



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра математического обеспечения и стандартизации
информационных технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ
по дисциплине «Тестирование и верификация программного обеспечения»

Практическое занятие № 3

Студент группы *ИКБО-50-23, Каушина А.В.*

(подпись)

Преподаватель *Ильичев Г.П.*

(подпись)

Отчет представлен « » октября 2025 г.

Москва 2025 г.

Цели и задачи

Цель работы: Цель работы состоит в изучении и применении различных подходов к разработке программного обеспечения, основанного на тестировании, для повышения качества, надёжности и поддерживаемости кода.

Для достижения поставленной цели работы студентам необходимо выполнить ряд задач:

- 1) изучить теоретические основы методологий тестирования: TDD, ATDD, BDD и SDD;
- 2) исследовать преимущества и недостатки каждого подхода;
- 3) реализовать практические примеры для каждого метода;
- 4) проанализировать влияние интеграции тестирования на архитектуру и качество программного продукта;
- 5) подготовить итоговый отчёт с выводами и рекомендациями по интеграции подходов в современные процессы разработки.

Практическая часть

06. Система управления библиотекой

Функции — добавление книги, поиск книги, выдача книги читателю, возврат книги.

TDD — тесты для каждой операции (например, проверка наличия книги, обработка ошибок при выдаче).

ATDD — приёмочные тесты, описывающие сценарии выдачи и возврата.

BDD — сценарий «Given книга доступна, When читатель запрашивает книгу, Then книга выдаётся».

SDD — примеры спецификаций с входными данными (ISBN, название) и результатами операций.

1. Реализация с помощью TDD

Реализация с помощью TDD (Test-Driven Development). Были написаны юнит-тесты для ключевых функций системы, реализован минимально необходимый код и проведен рефакторинг. На листинге 1 представлена реализация тестов на языке Java.

Листинг 1 – юнит-тесты для системы управления библиотекой

```
import org.example.Book;
import org.example.LibraryManager;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

public class LibraryManagerTest {

    private LibraryManager library;

    //создание библиотеки перед каждым тестом
    @BeforeEach
    void setUp() {
        library = new LibraryManager();
    }

    //тест добавления книги
    @Test
    void testAddBook() {
        library.addBook(new Book("12345", "Война и мир", "Толстой", 3));
        assertEquals(1, library.getAllBooks().size());
    }

    //тест добавления книг с одинаковым isbn
    @Test
    void testFailAddBook() {
        library.addBook(new Book("12345", "Война и мир", "Толстой", 3));
        library.addBook(new Book("12345", "Мастер и Маргарита", "Булгаков",
7));
        assertEquals(1, library.getAllBooks().size());
    }

    //поиск книги по идентификатору
    @Test
    void testFindBookByISBN() {
        library.addBook(new Book("111", "Мастер и Маргарита", "Булгаков",
2));
        Book found = library.findBookByISBN("111");
        assertNotNull(found);
        assertEquals("Мастер и Маргарита", found.getTitle());
    }

    //тест ошибки поиска
    @Test
    void testFindBookNonExistentISBN() {
        library.addBook(new Book("111", "Мастер и Маргарита", "Булгаков",
2));
        Book found = library.findBookByISBN("222");
        assertNull(found);
    }

    //тест выдачи книги
    @Test
    void testIssueBookToReader() {
        library.addBook(new Book("222", "Преступление и наказание",
"Достоевский", 1));
        boolean issued = library.issueBook("222");
        assertTrue(issued);
        assertEquals(0, library.findBookByISBN("222").getAvailableCopies());
    }

    //тест ошибки выдачи
    @Test
```

```

void testIssueBookNotAvailable() {
    library.addBook(new Book("333", "Евгений Онегин", "Пушкин", 0));
    boolean issued = library.issueBook("333");
    assertFalse(issued);
}

//тест возврата
@Test
void testReturnBook() {
    library.addBook(new Book("444", "Отцы и дети", "Тургенев", 0));
    boolean returned = library.returnBook("444");
    assertTrue(returned);
    assertEquals(1, library.findBookByISBN("444").getAvailableCopies());
}
}

```

На рисунке 1 показано, что большинство тестов завалены из-за отсутствия реализации методов.

```

[INFO]
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running LibraryManagerTest
[ERROR] Tests run: 7, Failures: 2, Errors: 3, Skipped: 0, Time elapsed: 0.067 s <<< FAILURE! -- in LibraryManagerTest

```

Рисунок 1 - провал тестов

В соответствии с методологией TDD на втором этапе был реализован код, представленный на листинге 2.

Листинг 2 – реализация функций системы управления библиотекой

```

public class Book {
    private String isbn;
    private String title;
    private String author;
    private int availableCopies;

    public Book(String isbn, String title, String author, int
availableCopies) {
        this.isbn = isbn;
        this.title = title;
        this.author = author;
        this.availableCopies = availableCopies;
    }

    public String getIsbn() {
        return isbn;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }

    public int getAvailableCopies() {

```

```

        return availableCopies;
    }

    public void issue() {
        if (availableCopies > 0)
            availableCopies--;
    }

    public void returnCopy() {
        availableCopies++;
    }
}

public class LibraryManager {

    private List<Book> books;

    public LibraryManager() {
        books = new ArrayList<>();
    }

    public void addBook(Book book) {
        Book existBook = findBookByISBN(book.getIsbn());
        if (existBook == null) {
            books.add(book);
        }
    }

    public List<Book> getAllBooks() {
        return books;
    }

    public Book findBookByISBN(String isbn) {
        return books.stream()
            .filter(b -> b.getIsbn().equals(isbn))
            .findFirst()
            .orElse(null);
    }

    public boolean issueBook(String isbn) {
        Book book = findBookByISBN(isbn);
        if (book != null && book.getAvailableCopies() > 0) {
            book.issue();
            return true;
        }
        return false;
    }

    public void returnBook(String isbn) {
        Book book = findBookByISBN(isbn);
        if (book != null) {
            book.returnCopy();
            return true;
        }
        return false;
    }
}

```

Запустив тесты (командой `mvn test`), можно убедиться, что они все успешно пройдены (рисунок 2).

```
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running LibraryManagerTest
[INFO] Tests run: 7, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.047 s -- in LibraryManagerTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 7, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.089 s
```

Рисунок 2 - успешное прохождение тестов

2. Реализация с помощью ATDD

Реализация с помощью ATDD (Acceptance Test-Driven Development). Реализованы приёмочные тесты для сценариев использования приложения, которые были согласованы с конечными пользователями.

Сценарий 1. Выдача книги читателю

Предусловие: в библиотеке есть книга с ISBN 111, названием «Мастер и Маргарита» и количеством экземпляров = 3.

Действия:

1. Пользователь открывает раздел «Выдать книгу».
2. Вводит ISBN книги 111.
3. Подтверждает выдачу книги.

Ожидаемый результат: количество доступных экземпляров книги уменьшается на 1, система сообщает: «Книга успешно выдана».

Сценарий 2. Попытка выдать книгу, которой нет в наличии

Предусловие: в библиотеке есть книга с ISBN 222, названием «Преступление и наказание», количество экземпляров = 0.

Действия:

1. Пользователь выбирает «Выдать книгу».
2. Вводит ISBN книги 222.

Ожидаемый результат: система не позволяет выдать книгу, отображается сообщение: «Невозможно выдать книгу».

Сценарий 3. Успешный возврат книги в библиотеку

Предусловие: книга с ISBN 333, названием «Отцы и дети», находится в выданном состоянии (0 доступных экземпляров).

Действия:

1. Пользователь выбирает «Вернуть книгу».
2. Вводит ISBN книги 333.

Ожидаемый результат: количество доступных экземпляров увеличивается на 1, система сообщает: «Книга успешно возвращена».

Сценарий 4. Ошибка при возврате книги (несуществующий ISBN)

Предусловие: в библиотеке есть книги с ISBN: 111, 222, 333.

Действия:

1. Пользователь выбирает «Возврат книги»;
2. Вводит ISBN книги 999.

Ожидаемый результат: возврат не выполняется; система сообщает: «Невозможно вернуть книгу».

Листинг 3 – реализация тестов

```
public class ATDD {  
  
    private LibraryManager library;  
  
    @BeforeEach  
    void setup() {  
        library = new LibraryManager();  
        library.addBook(new Book("111", "Мастер и Маргарита", "Булгаков",  
2));  
        library.addBook(new Book("222", "Преступление и наказание",  
"Достоевский", 0));  
        library.addBook(new Book("333", "Евгений Онегин", "Пушкин", 0));  
    }  
  
    //успешная выдача книги  
    @Test  
    void testIssueAvailableBook() {  
        boolean issued = library.issueBook("111");  
        Book book = library.findBookByISBN("111");  
    }  
}
```

```

        assertTrue(issued, "Книга должна быть успешно выдана");
        assertEquals(1, book.getAvailableCopies(), "Количество доступных
экземпляров должно уменьшиться на 1");
    }

    //ошибка выдачи книги
    @Test
    void testIssueUnavailableBook() {
        boolean issued = library.issueBook("222");
        Book book = library.findBookByISBN("222");

        assertFalse(issued, "Книга не должна быть выдана, если экземпляров
нет");
        assertEquals(0, book.getAvailableCopies(), "Количество экземпляров не
должно измениться");
    }

    //успешный возврат книги
    @Test
    void testReturnBookSuccess() {
        boolean returned = library.returnBook("333");
        Book book = library.findBookByISBN("333");

        assertTrue(returned, "Книга должна быть успешно возвращена");
        assertEquals(1, book.getAvailableCopies(), "Количество доступных
экземпляров должно увеличиться на 1");
    }

    //ошибка возврата книги
    @Test
    void testReturnBookError() {
        boolean returned = library.returnBook("999");
        assertFalse(returned, "Возврат не должен выполняться, если книга не
найдена");
    }
}

```

Запустив тесты, можно убедиться, что они все успешно пройдены (рисунок 3).



```

Terminal Local x + v
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running ATDD
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.042 s -- in ATDD
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----

```

Рисунок 3 - успешное прохождение тестов

3. Реализация с помощью BDD

Созданы сценарии на понятном для всех участников проекта языке (Gherkin), которые описывают поведение системы. Сценарий представлен на листинге 4.

Листинг 4 – сценарий

```
Feature: Issuing a book to a reader

  Scenario: Successfully issuing an available book
    Given there is a book in the library with ISBN "444", title "Отцы и
    дети", author "Тургенев", and 3 available copies
    When the reader requests the book with ISBN "444"
    Then the book is successfully issued
    And the number of available copies becomes 2
```

Автоматизируем реализацию сценария с использованием Cucumber (листинг 5).

Листинг 5 – автоматизация сценария

```
public class BDD {
    private LibraryManager libraryManager;
    private boolean issueResult;
    private Book book;

    @Given("there is a book in the library with ISBN {string}, title
    {string}, author {string}, and {int} available copies")
    public void bookExists(String isbn, String title, String author, int
    copies) {
        libraryManager = new LibraryManager();
        book = new Book(isbn, title, author, copies);
        libraryManager.addBook(book);
    }

    @When("the reader requests the book with ISBN {string}")
    public void requestBook(String isbn) {
        issueResult = libraryManager.issueBook(isbn);
    }

    @Then("the book is successfully issued")
    public void bookIssued() {
        assertTrue(issueResult, "Книга должна быть выдана");
    }

    @Then("the number of available copies becomes {int}")
    public void changeAvailableCopies(int expectedCopies) {
        Book foundBook = libraryManager.findBookByISBN(book.getIsbn());
        assertEquals(expectedCopies, foundBook.getAvailableCopies(),
        "Количество копий должно совпадать");
    }
}
```

Запустив сценарий, можно убедиться, что тест успешно пройден (рисунок 4).

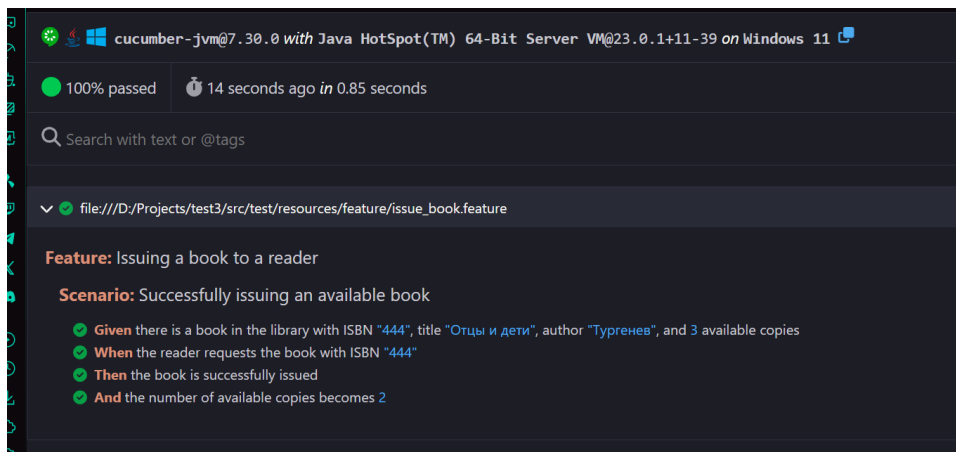


Рисунок 4 - успешное завершение сценария

4. Реализация с помощью SDD

Реализация с помощью SDD (Specification by Example). В проекте созданы спецификации требований с использованием конкретных примеров, которые затем были преобразованы в автоматизированные тесты. Спецификации для подхода SDD описаны в виде таблицы 1.

Таблица 1 – спецификации для подхода SDD

ISBN	Название книги	Автор	Доступно копий	Действие	Ожидаемый результат	Доступно копий после
111	Мастер и Маргарита	Булгаков	3	Выдать книгу	Успешно	2
111	Мастер и Маргарита	Булгаков	1	Выдать книгу	Успешно	0
111	Мастер и Маргарита	Булгаков	0	Выдать книгу	Неудача	0
333	Евгений Онегин	Пушкин	2	Вернуть книгу	Успешно	3
999	Несуществующая	Автор	—	Выдать книгу	Неудача	—

На основе данных сформирован CSV-файл для работы тестов (рисунок 5).

	c1	c2	c3	c4	c5	c6	c7
1	isbn	title	author	initial_copies	action	expected_result	final_copies
2	111	Мастер и Маргарита	Булгаков	3	issue	success	2
3	111	Мастер и Маргарита	Булгаков	1	issue	success	0
4	111	Мастер и Маргарита	Булгаков	0	issue	failure	0
5	333	Евгений Онегин	Пушкин	2	return	success	3
6	999	Несуществующая	Автор	-1	issue	failure	-

Рисунок 5 - спецификация в CSV

На основе спецификаций были написаны автоматизированные тесты (листинг 6).

Листинг 6 – тесты по спецификации SDD

```
public class SDD {

    @ParameterizedTest(name = "[{index}] {0}: {4} → {5}")
    @CsvFileSource(
        resources = "/book_operations.csv",
        numLinesToSkip = 1,
        delimiter = ','
    )
    void shouldHandleBookOperations(
        String isbn,
        String title,
        String author,
        String initialCopiesStr,
        String action,
        String expectedResult,
        String finalCopiesStr) {

        LibraryManager manager = new LibraryManager();

        int initialCopies;
        if (initialCopiesStr == null || initialCopiesStr.isBlank() || "-".equals(initialCopiesStr.trim())) initialCopies = -1;
        else initialCopies = Integer.parseInt(initialCopiesStr.trim());

        if (initialCopies >= 0) {
            Book book = new Book(isbn, title, author, initialCopies);
            manager.addBook(book);
        }

        boolean result = "issue".equals(action) ? manager.issueBook(isbn) : manager.returnBook(isbn);

        boolean expectedBool = "success".equals(expectedResult);
        assertEquals(expectedBool, result,
            () -> String.format("Ожидался %s для действия '%s' на ISBN %s", expectedResult, action, isbn));

        if (!"-".equals(finalCopiesStr)) {
            int expectedCopies = Integer.parseInt(finalCopiesStr);
            Book book = manager.findBookByISBN(isbn);
            if (book != null) {
                assertEquals(expectedCopies, book.getAvailableCopies(),
                    () -> String.format("После %s должно остаться %d копий", action, expectedCopies));
            }
        }
    }
}
```

Запустив тесты, можно убедиться, что все они пройдены (рисунок 6).







Test Cases		
[Summary] [Package List] [Test Cases]		
Issuing a book to a reader		
	Successfully issuing an available book	0.069 s
SDD		
	shouldHandleBookOperations(String, String, String, String, String, String, String, String)[1]	0.102 s
	shouldHandleBookOperations(String, String, String, String, String, String, String, String)[2]	0.004 s
	shouldHandleBookOperations(String, String, String, String, String, String, String, String)[3]	0.004 s
	shouldHandleBookOperations(String, String, String, String, String, String, String, String)[4]	0.009 s
	shouldHandleBookOperations(String, String, String, String, String, String, String, String)[5]	0.004 s

Рисунок 6 - результат прохождения тестов

Заключение

В данной работе реализована система управления библиотекой с поддержкой четырёх ключевых функций: добавление книги, поиск по ISBN, выдача книги читателю и возврат книги. Применены четыре современных подхода к разработке через тестирование:

1. TDD: написаны юнит-тесты для каждой операции реализована базовая функциональность и выполнен рефакторинг кода.
2. ATDD: разработаны приёмочные тесты, согласованные с бизнес-логикой. Они описывают ключевые пользовательские сценарии: «Читатель получает книгу, если она доступна», «Выдача отклоняется при отсутствии копий», «Возврат увеличивает количество доступных экземпляров», «Возврат недоступен при отсутствии книги в библиотеке».
3. BDD: сформулированы сценарии на языке Gherkin, понятные как разработчикам, так и бизнес-пользователям.
4. SDD: составлена спецификация в виде таблицы с конкретными примерами (ISBN, название, автор, начальное количество копий, действие, ожидаемый результат, количество копий после операции), позволяющая создать живую документацию и автоматизированные тесты на её основе.

Итоговая реализация демонстрирует комплексное применение TDD, ATDD, BDD и SDD на одном проекте. Такой подход гарантирует корректность логики на всех уровнях, обеспечивает читаемость и согласованность требований, создаёт живую документацию, повышает надежность и упрощает поддержку.