УДК 004.51

ИССЛЕДОВАНИЕ СОВРЕМЕННЫХ ОТЕЧЕСТВЕННЫХ МЕТОДОВ МЕНЕДЖМЕНТА СОСТОЯНИЯ

Юлкина Н.С.[1]
Научный руководитель – Пышкина Н. Ю.[1]
[1]ГУМРФ имени адмирала С.О. Макарова
otd_o@gumrf.ru

Введение

Менеджер состояний – один из элементов браузерного приложения, необходимый для управления данными на клиентской стороне. По мере масштабирования приложения разрозненное хранение и обработка данных критически снижают управляемость информационных потоков и усложняют архитектуру системы. Менеджер состояний позволяет обеспечить консистентное, детерминированное и надежное управление большим объёмом данных [1]. Часть данных должна храниться исключительно на клиентской части приложения, при этом их состояние должно быть достаточно актуальным, не противоречить серверному состоянию в данный конкретный момент. Слой управления состоянием имеет критическое значение, поскольку непосредственно определяет данные, доступные для отображения в браузере пользователя. В связи с этим выбор программного решения для работы с состоянием требует методологически обоснованного подхода.

Основная часть

При выборе инструмента управления состоянием приоритет должен оставаться за решениями с открытым исходным кодом, что обеспечивает необходимый уровень прозрачности. При этом географический и политический контекст размещения команды разработчиков становится важным фактором стратегического планирования, влияющим на долгосрочную доступность, поддержку и развитие выбранной технологии.

В рамках исследования технологического импортозамещения следует отметить, что использование программного обеспечения иностранного происхождения сопряжено с рядом системных рисков, среди которых особое место занимает угроза юридической блокировки доступа к ресурсам разработки. Также блокировка доступа российских разработчиков к централизованным репозиториям (например, npm) делает невозможным получение обновлений, в том числе устраняющих выявленные недостатки системы защиты. Следствием такой изоляции становится как прекращение сопровождения разрабатываемого программного обеспечения, так и неконтролируемое накопление в нём дефектов безопасности. Загрузка корректирующих релизов, необходимых для устранения уязвимых состояний, при этом полностью исключена.

Кроме того, для проектов, работающих с чувствительными данными или функционирующих в регулируемых отраслях, ключевым требованием является использование доверенного программного обеспечения, гарантии которого зарубежные решения предоставить не могут в виду невозможности законодательного влияния на них.

В связи с этим, данное исследование ставит своей целью провести сравнительный анализ наиболее перспективных отечественных решений для управления состоянием приложений на предмет их способности стать полноценной заменой текущему отраслевому стандарту, представленному иностранной библиотекой Redux. Методология исследования включает практическую реализацию типового веб-приложения с применением каждой из рассматриваемых библиотек, анализ их функциональных возможностей, а также выявление потенциальных ограничений в сравнении с референсным решением.

Исследование проводится на основе следующих отечественных решений: reatom, effector, storeon. Библиотеки reatom и storeon являются минималистичными решениями, позволяющими писать атомарный код без дублирования, а effector - флагманское реактивное решение, подходящее для написания довольно сложной логики [2].

По итогу написания проектов с использованием представленных библиотек оценивается: размер итоговых проектов, качество кода, количество строк кода, а также функциональность. Каждая из описанных библиотек хорошо справляется с задачей управления состояний с упором на свою нишу. В то время как Effector предоставляет более широкий функционал, storeon делает код минималистичным [3], а reatom позволяет улучшить опыт разработчика при помощи понятной реактивной модели, атомарного дизайна и нативной поддержки typescript [4].

Выводы

Проведено исследование посредством анализа и тестирования различных современных решений для организации управления состоянием веб-приложения.

Литература

1. Документация React [Электронный ресурс]. URL: https://reactdev.ru/learn/managing-state/ (дата обращения: 10.02.2026).
2. Документация разработчика Effector [Электронный ресурс]. URL: https://effector.dev/ru/ (дата обращения: 10.02.2026).
3. Документация разработчика Storeon [Электронный ресурс]. URL: https://github.com/storeon/storeon (дата обращения: 10.02.2026).
4. Документация разработчика Reatom [Электронный ресурс]. URL: https://v1000.reatom.dev/ (дата обращения: 10.02.2026).