УДК 004.514:004.451

ПРОЕКТИРОВАНИЕ ПРОИЗВОДИТЕЛЬНОГО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА В HARMONY OS НА ОСНОВЕ ДЕКЛАРАТИВНОГО ПОДХОДА ARK UI

Орлова А.В.[1]
Научный руководитель – канд. техн. наук Балакшин П.В.[1]
[1]Университет ИТМО
nastena.v.orlova@mail.ru

Введение

Современные мобильные и распределённые операционные системы требуют высокой интерактивности пользовательского интерфейса при ограниченных вычислительных и энергетических ресурсах. В HarmonyOS, разработанной компанией Huawei, применяется декларативный подход к построению UI, реализованный во фреймворке ArkUI, основанный на реактивной модели обновления состояния.

Несмотря на преимущества декларативной парадигмы, при росте структурной и визуальной сложности интерфейса возникает проблема обеспечения стабильной производительности. Избыточные реактивные зависимости и глубокая вложенность компонентов могут приводить к снижению частоты кадров, увеличению времени отклика и росту энергопотребления. В зарубежной практике вопросы оптимизации декларативных интерфейсов активно исследуются в рамках таких фреймворков как Flutter, React и SwiftUI. Однако специфика распределённой архитектуры HarmonyOS и разнообразие поддерживаемых устройств требуют дополнительной адаптации этих подходов. В отечественных исследованиях влияние архитектуры ArkUI на производительность UI остаётся недостаточно систематизированным.

Таким образом, актуальной научной задачей является разработка модели проектирования производительного пользовательского интерфейса в HarmonyOS, обеспечивающей баланс между декларативной гибкостью и эффективным использованием вычислительных ресурсов.

Основная часть

1. Предлагаемое решение направлено на повышение производительности пользовательского интерфейса в HarmonyOS за счёт системного проектирования UI на основе декларативной модели ArkUI [1, 2]. Ключевая идея заключается не в точечной оптимизации отдельных компонентов, а в формировании архитектурной модели интерфейса, которая изначально учитывает особенности реактивного обновления, жизненного цикла компонентов и механизмов рендеринга платформы [1]. В декларативной среде интерфейс является функцией состояния. Следовательно, основным источником потерь производительности становится не сам процесс отрисовки, а избыточные изменения состояния и неконтролируемое распространение обновлений по дереву компонентов [2, 3].

Предлагаемое решение основывается на трёх взаимосвязанных принципах:

1. **Локализация состояния** — ограничение области влияния изменений за счёт декомпозиции интерфейса на независимые функциональные модули [4].
2. **Контролируемая реактивность** — минимизация каскадных перерисовок через проектирование мелкозернистых компонентов [3].
3. **Адаптивная сложность интерфейса** — динамическая настройка визуальных эффектов, анимаций и глубины вложенности в зависимости от производительности устройства [5].

Таким образом, предлагается архитектурно-ориентированный подход, при котором производительность становится проектным требованием на этапе моделирования интерфейса, а не задачей последующей оптимизации.

2. Проблема снижения производительности в декларативных интерфейсах возникает при увеличении сложности UI и масштабировании приложения на устройства с различными аппаратными характеристиками [3]. Оптимальным решением является внедрение многоуровневой модели проектирования, включающей:

- **уровень архитектуры** — разделение интерфейса на функциональные домены с автономным состоянием [4];
- **уровень компонентов** — проектирование лёгких и переиспользуемых UI-элементов [1];
- **уровень исполнения** — минимизация операций в UI-потоке и перенос вычислений в фоновые процессы;
- **уровень адаптации** — механизм профилирования устройства и динамической настройки сложности интерфейса [5].

В рамках данного подхода интерфейс проектируется как масштабируемая система с возможностью деградации визуальных эффектов без потери функциональности. Оптимальность решения заключается в его универсальности: оно применимо как к мобильным устройствам, так и к IoT-сценариям, где ресурсы ограничены. Кроме того, решение не требует модификации системных механизмов платформы и реализуется средствами самого ArkUI, что обеспечивает экономичность внедрения [1-2, 5].

3. Предложенный подход объединяет архитектурные принципы проектирования и методы анализа производительности, адаптированные к особенностям ArkUI и HarmonyOS [1, 2]. Его оригинальность заключается в переходе от локальной оптимизации к системному управлению реактивностью интерфейса.

Экономическая эффективность достигается за счёт:

- отсутствия необходимости использования сторонних инструментов [4];
- сокращения времени отладки;
- уменьшения затрат на поддержку и масштабирование приложения.

Практическое применение возможно при разработке адаптивных интерфейсов для распределённых систем и устройств с различной вычислительной мощностью.

Выводы

В работе обоснована необходимость перехода от локальной оптимизации компонентов к системному архитектурному проектированию пользовательского интерфейса в HarmonyOS на основе декларативной модели ArkUI. Показано, что ключевым фактором производительности является не только скорость рендеринга, но и характер управления состоянием и распространения реактивных обновлений.

Предложенная многоуровневая модель проектирования позволяет минимизировать избыточные перерисовки, стабилизировать работу UI на устройствах с различными аппаратными характеристиками и обеспечить масштабируемость интерфейса без потери функциональности.

Таким образом, разработанный подход формирует методическую основу для создания производительных, адаптивных и экономически эффективных интерфейсов в среде HarmonyOS и может быть использован при проектировании современных распределённых и мобильных приложений.

Литература

1. Huawei Device Co., Ltd. ArkUI Development Guide. – Официальная документация [Электронный ресурс]. – Режим доступа: https://developer.huawei.com/consumer/en/doc/arkui (дата обращения: 09.02.2026).

2. Huawei Device Co., Ltd. HarmonyOS Application Development Overview. [Электронный ресурс]. – Режим доступа: https://developer.huawei.com/consumer/en/doc/harmonyos-guides/arkts-ui-development-overview (дата обращения: 11.02.2026).
3. Zhang L., Li H. Reactive UI Frameworks and Performance Optimization Strategies // Journal of Software Engineering. – 2022.
4. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. – Addison-Wesley, 1994.
5. Tanenbaum A., Van Steen M. Distributed Systems: Principles and Paradigms. – Pearson, 2017.