МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ПАТЕРНЫ ПРОЕКТИРОВАНИЯ

Методические указания к лабораторным занятиям

для направления подготовки:

09.03.04 Программная инженерия

Ставрополь, 2026

Н 63 **Патерны проектирования:** методические указания к лабораторным занятиям для студентов направления 09.03.04 Программная инженерия / Николаев Е.И. – Ставрополь: Изд-во СКФУ, 2026. – 27 с.

Автор:

канд. техн. наук, доцент **Е.И. Николаев**

ОГЛАВЛЕНИЕ

1. Методика и порядок выполнения лабораторных работ 4
2. Краткое теоретическое обоснование..... 7
Лабораторная работа 1. Порождающие паттерны: одиночка, строитель 11
Лабораторная работа 2. Порождающие паттерны: Абстрактная фабрика, Фабричный метод, Прототип..... 12
Лабораторная работа 3. Структурные паттерны: адаптер, мост 13
Лабораторная работа 4. Структурные паттерны: компоновщик, декоратор..... 14
Лабораторная работа 5. Структурные паттерны: Фасад, приспособленец 15
Лабораторная работа 6. Структурные паттерны: Заместитель 16
Лабораторная работа 7. Поведенческие паттерны: Команда, Итератор 17
Лабораторная работа 8. Поведенческие паттерны: Цепочка обязанностей 18
Лабораторная работа 9. Поведенческие паттерны: Посредник, снимок..... 19
Лабораторная работа 10. Поведенческие паттерны: наблюдатель, состояние..... 20
Лабораторная работа 11. Поведенческие паттерны: Стратегия, шаблонный метод..... 21
Лабораторная работа 12. Архитектурные паттерны..... 22
Лабораторная работа 13. Паттерны микросервисов и распределённых систем 23
Лабораторная работа 14. Системные паттерны и работа с данными 24
Лабораторная работа 15. Идиоматические паттерны..... 25
Список литературы 26

1. МЕТОДИКА И ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

1.1. Решение прикладных задач с использованием шаблонов проектирования

Каждая лабораторная предполагает написание программного проекта на языках программирования высокого уровня с обязательным применением одного или нескольких шаблонов проектирования.

1.2. Порядок защиты

Студент демонстрирует готовый, работающий проект приложения, предоставляет диаграмму классов (и прочие визуальные материалы в нотации UML), демонстрирующую факт реализации требуемых шаблонов проектирования в рамках лабораторной работы. Также предоставляется отчет по лабораторной работе (*.docx, *.pdf). Рекомендуется сформировать репозитарий GitHub с исходными кодами лабораторных работ.

1.3. Форма отчета

Студент предоставляет отчет в электронном виде. Отчет должен содержать следующие элементы:

1. Титульный лист.
2. Исходный код (в случае объемного материала – ссылка на GitHub).
3. Диаграммы классов для подтверждения реализации паттернов проектирования.

1.4. Варианты заданий

Лабораторные работы заключаются в решении прикладной задачи на языке высокого уровня с применением шаблонов проектирования. Примерный перечень прикладных задач (студент может самостоятельно предложить и обосновать свою прикладную задачу):

1. Разработка приложения «Интерактивная карта»
2. Разработка приложения «Менеджер фотографий»
3. Разработка приложения «Траектория движения»
4. Разработка приложения «Игра «Собиратель монет»
5. Разработка приложения «Свертка изображения»
6. Разработка приложения «Матричный калькулятор»
7. Разработка приложения «Аквалайзер»
8. Разработка приложения «Графический пароль»
9. Разработка приложения «Игра «Танки»
10. Разработка приложения «Игра «Змейка»
11. Разработка приложения «Расчет стоимости поездки»
12. Разработка приложения «Расчет баллистической траектории»
13. Разработка приложения «Интерактивный анализ набора данных»
14. Разработка приложения «Моделирование популяции»
15. Разработка приложения «Обработка изображения фильтрами»
16. Разработка приложения «Игра «Цветы против грибов»
17. Разработка приложения «Фракталы»
18. Разработка приложения «Искусственный интеллект»
19. Разработка приложения «Филиалы и сотрудники»
20. Разработка приложения «Игра «Ракета»
21. Разработка приложения "Игра «Блиц»
22. Разработка приложения «Поиск текста в документах»
23. Разработка приложения «Анализ матричных данных»
24. Разработка приложения «Игра «Поиск крота»

25. Разработка приложения «Вычисления со скобками»

Вариант студента может меняться при выполнении различных лабораторных работ.

2. КРАТКОЕ ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ

2.1. Классические паттерны проектирования

Существует 23 классических паттерна проектирования (шаблона), разделенных на три основные группы.

1. Порождающие паттерны (Creational). Регулируют способы создания объектов, скрывая логику их инициализации:

Фабричный метод (Factory Method) – создает объекты через специальный метод, а не прямой вызов конструктора.

Абстрактная фабрика (Abstract Factory) – создает семейства связанных объектов без привязки к конкретным классам.

Строитель (Builder) – собирает сложные объекты пошагово.

Прототип (Prototype) – копирует существующие объекты вместо создания с нуля.

Одиночка (Singleton) – гарантирует, что у класса есть только один экземпляр, и дает к нему глобальную точку доступа.

2. Структурные паттерны (Structural). Определяют, как объединять классы и объекты в крупные структуры, сохраняя гибкость.

Адаптер (Adapter) – позволяет объектам с несовместимыми интерфейсами работать вместе.

Мост (Bridge) – разделяет один большой класс на два отдельных дерева («абстракция» и «реализация»).

Компоновщик (Composite) – объединяет объекты в древовидную структуру для работы с ними как с единым целым.

Декоратор (Decorator) – добавляет объектам новые обязанности на лету, оборачивая их в обертки.

Фасад (Facade) – предоставляет простой интерфейс к сложной системе классов или библиотеке.

Легковес / Приспособленец (Flyweight) – экономит память, разделяя общее состояние между множеством мелких объектов.

Заместитель (Proxy) – подменяет реальный объект и контролирует доступ к нему.

3. Поведенческие паттерны (Behavioral). Отвечают за эффективное общение и распределение обязанностей между объектами:

Цепочка обязанностей (Chain of Responsibility) – передает запрос по цепочке потенциальных обработчиков.

Команда (Command) – заворачивает запросы или простые операции в отдельные объекты.

Индикатор / Итератор (Iterator) – последовательно обходит элементы коллекции без раскрытия ее внутреннего устройства.

Посредник (Mediator) – убирает прямые связи между объектами, заставляя их общаться через один объект-посредник.

Снимок (Memento) – сохраняет и восстанавливает внутреннее состояние объекта без нарушения инкапсуляции.

Наблюдатель (Observer) – оповещает подписчиков об изменениях в состоянии другого объекта.

Состояние (State) – позволяет объекту менять поведение при изменении внутреннего состояния.

Стратегия (Strategy) – подменяет алгоритмы на лету внутри объекта.

Шаблонный метод (Template Method) – задает «скелет» алгоритма, перекладывая реализацию шагов на подклассы.

Посетитель (Visitor) – добавляет новую операцию к структуре классов без ее изменения.

2.2. Архитектурные паттерны (Enterprise & UI)

Определяют общую структуру всего приложения или его крупных модулей.

MVC (Model-View-Controller) – разделяет данные (модель), интерфейс (вид) и логику управления (контроллер).

MVP (Model-View-Presenter) – развитие MVC для UI, где Presenter полностью берёт на себя логику синхронизации, делая View «тупым».

MVVM (Model-View-ViewModel) – стандарт для современных UI-фреймворков. Использует двустороннее связывание данных (Data Binding).

Clean Architecture / Hexagonal (Ports and Beans) – изолирует бизнес-логику от внешних деталей (БД, фреймворки, UI) с помощью интерфейсов.

2.3. Паттерны микросервисов и распределённых систем

Решают проблемы масштабирования, сетевых сбоев и управления данными в облаке.

Saga – управляет распределёнными транзакциями между микросервисами через цепочку локальных транзакций и компенсирующих действий в случае сбоя.

CQRS (Command Query Responsibility Segregation) – разделяет операции чтения данных и операции записи на уровне моделей или даже баз данных.

Event Sourcing (Внедрение событий) – сохраняет не текущее состояние объекта, а полную историю всех изменений (событий) в виде лога.

Circuit Breaker (Предохранитель) – предотвращает лавинообразные сбои в сети, временно блокируя запросы к упавшему сервису.

API Gateway (Шлюз API) – единая точка входа для клиентов, которая перенаправляет запросы в нужные микросервисы, а также занимается авторизацией и логированием.

2.4. Системные паттерны и работа с данными (Data Access)

Помогают организовать слой работы с базами данных.

Repository (Репозиторий) – посредник между слоем бизнес-логики и слоем данных, имитирующий коллекцию объектов в памяти.

Data Mapper – слой маппинга, который переносит данные между объектами в коде и таблицами БД, изолируя их друг от друга (основа ORM типа Hibernate/Entity Framework).

Active Record – подход, где объект содержит в себе и данные, и методы работы с БД (сохранить, удалить). Популярен в Ruby on Rails, Django.

Unit of Work (Единица работы) – отслеживает все изменения объектов в рамках одной бизнес-транзакции и сохраняет их в БД одним пакетом.

2.5. Идиоматические паттерны (Concurrency & Async)

Ориентированы на многопоточность и асинхронное программирование.

Producer-Consumer (Производитель-Потребитель) – один поток создаёт задачи и складывает их в очередь, другие потоки забирают их оттуда и обрабатывают.

Thread Pool (Пул потоков) – повторно использует фиксированное количество потоков вместо создания новых на каждую задачу.

Future / Promise – объект-пустышка, который выступает прокси для результата, который ещё не вычислен (появится асинхронно позже).

ЛАБОРАТОРНАЯ РАБОТА 1. ПОРОЖДАЮЩИЕ ПАТТЕРНЫ: ОДИНОЧКА, СТРОИТЕЛЬ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов «Строитель» или «Одиночка».

ЛАБОРАТОРНАЯ РАБОТА 2. ПОРОЖДАЮЩИЕ ПАТТЕРНЫ: АБСТРАКТНАЯ ФАБРИКА, ФАБРИЧНЫЙ МЕТОД, ПРОТОТИП

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием 2 паттернов из списка: «Абстрактная фабрика», «Фабричный метод», «Прототип».

ЛАБОРАТОРНАЯ РАБОТА 3. СТРУКТУРНЫЕ ПАТТЕРНЫ: АДАПТЕР, МОСТ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Адаптер», «Мост».

ЛАБОРАТОРНАЯ РАБОТА 4. СТРУКТУРНЫЕ ПАТТЕРНЫ: КОМПОНОВЩИК, ДЕКОРАТОР

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Компоновщик», «Декоратор».

ЛАБОРАТОРНАЯ РАБОТА 5. СТРУКТУРНЫЕ ПАТТЕРНЫ: ФАСАД, ПРИСПОСОБЛЕНЕЦ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Фасад», «Приспособленец».

ЛАБОРАТОРНАЯ РАБОТА 6. СТРУКТУРНЫЕ ПАТТЕРНЫ: ЗАМЕСТИТЕЛЬ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттерна «Заместитель».

ЛАБОРАТОРНАЯ РАБОТА 7. ПОВЕДЕНЧЕСКИЕ ПАТТЕРНЫ: КОМАНДА, ИТЕРАТОР

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Команда», «Итератор / Индикатор».

ЛАБОРАТОРНАЯ РАБОТА 8. ПОВЕДЕНЧЕСКИЕ ПАТТЕРНЫ: ЦЕПОЧКА ОБЯЗАННОСТЕЙ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттерна «Цепочка обязанностей».

ЛАБОРАТОРНАЯ РАБОТА 9. ПОВЕДЕНЧЕСКИЕ ПАТТЕРНЫ: ПОСРЕДНИК, СНИМОК

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Посредник», «Снимок».

ЛАБОРАТОРНАЯ РАБОТА 10. ПОВЕДЕНЧЕСКИЕ ПАТТЕРНЫ: НАБЛЮДАТЕЛЬ, СОСТОЯНИЕ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Наблюдатель», «Состояние».

ЛАБОРАТОРНАЯ РАБОТА 11. ПОВЕДЕНЧЕСКИЕ ПАТТЕРНЫ: СТРАТЕГИЯ, ШАБЛОННЫЙ МЕТОД

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием паттернов из списка: «Стратегия», «Шаблонный метод».

ЛАБОРАТОРНАЯ РАБОТА 12. АРХИТЕКТУРНЫЕ ПАТТЕРНЫ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием одного из паттернов из списка: MVC, MVP, MVVM, Clean Architecture.

ЛАБОРАТОРНАЯ РАБОТА 13. ПАТТЕРНЫ МИКРОСЕРВИСОВ И РАСПРЕДЕЛЁННЫХ СИСТЕМ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием одного из паттернов из списка: Saga, CQRS, Event Sourcing, Circuit Breaker, API Gateway.

ЛАБОРАТОРНАЯ РАБОТА 14. СИСТЕМНЫЕ ПАТТЕРНЫ И РАБОТА С ДАННЫМИ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием одного из паттернов из списка: Repository, Data Mapper, Active Record, Unit of Work.

ЛАБОРАТОРНАЯ РАБОТА 15. ИДИОМАТИЧЕСКИЕ ПАТТЕРНЫ

1. Задача

Реализуйте решение задачи из предметной области в соответствии с вариантами (пп. 1.4) с использованием одного из паттернов из списка: Producer-Consumer, Thread Pool, Future / Promise.

СПИСОК ЛИТЕРАТУРЫ

Книги

1. Гамма, Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. – Санкт-Петербург : Питер, 2022. – 368 с.

2. Гранди, М. Шаблоны проектирования в Java / М. Гранди ; перевод с английского Ю. Н. Артеменко. – Москва : Вильямс, 2021. – 448 с.

3. Зандстра, М. PHP. Объекты, шаблоны и методики программирования / М. Зандстра. – 5-е изд. – Санкт-Петербург : Диалектика, 2020. – 736 с.

4. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. – Санкт-Петербург : Питер, 2023. – 352 с.

5. Фриман, Э. Head First. Паттерны проектирования / Э. Фриман, Э. Робсон. — 2-е изд. – Санкт-Петербург : Питер, 2024. – 704 с.

6. Evans, E. Domain-Driven Design: Tackling Complexity in the Heart of Software / E. Evans. – Boston : Addison-Wesley, 2020. – 560 p.

7. Fowler, M. Patterns of Enterprise Application Architecture / M. Fowler. – Boston : Addison-Wesley, 2022. – 576 p.

8. Nystrom, R. Game Programming Patterns / R. Nystrom. – Text : electronic // Game Programming Patterns : [site]. – URL: https://gameprogrammingpatterns.com (accessed: 10.09.2026).

Перечень Интернет-ресурсов

1. https://habr.com/ru/articles/210288/ – шпаргалка по паттернам проектирования.

2. https://metanit.com/sharp/patterns/ – материалы по паттернам проектирования на языке C#.