УДК 004.42

Создание системы контроля пространственно-временного состояния техногенных объектов с использованием шаблона проектирования Model-View-Presenter

А. Р. Аргинбаев[1], П. Ю. Бугаков[1]*

[1] Сибирский государственный университет геосистем и технологий, г. Новосибирск, Российская Федерация

* e-mail: arthur.arginbaev@gmail.com

Аннотация. Статья посвящена проблеме разработке программного обеспечения с использованием архитектурных шаблонов проектирования, предназначенных для отделения данных от их представления и управляющей логики приложения. Цель работы заключается в создании информационной системы контроля пространственно-временного состояния техногенных объектов с использованием архитектурного шаблона Model-View-Presenter (MVP). Разработка выполнялась на языке программирования C# в интегрированной среде разработки Microsoft Visual Studio на платформе Windows Forms (.NET Framework) с применением встраиваемой СУБД SQLite. В результате выполненной работы была изучена базовая структура и связи классов MVP, сформулированы требования к функциональным возможностям, а также разработан прототип информационной системы контроля пространственно-временного состояния техногенных объектов. На основе полученных результатов дана оценка достоинств и недостатков применения шаблона проектирования MVP при создании программного обеспечения, обладающего подобными функциональными характеристиками.

Ключевые слова: программное обеспечение, шаблон проектирования, модель, представление, представитель, Model-View-Presenter

Creation of a system for monitoring the spatio-temporal state of a man-made object using the Model-View-Presenter design pattern

A. R. Arginbaev[1], P. Yu. Bugakov[1]*

[1] Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation

* e-mail: arthur.arginbaev@gmail.com

Abstract. The article is devoted to the problem of software development using architectural design patterns designed to separate data from their presentation and control logic of the application. The purpose of the work is to create an information system for monitoring the spatial and temporal state of man-made objects using the architectural template Model-View-Presenter (MVP). The development was carried out in the C# programming language in the Microsoft Visual Studio integrated development environment on the Windows Forms platform (.NET Framework) using the embedded SQLite DBMS. As a result of the work performed, the basic structure and connections of the MVP classes were studied, functional requirements were formulated, and a prototype of an information system for monitoring the spatial and temporal state of man-made objects was developed. Based on the results obtained, an assessment of the advantages and disadvantages of using the MVP design pattern when creating software with similar functional characteristics is given.

Keywords: software, design pattern, model, representation, representative, Model-View-Presenter

Введение

Мониторинг пространственно-временного состояния техногенных объектов вляется одной из важнейших задач современной геодезии. Он представляет собой комплекс измерительных и описательных мероприятий по выявлению деформаций, определению их рода и причин их возникновения. Во избежание чрезвычайных ситуаций, специалисты в области геодезии ведут постоянное наблюдение за пространственно-временным состоянием зданий, сооружений или их конструктивных элементов. Для оперативного сбора и обработки результатов геодезических измерений, выполненных в процессе наблюдения за объектом, используется специальное программное обеспечение, реализующее алгоритмы выявления и локализации деформационных процессов в его конструкции [1–5].

Согласно учебному плану кафедры прикладной информатики и информационных систем СГУГиТ в 4 семестре обучающиеся по направлению 09.03.02 Информационные системы и технологии выполняют курсовую работу по дисциплине «Моделирование систем». Основной целью курсовой работы является разработка информационной системы анализа пространственно-временного состояния техногенного объекта и выполнение вычислительного эксперимента на примере персонального варианта исходных данных. Поставленная задача помогает обучающимся приобрести базовые навыки создания программного обеспечения, а также формирует представление о проблемно-ориентированном подходе в проектной деятельности.

Как правило, разработка информационной системы выполняется на языке программирования C# в интегрированной среде Microsoft Visual Studio на платформе Windows Forms (.NET Framework) с применением встраиваемой СУБД SQLite [6–10]. При написании программы в рамках курсовой работы обучающиеся используют архитектурный паттерн Model-View-Controller, который заключается в разделении данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер. Однако написание программы с применением данного шаблона может вызывать неудобства, связанные со спецификой платформы разработки Windows Forms, из-за чего возникает необходимость выбора иного паттерна проектирования.

Цель данной работы заключается в создании информационной системы контроля пространственно-временного состояния техногенного объекта с использованием архитектурного шаблона Model-View-Presenter.

Для достижения поставленной цели необходимо выполнить следующие задачи:

– определить назначение и функционал разрабатываемой информационной системы;

– изучить базовую структуру и связи классов MVP;

– выполнить разработку прототипа информационной системы с использованием архитектурного шаблона проектирования MPV;

– оценить преимущества и недостатки используемого шаблона проектирования.

В качестве исходных данных для работы программы используется таблица измеренных высот геодезических марок, закрепленных на исследуемом объекте, значение допустимой точности выполненных измерений, а также коэффициент экспоненциального сглаживания, используемый при прогнозировании состояния техногенного объекта. Высоты всех марок измеряются многократно в соответствии с графиком наблюдений. Момент измерения высот марок называется эпохой и представлен в таблице отдельной строкой, включающей поля по количеству марок [11–15].

Результаты работы программного обеспечения должны быть представлены набором таблиц фазовых координат и соответствующих им графиков, показывающих изменения состояния исследуемого техногенного объекта и его структурных блоков во времени. При выходе состояния за пределы допустимого диапазона программа должна проинформировать об этом пользователя, указать моменты времени, когда были обнаружены критические деформации, а также предоставить возможность дальнейшего более детального исследования объекта.

Каждый цикл измерений характеризуется точкой метрического пространства, образованной координатами точек геодезической системы:

$$H_n = \{h_{n,1}, h_{n,2}, ..., h_{n,m}\}, \quad (1)$$

где h – высота марки; n – номер измерения; m – номер марки.

Положение объекта в двумерном пространстве определяется по фазовым координатам μ (длина вектора, образуемого двумя точками) и α (угол между начальным и текущим векторами).

Длина вектора вычисляется по формуле:

$$\mu_n = \left|\overline{H_n}\right| = \sqrt{h_{n,1} + h_{n,2} + ... + h_{n,m}} \quad (2)$$

Угол между векторами вычисляется по формуле:

$$\alpha_n = \arccos\left(\frac{\overline{H_0} \cdot \overline{H_n}}{\mu_0 \cdot \mu_n}\right) \quad (3)$$

На основе значений, рассчитаных для каждого измерения, вычисляются прогнозные значения методом экспоненциального сглаживания:

$$S_t = A \cdot y_t + (1 - A) \cdot S_{t-1}, \quad (4)$$

где S_t – прогнозное значение, A – коэффициент экспоненциального сглаживания от 0 до 1, y_t – реальное значение для расчета прогнозной величины, S_{t-1} –прогнозное значение за предыдущий период времени.

Для контроля и проверки системы на устойчивость используются предельные значения фазовых координат, рассчитываемые по формуле:

$$H^{\pm} = H \pm \varepsilon \Rightarrow \mu^{\pm}, \alpha^{\pm}, \quad (5)$$

где ε – точность измерения каждой отметки.

В случае выхода фазовых значений за границы предельных система перестает быть устойчивой.

При разработке функционально насыщенного программного обеспечения выбор шаблона проектирования представляет собой одну из ключевых задач, решение которой должно начинаться на начальном этапе архитектурного проектирования.

Архитектурные шаблоны определяют структуру информационной системы, на которую в дальнейшем наращивается актуальный и востребованный функционал. Подходы к ее организации диктуются условиями эксплуатации конкретного программного продукта [16–20].

В разрабатываемом приложении чаще всего изменяется пользовательский интерфейс. Пользователю нужно видеть данные в различном представлении и все представления должны отражать текущее состояние данных.

Следовательно, для комфортной разработки необходимо отделить функциональность пользовательского интерфейса от функциональности приложения и при этом обеспечить быстрый отклик на действия пользователя и изменения в базовых данных приложения, создавать, поддерживать и координировать несколько представлений пользовательского интерфейса при изменении базовых данных.

На первый взгляд, адекватным решением для подобных условий является самый распространненый шаблон для разработки пользовательского интерфейса «Model – View – Controller» (MVC). Но в оригинальном паттерне именно Контроллер должен реагировать на внешние события, однако в Windows Forms все обработчики уже встроены в Представление.

Данный факт вынуждает либо адаптировать сгенерированный дизайнером код под вынесение обработчика событий во внешний Контроллер, либо интегрировать Контроллер в Представление. Оба варианта обладают существенными недостатками, усложняющими процесс разработки.

Можно сделать вывод, что абстракции «Модель», «Представление» и «Контроллер» не подходят для сред типа Windows Forms. Таким образом, требуется модификация паттерна, которая с учетом упомянутых недостатков позволяла бы:

– эффективно отделять модель от ее представлений;

– без ограничений пользоваться дизайнером форм и имеющимися библиотеками;

– тестировать логику Контроллера независимо от Представления и сводила бы логику Представления к минимуму;

– избегать лишних обращений к Модели.

Одним из возможных решений, отвечающим всем вышеприведенным требованиям, является паттерн Model-View-Presenter (MVP).

Model-View-Presenter – шаблон, разработанный для облегчения модульного тестирования и отделения логики от отображения. Для реализации MVP используют три основных класса (рис. 1):

1) Модель (англ. Model) – предоставляет данные и реагирует на команды, изменяя свое состояние. Хранит набор полей и методов для реализации бизнес-логики приложения, а также набор событий, оповещающий Представителя об изменении тех или иных значений;

2) Представление (англ. View) – отображает данные и перенаправляет события от пользователя в Представителя. Хранит экземпляр Представителя, реализует соответствующий интерфейс;

3) Представитель (англ. Presenter) – реализует взаимодействие между Моделью и Представлением, при необходимости запрашивает данные и передает их для отображения. Хранит ссылку на реализацию Представления, хранит экземпляр Модели, реализует соответствующий интерфейс.

Взаимодействуя с элементами управления, пользователь инициирует события (рис. 2), которые обрабатываются Представлением и перенаправляются Представителю. Представитель их обрабатывает, вызывая изменение Модели, обрабатывает события ее изменения и вызывает обновление Представления.

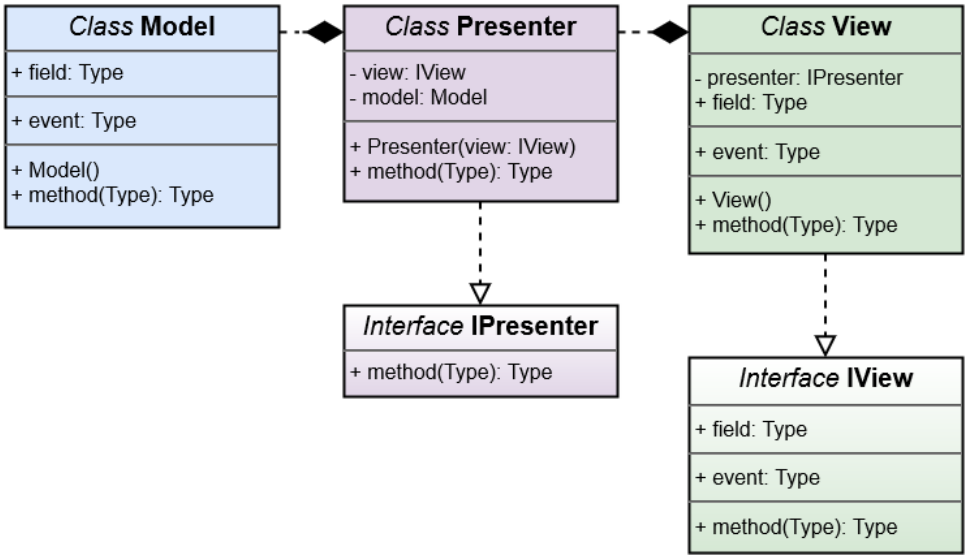

Рис. 1. Структура и связи классов MVP

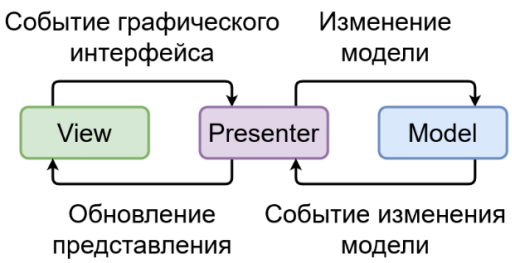

Рис. 2. Обработка событий

Результаты

Разработанное программное обеспечение позволяет выполнить вычисление длины вектора, угла между текущим и начальным вектором, прогнозные и предельные значения, а также состояние системы для каждого цикла измерений. Пользовательский интерфейс построен в соответствии со спецификацией Multiple-document interface (MDI), что обеспечивает пользователю возможность одновременной работы с несколькими окнами.

При запуске программы загружается родительское окно, все остальные окна являются дочерними и находятся в ее рабочем пространстве. Это позволяет пользователю самостоятельно настраивать интерфейс, вызывая только необходимые для работы окна.

Возможность загрузки исходных данных реализована через вкладку «База данных», подпункт «Подключить». После успешного подключения к базе данных (рис. 3), пользователь, вызвав соответствующие окна, может изменить настройки и выбранную таблицу марок, а также изучить схему объекта.

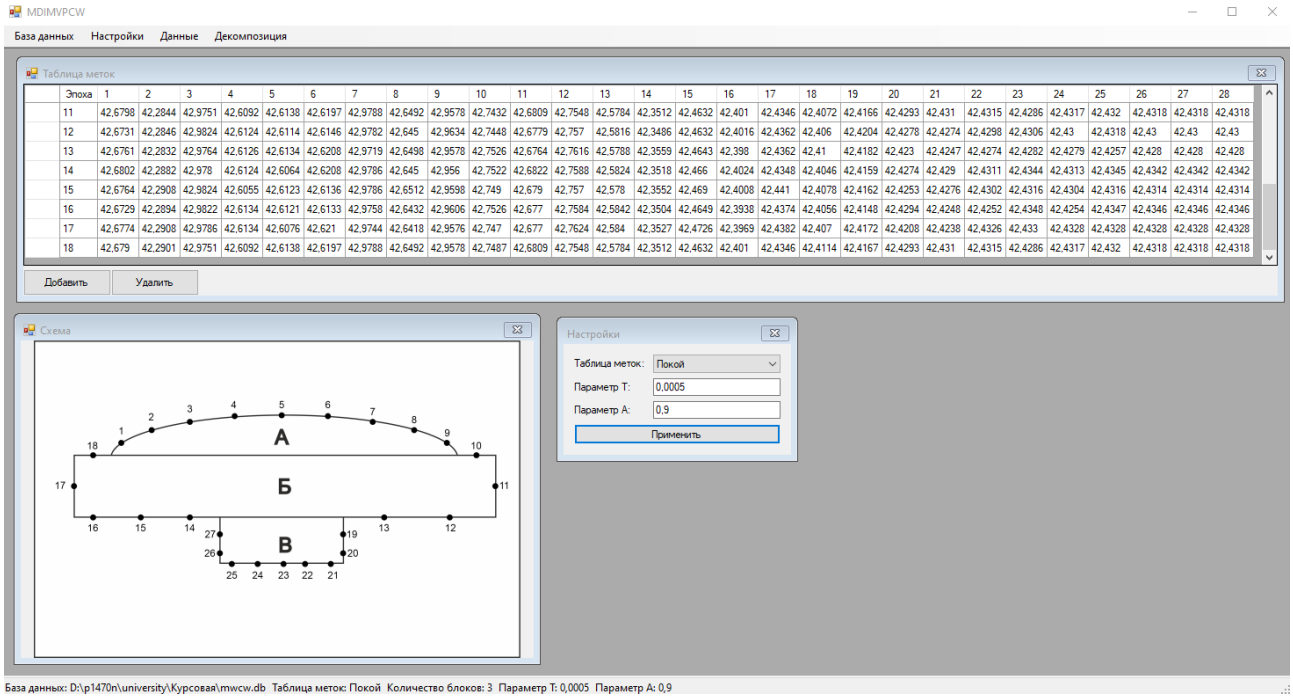

Рис. 3. Загруженные исходные данные

Затем пользователь может приступать к декомпозиции объекта по имеющимся данным (рис. 4).

Для проведения расчетов I уровня необходимо выбрать «Первый уровень» во вкладке «Декомпозиция». После этого появится форма №1, отображающая изменение состояния объекта, его прогнозных и предельных значений в фазовом пространстве.

Для проведения расчетов II уровня необходимо выбрать «Второй уровень» во вкладке «Декомпозиция». После этого появится форма №2, для распределения

марок по блокам, после которого появится форма №3, отображающая изменение состояния блоков объекта, их прогнозных и предельных значений в фазовом пространстве.

Применение разработанной программы позволяет оперативно принимать решения для предотвращения аварийных ситуаций на техногенных объектах, которые могут причинить материальный ущерб или угрожать жизни и здоровью людей.

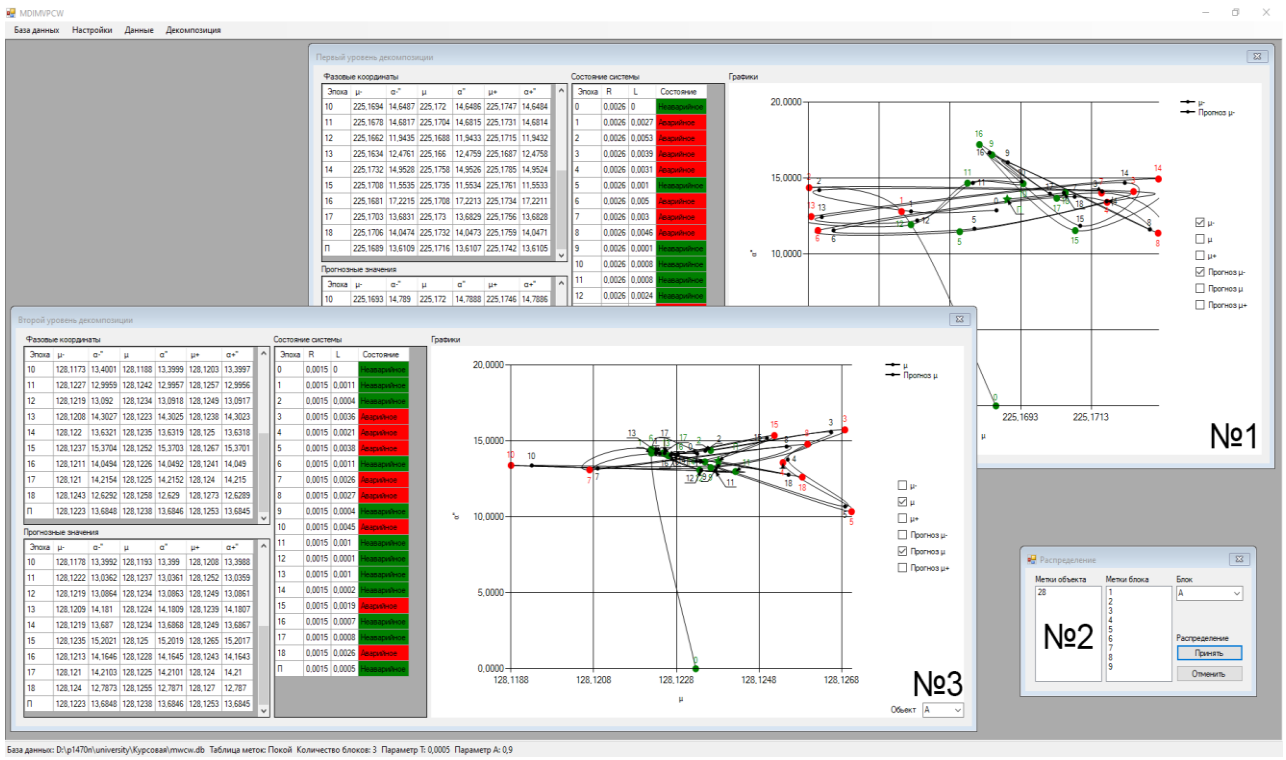

Рис. 4. Результат расчета для определения состояния техногенного объекта

Анализируя опыт применения архитектурного шаблона MPV при создании прикладного программного обеспечения, можно выделить его достоинства и недостатки. К преимуществам можно отнести автоматическое обновление Представления при изменении Модели, а также возможность реализации нескольких вариантов Представлений (и Представителей) без изменения Модели. К недостатку шаблона проектирования MVP можно отнести усложненную структуру классов, строгие требования к реализации и возросшие требования к уровню квалификации разработчика.

При разработке информационной системы контроля пространственно-временного состояния техногенных объектов применение архитектурного шаблона MVP позволило эффективно отделить модель данных от ее представления в пользовательском интерфейсе.

Таким образом, дальнейшее сопровожние разработанной информационной системы будет происходить проще и с меньшими временными затратами по сравнению с программным обеспечением, созданным на основе паттерна MVC.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Бугакова Т. Ю. Моделирование изменения пространственно-временного состояния инженерных сооружений и природных объектов по геодезическим данным: статья в журнале Вестник СГГА. – Вып. 1 (29). – Новосибирск : СГУГиТ, 2015. – С. 34–42.

2. Бугакова Т. Ю., Борисов Д. А. Модель определения пространственно-временного состояния техногенных систем методами по данным геодезических наблюдений: статья в журнале Интерэкспо ГЕО-Сибирь-2015. – Новосибирск : СГУГиТ, 2015. – С. 56–62.

3. Шеховцов Г. А., Шеховцова Р. П. Современные геодезические методы определения деформаций инженерных сооружений: монография. – Нижний Новгород: Нижегород. гос. архит.-строит. ун-т., 2014. – 256 с.

4. Вовк И. Г. Системно-целевой подход в прикладной геоинформатике: статья в журнале Вестник СГГА. – Вып. 2 (18). – Новосибирск : СГУГиТ, 2012. – с. 115–124.

5. Вовк И. Г. Математическое моделирование в прикладной геоинформатике: статья в журнале Вестник СГГА. – Вып. 1 (17). – Новосибирск : СГУГиТ, 2012. – с. 94–103.

6. Прайс М. C# 9 и .NET 5. Разработка и оптимизация. – СПб : Питер, 2022. – 832 с. – ISBN 978-5-4461-2921-8. – Текст : непосредственный.

7. Гриффитс И. Программируем на C# 8.0. Разработка приложений. – СПб : Прогресс книга, 2021. – 944 с. – ISBN 978-5-4461-1638-6. – Текст : непосредственный.

8. Евдокимов П. В. C#. Практическое руководство. – СПб : Наука и техника, 2022. – 416 с. – ISBN 978-5-94387-300-3. – Текст : непосредственный.

9. Шилдт Г. C# 4.0: полное руководство. – М. : ООО "И.Д. Вильямс", 2011. – 1056 с. –ISBN 978-5-8459-1684-6. – Текст : непосредственный.

10. Петцольд Ч. Программирование для Microsoft Windows на C#. В 2-х томах. Том 1. – М. : Издательско-торговый дом «Русская Редакция», 2002. – 576 с. – ISBN 5-7502-0210-0. – Текст : непосредственный.

11. Бугакова Т. Ю., Вовк И. Г. Математическое моделирование пространственно-временного состояния систем: Материалы V Всероссийской научно-технической конференции «Актуальные вопросы строительства». – Новосибирск: НГАСУ (Сибстрин), 2012. – Т. 2. – с. 100– 105.

12. Бугакова Т. Ю. Математическое моделирование пространственно-временного состояния систем по геометрическим свойствам и оценка техногенного риска методом экспоненциального сглаживания: статья в журнале Вестник СГГА. – Вып. 4 (20). – Новосибирск : СГГА, 2012. – с. 47– 58.

13. Вовк И. Г., Бугакова Т. Ю. Математическое моделирование пространственно-временного состояния систем по геометрическим свойствам: статья в журнале Интерэкспо ГЕО-Сибирь-2012. – Том: 3. – Новосибирск : СГГА, 2012. – с. 26–31.

14. Бугакова Т. Ю. К вопросу оценки риска геотехнических систем по геодезическим данным: статья в журнале Гео-Сибирь. – Том: 1, Номер: 1. – Новосибирск : СГГА, 2011. – с. 151–157.

15. Вовк И. Г., Бугакова Т. Ю. Декомпозиция и агрегирование систем при реализации системно-целевого подхода: статья в журнале Гео-Сибирь. – Том: 1, Номер: 2. – Новосибирск : СГГА, 2010. – с. 21–24.

16. Фримен Эр., Фримен Эл., Сьерра К., Бейтс Б. Паттерны проектирования. – СПб : Питер, 2011. – 656 с. – ISBN 978-5-459-00435-9. – Текст : непосредственный.

17. Мартин Р., Мартин М. Принципы, паттерны и методики гибкой разработки на языке C#. – СПб : Символ-Плюс, 2011. – 768 с. – ISBN 978-5-93286-197-4. – Текст : непосредственный.

18. Фаулер М. Архитектура корпоративных программных приложений. – М. : «Вильямс», 2007. – 544 с. – ISBN 5-8459-0579-6. – Текст : непосредственный.

19. Ларман К. Применение UML и шаблонов проектирования. 2-е издание. – М.: Издательский дом «Вильямс», 2004. – 624 с. – ISBN 5-8459-0250-9. – Текст : непосредственный.

20. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. – СПб : Питер, 2001. – 368 с. – ISBN 5-272-00355- 1. – Текст : непосредственный.