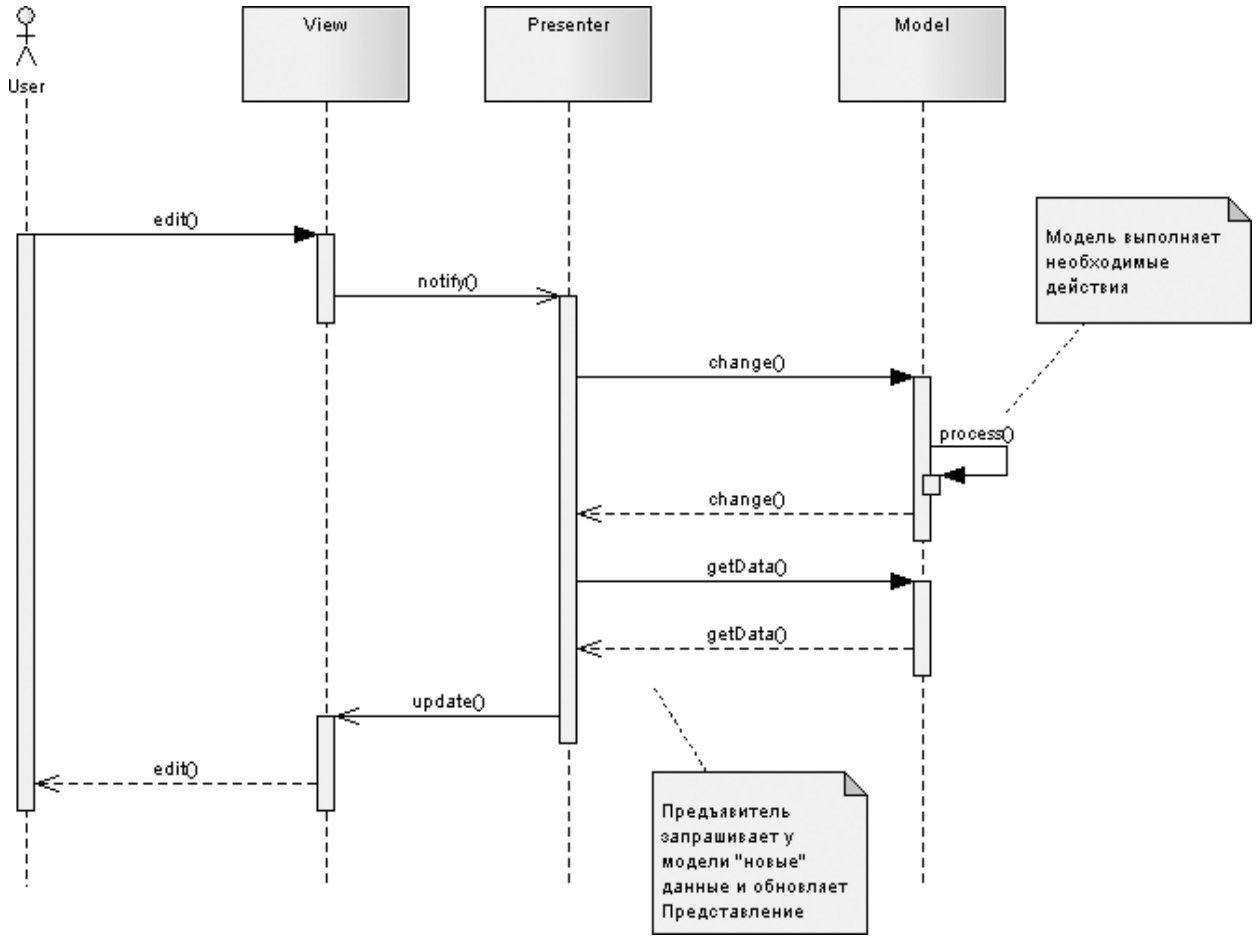
Одним из ключевых событий в развитии системного подхода к проектированию программных систем стал выход в 1995 году книги "Design Patterns: Elements of Reusable Object-Oriented Software" [1], которая ввела в широкое употребление термин «шаблоны проектирования». Одним из самых эффективных решений за всю историю развития методов разработки программных систем явился шаблон проектирования Model-View-Controller [2].

В основу архитектурного решения для системы статистического анализа разумно положить современную модификацию классического архитектурного подхода Model-View-Controller – архитектуру Model-View-Presenter (далее просто MVP) [3, 4]. Обычно выделяют три возможных вариации шаблона MVP: Passive View [5], Supervising Presenter [6] и Presentation Model [7]. Имея в виду тот факт, что наиболее существенная часть нашего приложения это

моделирование случайных величин и распределений различных статистик, можно заключить, что пользовательский интерфейс предполагается достаточно «легким» и простым. Таким образом, для нашей системы, учитывая предполагаемую слабую интерактивность пользовательского интерфейса, наилучшим выбором будет Passive View.

Шаблон MVP разделяет систему на три составляющие: модель (model), предъявитель (presenter), представление (view). Модификация Passive View предполагает следующее устройство системы: модель предоставляет данные и алгоритмы предметной области предъявителю и реагирует на его команды. Представление отображает данные модели, которые ему предоставляет предъявитель и принимает действия пользователя, делегируя их обработку предъявителю. Не допускается никакого прямого взаимодействия представления и модели. Предъявитель содержит логику работы приложения, обновляет представление и отдает модели команды на выполнение необходимых действий.

Диаграмма последовательностей [8] обработки некоторого абстрактного действия пользователя в модели MVP представлена на следующем рисунке:



В целом, такое решение дает несколько преимуществ: уменьшает связность системы и эффективно отделяет представление данных от логики – что существенно облегчает расширение и модификацию реализуемой системы. На основе такого решения достаточно легко можно организовать как отдельное

приложение, так и модуль, встраиваемый в более крупную систему (например, построенную на основе «плагинов» [9]).

Существует еще одна проблема, с которой обычно сталкивается разработчик, проектирующий нетривиальную программную систему. Это проблема организации приложения, состоящего из нескольких оконных форм и требующего определенной взаимосвязи между этими формами. На сегодняшний день не существует единого общепринятого решения такой проблемы, а стандартные шаблоны, прежде всего, описывают взаимодействия внутри триады модель-вид-контроллер или модель-вид-предъявитель. Тем не менее, существуют несколько возможных подходов к решению, применимых в тех или иных конкретных случаях.

Для нашей задачи в качестве решения было выбрано построение иерархии предъявителей с соответствующими представлениями. Мы выделяем в отдельную сущность так называемый предъявитель уровня приложения, который содержит в себе самую общую логику – набор возможных состояний приложения, текущее состояние и т.п. Для каждой логически независимой формы, созданной под решение конкретной задачи (это может быть, например, окно для настройки процедуры моделирования многомерных распределений или выбора критерия для вычислений мощности), создается пара предъявитель-представление, отвечающая за решение этой задачи. Управляет временем жизни объектов этой пары предъявитель верхнего уровня, он же предоставляет предъявителю нижнего уровня ссылку на модель, которая существует в единственном экземпляре.

Литература

1. Gamma E., Helm R., Johnson R., Vlissides J. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison Wesley, 1995, 416 p.
2. Reenskaug T. *Models - Views - Controllers*. Technical note, Xerox PARC, December 1979. (http://heim.ifi.uio.no/~trygver/mvc/index.html)
3. Boodhoo J.-P. *Microsoft Journal: Model-View-Presenter*. (http://msdn.microsoft.com/msdnmag/issues/06/08/DesignPatterns/default.aspx)
4. Бодягин И. *Model-View-Controller в .NET.* RSDN Magazine #2, 2006. (http://rsdn.ru/article/patterns/ModelViewPresenter.xml)
5. Fowler M. *Passive View*. (http://www.martinfowler.com/eaaDev/PassiveScreen.html)
6. Fowler M. *Supervising Controller*. (http://martinfowler.com/eaaDev/SupervisingPresenter.html)
7. Fowler M. *Presentation Model*. (http://martinfowler.com/eaaDev/PresentationModel.html)
8. Jacobson I., Booch G., Rumbaugh J. *The Unified Software Development Process.* Addison-Wesley Professional, 1999, 512 p.
9. Fowler M., Rice D., Foemmel M., Hieatt E., Mee R., Stafford R. *Patterns of Enterprise Application Architecture.* Addison-Wesley, 2002, 560 p.