УДК 004.415.533

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЭФФЕКТИВНОСТИ PAGE OBJECT И SCREENPLAY ПАТТЕРНОВ ПРИ АВТОМАТИЗАЦИИ СЛОЖНЫХ СЦЕНАРИЕВ В БИЗНЕС-ПРИЛОЖЕНИЯХ

Шипуль А.Р.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Косарева Е.М. – ассистент кафедры ПИКС

Аннотация. В статье рассматриваются архитектурные подходы к построению автоматизированных тестов пользовательского интерфейса – Page Object Model и Screenplay. Актуальность исследования обусловлена усложнением корпоративных бизнес-приложений и необходимостью обеспечения масштабируемости и сопровождаемости тестовой архитектуры. Проведен сравнительный анализ паттернов по критериям связности компонентов, повторного использования кода, устойчивости к изменениям интерфейса и прозрачности бизнес-логики. Установлено, что при автоматизации сложных многошаговых сценариев Screenplay обеспечивает более гибкую и устойчивую архитектуру по сравнению с классической моделью Page Object.

Ключевые слова: автоматизация тестирования, Page Object Model, Screenplay, бизнес-приложения, UI-тестирование, масштабируемость

Введение. Усложнение архитектуры современных ERP-, CRM- и банковских систем с их многоэтапными формами и динамическими элементами делает автоматизацию тестирования критическим фактором обеспечения качества.

Одной из ключевых проблем при автоматизации сложных сценариев является поддерживаемость тестов. Изменения структуры интерфейса, добавление промежуточных шагов или модификация бизнес-логики часто приводят к необходимости переработки большого объема тестового кода. Для минимизации подобных рисков применяются архитектурные паттерны проектирования тестов. Наиболее распространенным из них является Page Object Model, описанный в рекомендациях проекта Selenium [1]. Альтернативным подходом выступает паттерн Screenplay, активно развиваемый в рамках фреймворка Serenity BDD [2].

Целью работы является сравнительный анализ эффективности указанных паттернов при автоматизации сложных сценариев в бизнес-приложениях.

Основная часть. При автоматизации тестирования пользовательских интерфейсов корпоративных информационных систем ключевое значение приобретает выбор архитектурного подхода к построению тестовой системы. В условиях роста числа модулей, усложнения бизнес-процессов и регулярных изменений интерфейса именно структура тестовой архитектуры определяет её сопровождаемость и устойчивость к изменениям.

Сложный сценарий здесь определяется как многошаговый процесс (например, оформление финансовой транзакции) с разветвленной логикой, зависимостью от ролей пользователя и передачей состояния между экранами.

Сравнительный анализ архитектурных подходов выполняется по следующим критериям:

– связанность компонентов тестовой системы;

– возможность повторного использования кода;

– устойчивость к изменениям пользовательского интерфейса;

– прозрачность и читаемость бизнес-логики сценариев;

– масштабируемость архитектуры при увеличении числа сценариев.

Одним из наиболее распространённых подходов к организации UI-тестов является Page Object Model. Данный паттерн предполагает представление каждой страницы веб-приложения в виде отдельного программного класса, инкапсулирующего элементы интерфейса и методы

взаимодействия с ними [1]. Тестовый сценарий формируется посредством последовательного обращения к соответствующим объектам страниц.

Структурная организация тестовой системы при использовании Page Object Model (рисунок 1):

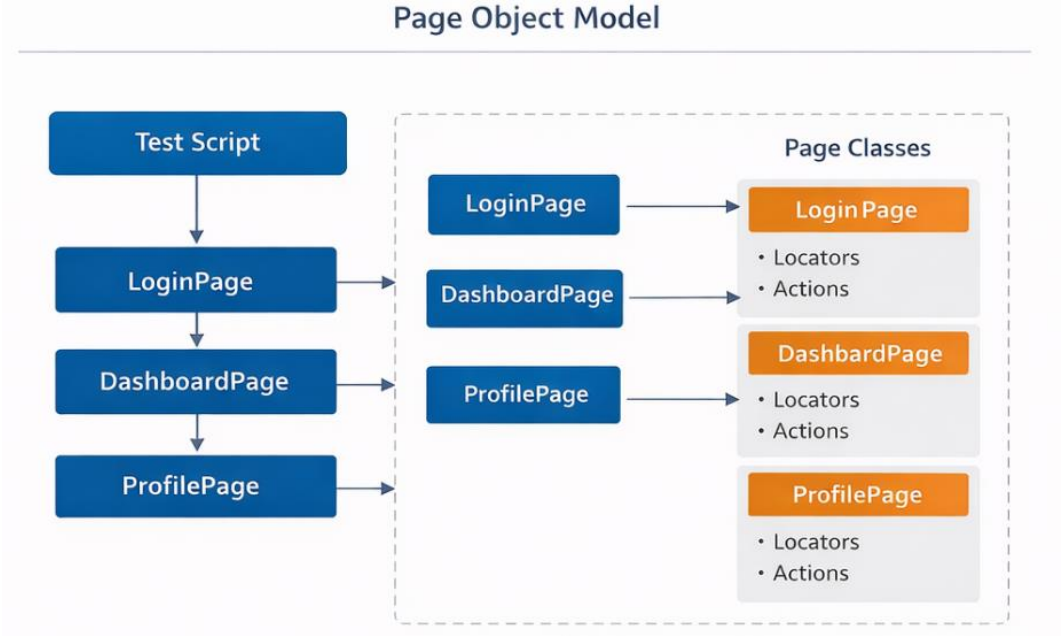

Рисунок 1 – Структурная схема взаимодействия тестового сценария с Page Object Model

Применение Page Object Model позволяет централизовать локаторы элементов, сократить дублирование кода и обеспечить базовую структурированность тестовой архитектуры. На начальном этапе внедрения автоматизации POM демонстрирует высокую эффективность благодаря простоте понимания и относительно невысоким требованиям к архитектурному проектированию. Особенно оправданным данный подход является в проектах с ограниченным числом экранов и линейной логикой пользовательских сценариев.

Кроме того, Page Object способствует разделению тестовой логики и технической реализации взаимодействия с интерфейсом. Это упрощает адаптацию тестов при незначительных изменениях DOM-структуры страницы. В небольших командах такой подход снижает порог входа в автоматизацию, поскольку структура «одна страница – один класс» интуитивно понятна даже начинающим инженерам.

Однако при масштабировании проекта начинают проявляться структурные ограничения данной модели. По мере увеличения количества страниц и сценариев классы Page Object усложняются, аккумулируя в себе как описание элементов, так и последовательность бизнес-действий. В результате один класс может содержать десятки методов, отражающих различные варианты использования страницы. Это приводит к нарушению принципа единственной ответственности и к постепенному росту связанности между компонентами тестовой системы [6].

Высокая связанность выражается в том, что изменение логики взаимодействия на одной странице способно повлиять на несколько тестовых сценариев одновременно. При наличии цепочек переходов между страницами формируются неявные зависимости, усложняющие рефакторинг. В условиях частых релизов и регулярных изменений интерфейса это приводит к увеличению времени поддержки тестов.

Дополнительной проблемой становится повторное использование действий. Например, операция поиска клиента или авторизации пользователя может присутствовать в нескольких модулях системы. В рамках Page Object такие действия зачастую реализуются в разных

классах страниц, что приводит к дублированию кода и повышению вероятности расхождения реализаций. Аналогичные проблемы сопровождения тестовых архитектур описаны в работах по шаблонам тестирования [3, 4].

Альтернативным подходом является паттерн Screenplay, основанный на модели взаимодействия «Актёр – Задача – Взаимодействие» [2, 5]. В данном случае тест описывает поведение пользователя, а не структуру страницы. Центральным элементом архитектуры выступает актёр, обладающий определёнными способностями и выполняющий задачи, состоящие из атомарных взаимодействий.

Общая схема организации тестовой системы в соответствии с паттерном Screenplay (рисунок 2):

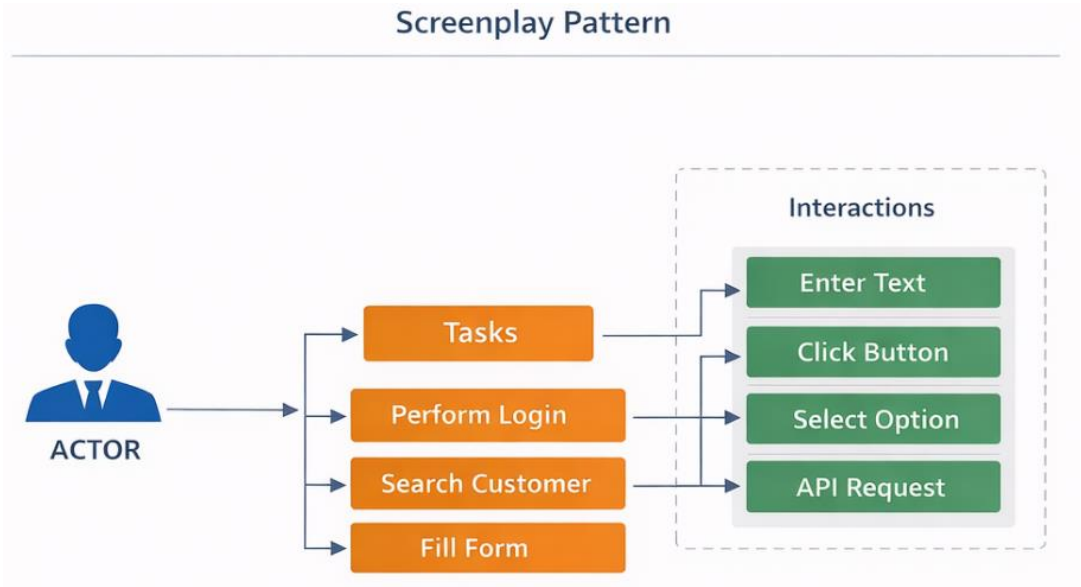

Рисунок 2 – Структурная схема архитектуры Screenplay

Screenplay обеспечивает глубокую декомпозицию: действия становятся переиспользуемыми атомарными блоками, абстрагированными от конкретных страниц до уровня бизнес-операций.

Данный подход согласуется с принципами объектно-ориентированного проектирования и способствует лучшей модульности системы. Изменение интерфейса требует корректировки только соответствующих взаимодействий или описаний элементов, не затрагивая бизнес-задачи целиком. Это существенно сокращает область влияния изменений и уменьшает риск возникновения каскадных ошибок в тестах.

Дополнительным преимуществом является повышение прозрачности сценариев. Тестовые отчеты формируются как последовательность логически завершенных действий пользователя, что делает их понятными не только разработчикам, но и аналитикам или менеджерам проекта.

Сравнительный анализ показывает, что Page Object является эффективным решением для небольших проектов с ограниченным числом сценариев и стабильным интерфейсом. Он обеспечивает быстрый старт автоматизации и минимальные архитектурные издержки. В то же время при автоматизации крупных бизнес-приложений с длительным жизненным циклом Screenplay демонстрирует более высокую устойчивость к изменениям и лучшую масштабируемость. Исследования в области автоматизации подтверждают, что более глубокая декомпозиция тестовой логики способствует снижению затрат на сопровождение в долгосрочной перспективе [7, 8].

Следует отметить, что внедрение Screenplay требует предварительного архитектурного проектирования и более высокой квалификации команды. На начальном этапе это может

увеличить трудозатраты и усложнить структуру проекта. Однако при длительном сопровождении системы данный подход позволяет избежать накопления технического долга, характерного для чрезмерно разросшихся Page Object-классов.

Таким образом, различия между рассматриваемыми паттернами проявляются прежде всего при работе со сложными, многошаговыми бизнес-сценариями, где критическое значение приобретают модульность, переиспользование и устойчивость к изменениям.

Заключение. В результате проведенного анализа установлено, что выбор между Page Object Model и Screenplay зависит от масштаба проекта и стратегических целей автоматизации. Page Object остается рациональным решением при ограниченных ресурсах и относительно простой структуре приложения.

При автоматизации сложных многошаговых сценариев в корпоративных системах Screenplay обеспечивает более гибкую архитектуру, снижает связанность компонентов и повышает устойчивость тестов к изменениям пользовательского интерфейса. Таким образом, для крупных бизнес-приложений с длительным циклом сопровождения применение Screenplay является более эффективным с точки зрения масштабируемости и поддерживаемости автоматизированной тестовой системы.

Список литературы

1. Freeman, S. Growing Object-Oriented Software, Guided by Tests / S. Freeman, N. Pryce. – Boston : Addison-Wesley, 2009. – 768 p. – ISBN 978-0-321-50362-6.

2. Meszaros, G. xUnit Test Patterns: Refactoring Test Code / G. Meszaros. – Boston : Addison-Wesley, 2007. – 883 p. – ISBN 978-0-321-20790-9.

3. Crispin, L. Agile Testing: A Practical Guide for Testers and Agile Teams / L. Crispin, J. Gregory. – Boston : Addison-Wesley, 2009. – 536 p. – ISBN 978-0-321-53446-0.

4. Fewster, M. Software Test Automation: Effective Use of Test Execution Tools / M. Fewster, D. Graham. – Harlow : Addison-Wesley, 1999. – 592 p. – ISBN 978-0-201-33140-3.

5. Smart, J. F. BDD in Action: Behavior-Driven Development for the Whole Software Lifecycle / J. F. Smart. – Shelter Island : Manning Publications, 2015. – 440 p. – ISBN 978-1-61729-165-4.

6. Martin, R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design / R. C. Martin. – Boston : Prentice Hall, 2017. – 432 p. – ISBN 978-0-13-449416-6.

7. Fowler, M. Patterns of Enterprise Application Architecture / M. Fowler. – Boston : Addison-Wesley, 2002. – 560 p. – ISBN 978-0-321-12742-6.

8. Selenium Documentation. Page Object Models / Selenium Project // Selenium Documentation [Электронный ресурс]. – Режим доступа: https://www.selenium.dev/documentation/test_practices/encouraged/page_object_models/ (дата обращения: 02.03.2026).

9. Serenity BDD Screenplay Pattern / Serenity BDD // Serenity Documentation [Электронный ресурс]. – Режим доступа: https://serenity-bdd.info/docs/screenplay (дата обращения: 02.03.2026).

UDC 004.415.533

COMPARATIVE ANALYSIS OF THE EFFECTIVENESS OF PAGE OBJECT AND SCREENPLAY PATTERNS IN AUTOMATING COMPLEX SCENARIOS IN BUSINESS APPLICATIONS

Shipul A.R.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Scientific advisor: Kosareva E.M. – Assistant Lecturer, Department of ICSD

Annotation. The article examines architectural approaches to building automated user interface tests, namely the Page Object Model and the Screenplay pattern. The relevance of the study is determined by the increasing complexity of corporate business applications and the need to ensure scalability and maintainability of test architecture. A comparative analysis of the patterns was carried out based on the criteria of component coupling, code reusability, resistance to interface changes, and transparency of business logic. It has been established that when automating complex multi-step scenarios, the Screenplay pattern provides a more flexible and sustainable architecture compared to the classical Page Object model.

Keywords: test automation, Page Object Model, Screenplay, business applications, UI testing, scalability